

# Как сделать игру на Unity

---

Лунев Артем

Инди разработчик

2



# Обо мне

- Занимаюсь геймдевом в качестве хобби
- 2 проекта в релизе
- Тяжелый опыт ошибок планирования
- Пару раз фрилансил
- Работал в команде

Зачем это все?



# Что такое игра

- Игра – форма деятельности в условиях ситуациях, направленная на воссоздание и усвоение общественного опыта, фиксированного в социально закрепленных способах осуществления предметных действий, в предметах науки и культуры (с) Википедия



С чего начнем?

В детстве мечтал  
делать игры



На самом деле...



ТЕПЕРЬ Я ИХ ДЕЛАЮ

# Море бумаги



# Дизайн документ

- 
- 
- Концепт-документ
  - Функциональная спецификация

# Концепт-документ

---

- Краткое и ёмкое описание концепции (идеи) игры, то есть, максимально сжатый документ, в котором рассказывается о том, какой будет игра, чем она будет интересна и как она должна выглядеть после разработки

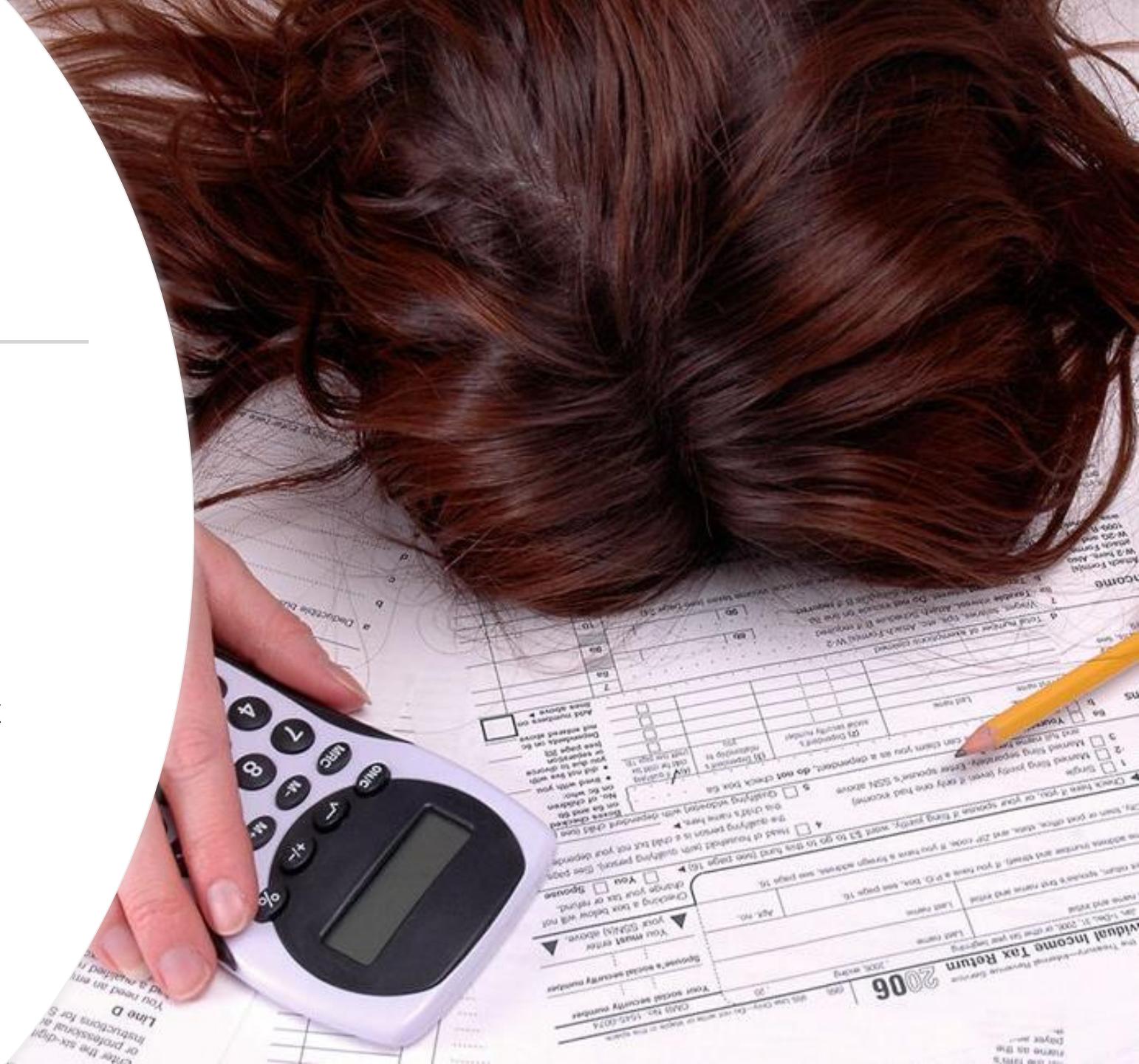
# Функциональная спецификация

---

- Формальное описание программного продукта, которое используется в качестве плана при создании программы. В минимальном виде функциональная спецификация должна четко определить цель создания продукта (его функцию)

## Зачем нужен диздок

- Поможет в декомпозиции задач
- Поможет приблизительно оценить бюджет
- Поможет оценить количество людей
- Поможет оценить время разработки
- Упростит объяснение задач для новых людей



# На что обратить внимание при разработке геймдизайна

---

# Ядро игры



Сюжет, если  
сюжетная игра



# Усложнение игры и развитие игрока



# Вызов для игрока и поощрение



## Что почитать

---

ВПЕРВЫЕ ВВЕДЕНИЕ В ГЕЙМДИЗАЙН, ПРОТОТИПИРОВАНИЕ И ГЕЙМДЕВ  
ОБЪЕДИНЕНЫ В ОДНУ КНИГУ

UNITY и C#  
ГЕЙМДЕВ от ИДЕИ  
до РЕАЛИЗАЦИИ

второе издание

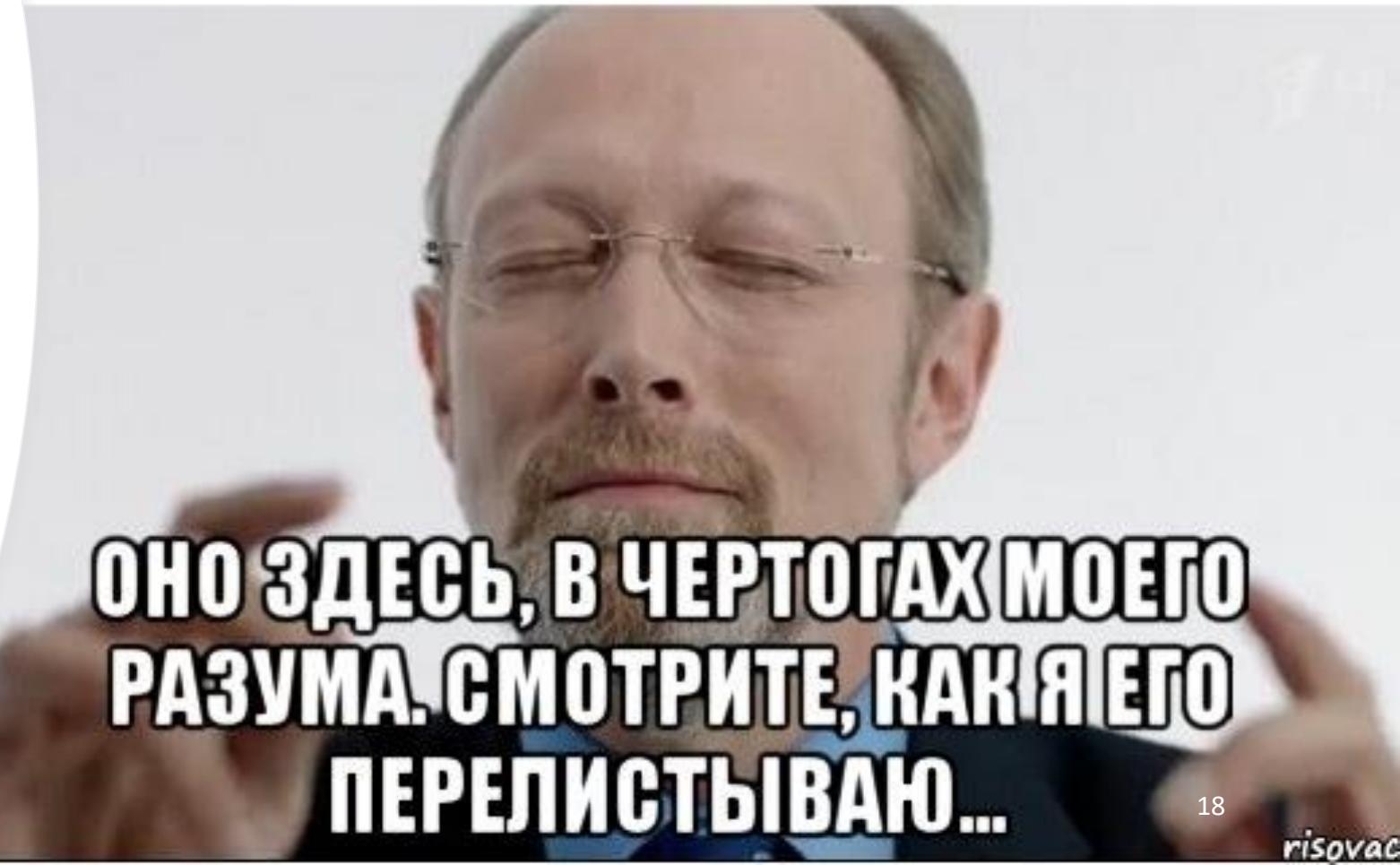


Джереми Гибсон **БОНД**

Предисловие Ричарда Лемарчанда

ГДЕ ТЗР

## Техническое задание



ОНО ЗДЕСЬ, В ЧЕРТОГАХ МОЕГО  
РАЗУМА. СМОТРИТЕ, КАК Я ЕГО  
ПЕРЕЛИСТЫВАЮ...

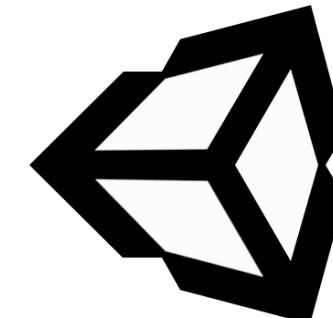
# Выбор игрового движка



Из чего можно  
выбрать?



**Construct 2**

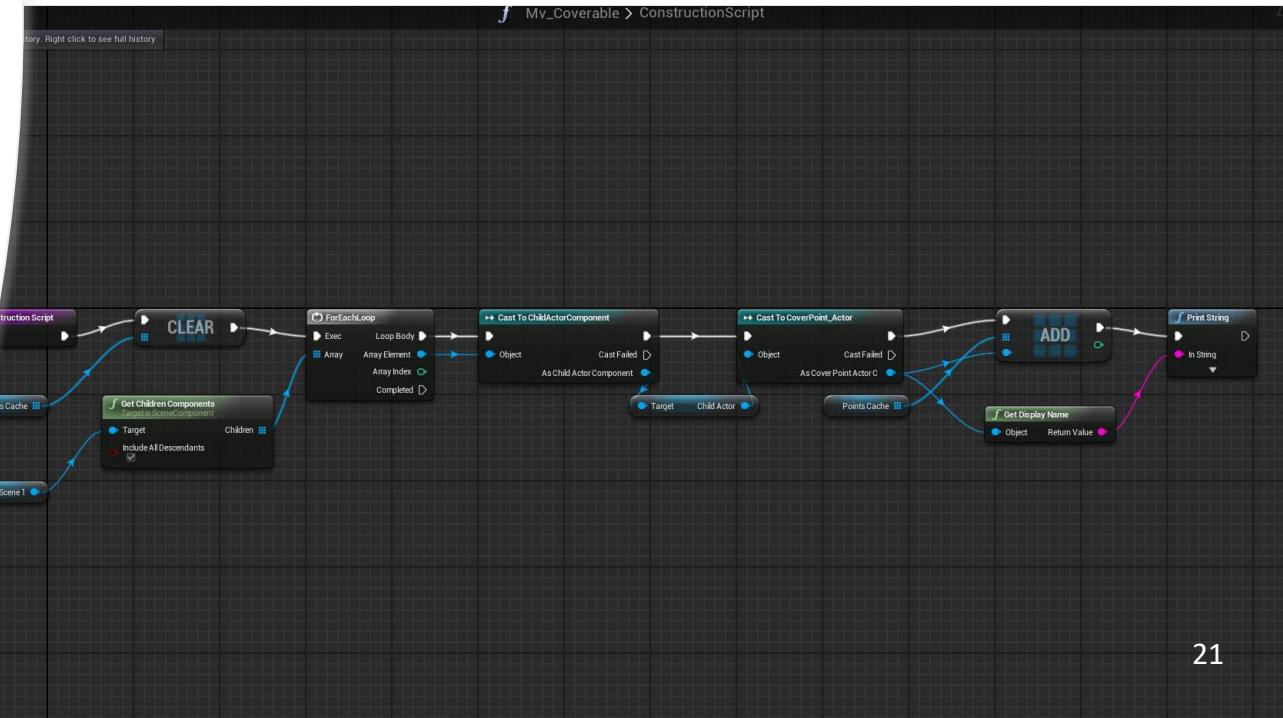


**Unity 3D**



## Плюсы Unreal Engine 4

- + Фотореализм из коробки
- + Движок заточен под AAA проекты
- +Blue prints +Кроссплатформенность
- +Очень гибкий



# Минусы Unreal Engine 4

- Код пишется на плюсах
- Высокий порог входления
- Сложная в освоении документация
- BluePrints может быть и минусом
- Множество сложных инструментов

This is a *header* file. You can think of it as being sort of like a table of contents for a C++ class. Before we can start building any new functionality, we must declare any file.

3. Add the following code underneath the declaration for **AFloatingActor**:

**Compile and Test Your C++ Code**

Save your work in both `FloatingActor.h` and `FloatingActor.cpp`. Then, in the **Solution Explorer**, right-click the project, click the **Build** command in the context menu, and wait for the build process to complete.

**Event Graph**

**Other Weapon Effects**

AV\_Character > Other Weapon Effects

Output from: Build

```
Using UnrealHeaderTool "D:\UnrealEngine\Projects\QuickStart\QuickStart".
Generation code generated for QuickStartEditor in 5.5724689 seconds
Visual Studio 2017 14.16.27023 toolchain (C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\VC\Tools\MSVC\14.16.27023\bin\Hostx86\x86)
Building 6 actions to XGE
-----Project: Default-----
start.init.gen.cpp (0:01.32 at +0:00)
gActor.cpp (0:01.45 at +0:00)
gActor.gen.cpp (0:01.42 at +0:00)
or-Quickstart-0002.lib (0:00.34 at +0:01)
```

Yoda's head on a green background with the text 'УЧИТЬ С++ ДОЛЖЕН ТЫ' overlaid.

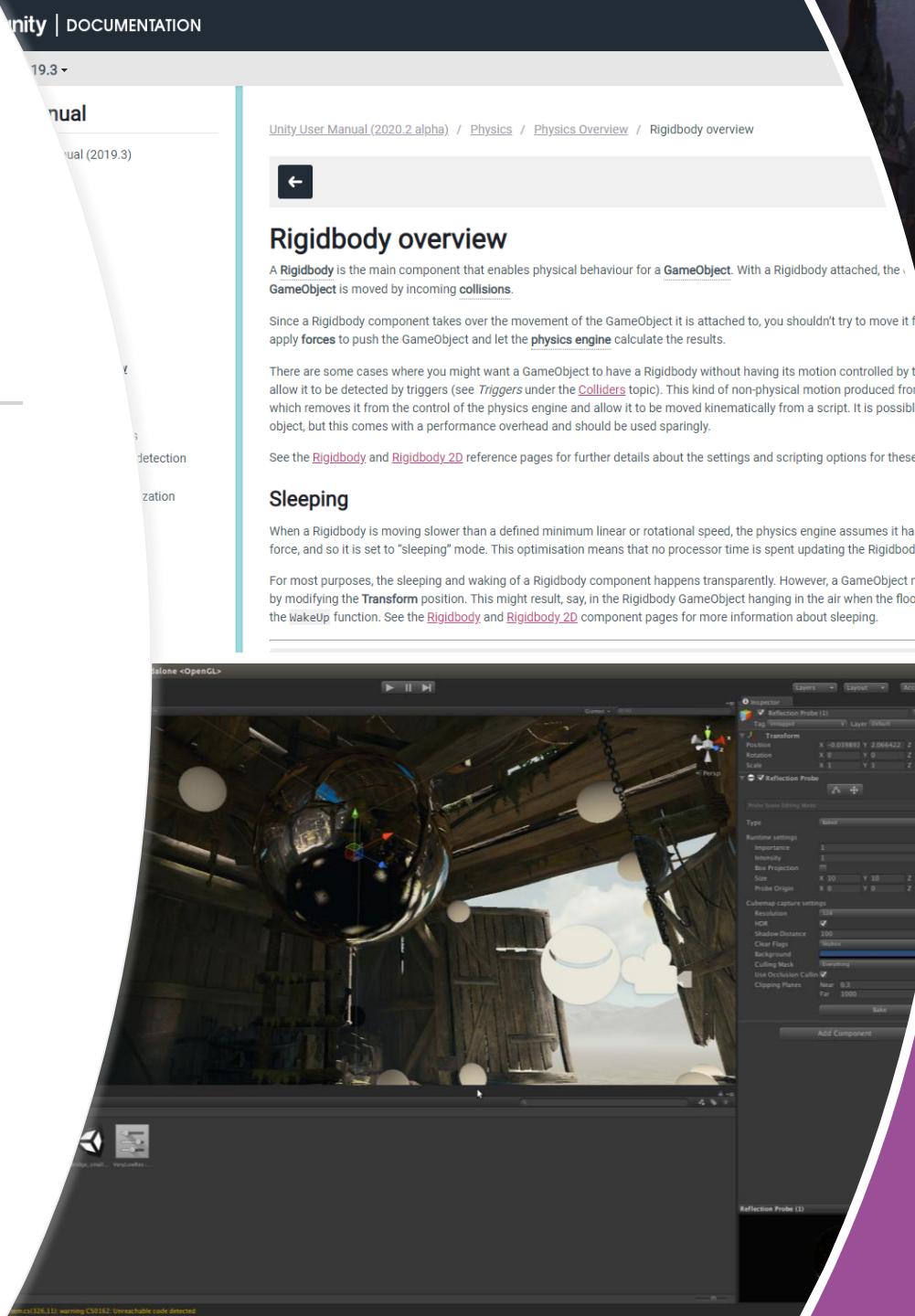
учить С++ должен  
ты

risovac

22

# Плюсы Unity

- + C#
- + Удобный редактор
- + Удобная документация
- + Большое комьюнити
- + Широкий выбор платформ
- + Много плагинов и ассетов
- + Отлично подходит для мобильных игр



The Unity Editor interface is shown, featuring a 3D scene with a large reflective sphere hanging from a chain. The Inspector panel on the right displays the properties of a 'Reflection Probe' component attached to the sphere. The properties include Position, Rotation, Scale, and various settings for Probe Scene Editing Mode and Culling Mask.

**Rigidbody overview**

A **Rigidbody** is the main component that enables physical behaviour for a **GameObject**. With a Rigidbody attached, the **GameObject** is moved by incoming [collisions](#).

Since a Rigidbody component takes over the movement of the GameObject it is attached to, you shouldn't try to move it if you apply **forces** to push the GameObject and let the **physics engine** calculate the results.

There are some cases where you might want a GameObject to have a Rigidbody without having its motion controlled by the physics engine. For example, if you want to allow it to be detected by triggers (see [Triggers](#) under the [Colliders](#) topic). This kind of non-physical motion produced from a Rigidbody is called **kinematic** motion. It removes it from the control of the physics engine and allows it to be moved kinematically from a script. It is possible to move a kinematic object, but this comes with a performance overhead and should be used sparingly.

See the [Rigidbody](#) and [Rigidbody\\_2D](#) reference pages for further details about the settings and scripting options for these components.

**Sleeping**

When a Rigidbody is moving slower than a defined minimum linear or rotational speed, the physics engine assumes it has come to rest and no longer needs to calculate its position. This optimisation means that no processor time is spent updating the Rigidbody until it moves again.

For most purposes, the sleeping and waking of a Rigidbody component happens transparently. However, a GameObject might find itself in a suspended state if it is modified by a script. This might result, say, in the Rigidbody GameObject hanging in the air when the floor has its `wakeUp` function. See the [Rigidbody](#) and [Rigidbody\\_2D](#) component pages for more information about sleeping.



## Минусы Unity

- Трудно настроить графику
  - HDRP требует обновления шейдеров
  - Редактировать UI бывает неудобно
  - ECS не из коробки
  - Повышать версию больно



# Известные игры на Unity



## Команда для инди разработки

- 2D/3D художник (100% нужен)
- Тестировщики (100% нужны)
- Геймдизайнер (по ситуации)
- UI/UX (по ситуации)
- Саунд дизайнер (по ситуации)



## На что обратить внимание

- Контроль работы команды, составление задач
- Новичков всегда необходимо обучить
- Мотивация команды



# Технологическая часть



# Демо 1. Интерфейс Unity. Сцена



# Демо 2. MonoBehaviour



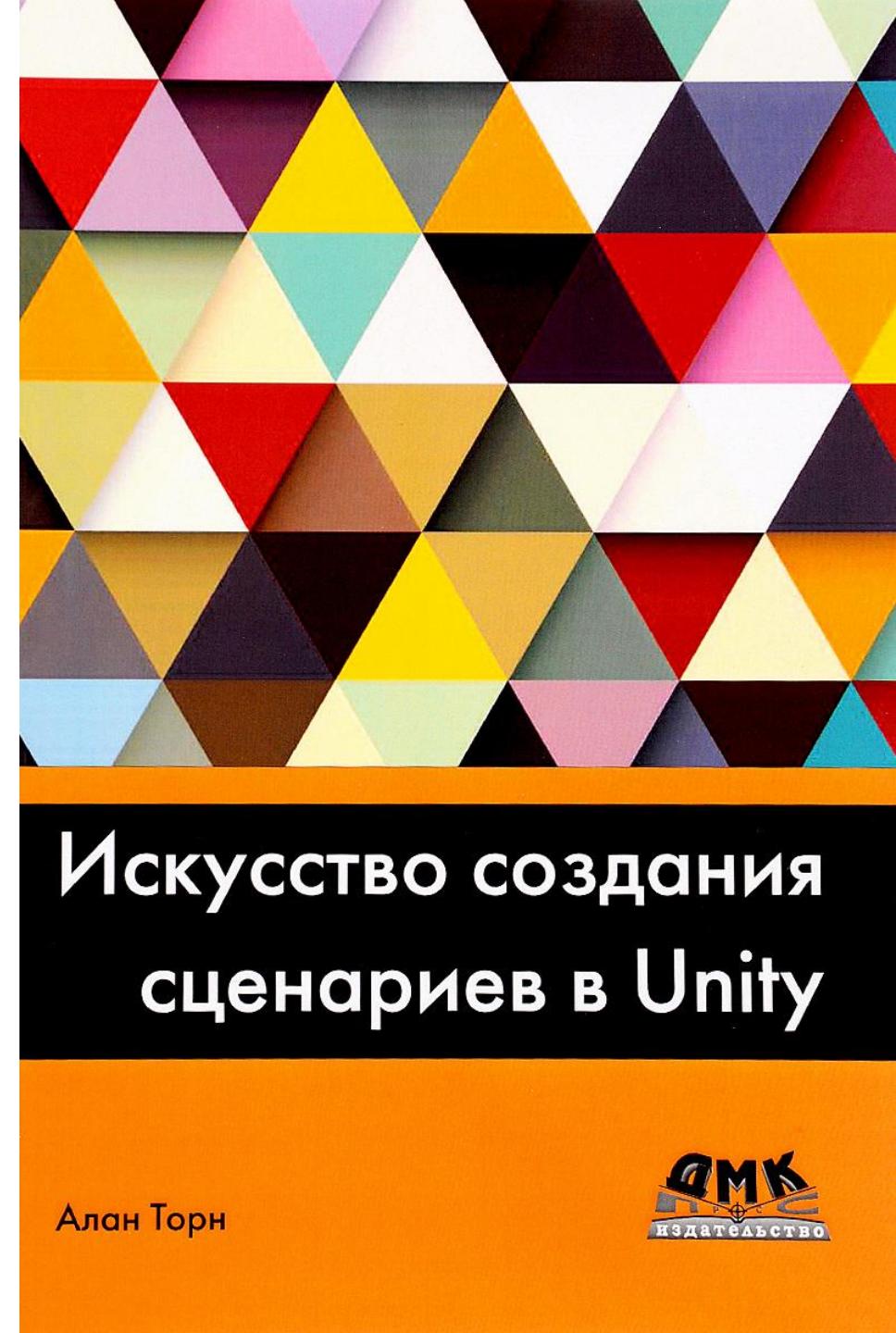
# Демо 3. ScriptableObject



## На что обратить внимание при разработке

---

- Архитектура игры
- EntryPoint
- Учитывайте возможность простой настройки баланса и игровых объектов



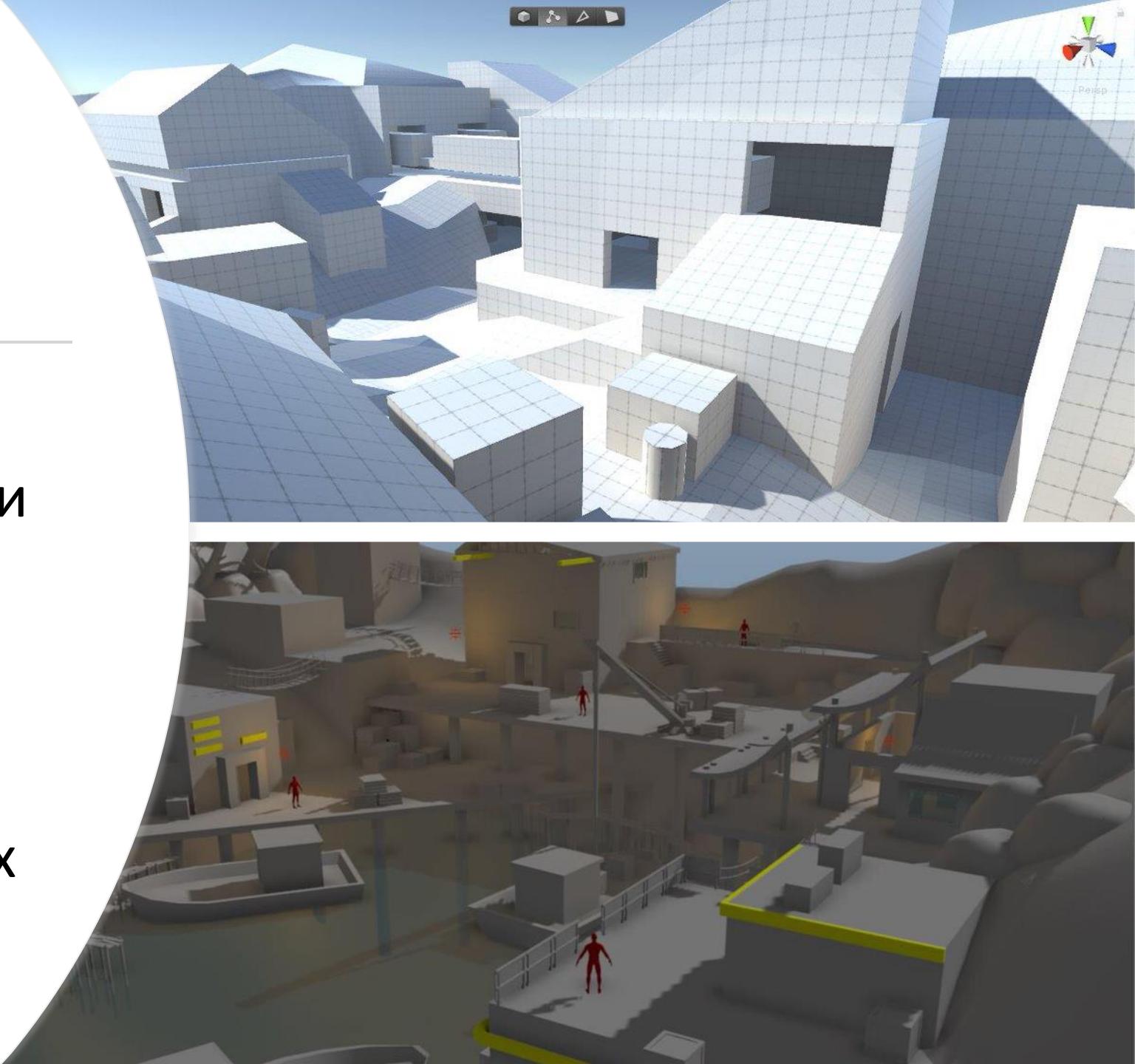
Алан Торн

# Дизайн уровней



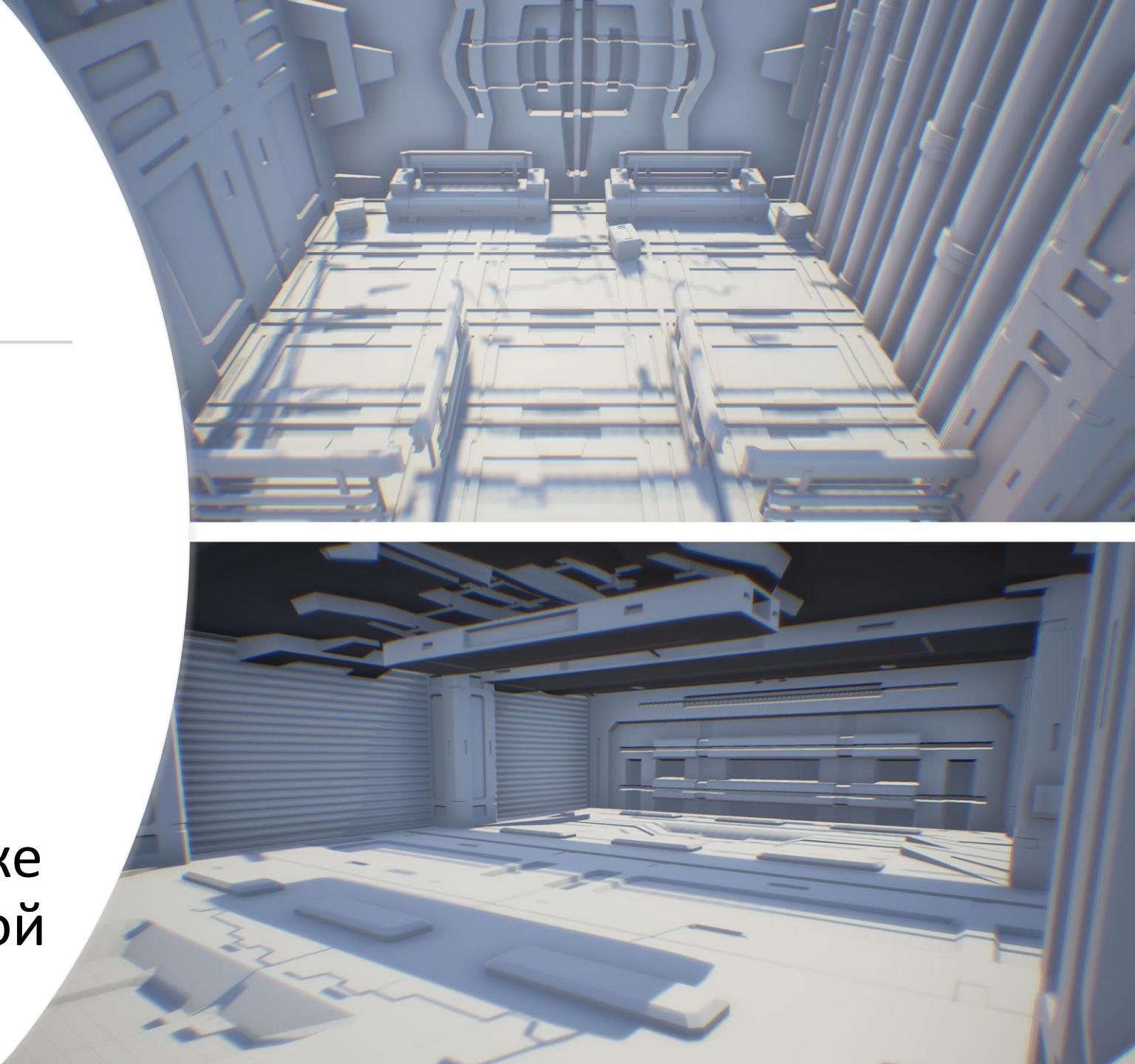
## GreyBox

- Уровень собирается из примитивной геометрии или упрощенных 3D моделей
- Задача: настройка метрик уровня, тестирование как сырых игровых механик так и самого уровня



## WhiteBox

- Усложняется геометрия прототипа, промежуточная стадия между Greybox и релизом.
- Задачи: тестирование метрик и механик на уже более детализированной локации



## Финальная стадия

- Графический пасс
- Завершающий пасс
- Полировка



Что дальше?



## Два пути

- Продвигать самому
- Положиться на издателя



# Мобильные платформы

---

- Play market <https://support.google.com/googleplay/android-developer/answer/6112435?hl=ru>
- App store <https://developer.apple.com/programs/how-it-works/>

# Если выпускаться на PC

---

- Steam <https://partner.steamgames.com/steamdirect>
- GOG  
[https://www.gog.com/news/say hello to gogcomindie](https://www.gog.com/news/say_hello_to_gogcomindie)
- Продвижение на разных сайтах <https://dtf.ru/indie/30700-5-luchshih-angloyazychnyh-ploshchadok-dlya-besplatnogo-prodvizheniya-indi-igry>

Конференции  
по геймдеву

Dev  
**GAMM!**

WHITE  
NIGHTS  
CONFERENCE

## Итог

- Начните с документации
- Команда подразумевает больше ответственности
- Unreal красивый, но сложный. Unity в этом плане проще, но не отстает
- Важно заранее подумать об архитектуре вашей игры
- Итерационно разрабатывайте уровни
- Продвижение стоит денег



# Источники и полезные ссылки

- Unity, ECS и все-все-все <https://habr.com/ru/post/358108/>
- Как написать диздок. Блог компании Mail.ru <https://habr.com/ru/company/mailru/blog/266369/>
- Как выложить игру в Steam [https://vk.com/@spidamoo\\_games-kak-vylozhit-igru-na-stim-poshagovaya-instrukciya](https://vk.com/@spidamoo_games-kak-vylozhit-igru-na-stim-poshagovaya-instrukciya)
- Алан Торн. Искусство создания сценариев в Unity
- Статья компании FoxTime с примерами диздоков <https://appfox.ru/blog/avtorskiy/delimsya-o-putem-5-primerov-napisaniya-dokumentatsii-po-razrabotke-igr/>
- Создание игровых уровней: советы и хитрости <https://habr.com/en/post/274625/>
- Как левел дизайнеры создают уровни <https://www.school-xyz.com/kak-level-dizajnery-sozdayut-urovni>
- Канал Brackeys <https://www.youtube.com/user/Brackeys>
- Бонд Д. Unity и C#. Геймдев от идеи до реализации

Спасибо за внимание!

