Муниципальное бюджетное общеобразовательное учреждение

"Средняя общеобразовательная школа №32

с углубленным изучением отдельных предметов "

**Информационно-практико-ориентированный**

**проект.**

**"Создание игры"**

Выполнил:Сергеев Владимир

ученик 10 Б класса В МБОУ СОШ №32

Руководители:Сабирова Светлана Васильевна

Учитель средних и старших классов МБОУ СОШ №32 высшей квалификационной категории

Ижевск 2021

Содержание

1.Введение................................................................................................................

2.Теоритическая часть..............................................................................................

3.Практическая часть...............................................................................................

4. Заключение ……………………………………………………………………..

5. Приложение...............................................................................................................

6.Интернет ресурсы.......................................................................

**1.Введение.**

В нашем современном мире бешенство людей не по наслышке слышали что, такое компьютер. У компьютера много применений это помощь в работе вычислительных задач , использование многочисленных программ , использование информации в интернете и так далее. Большинство людей используют компьютер с целью получения удовлетворения от компьютерных игр. На сегодняшние время эта тема является очень популярной и актуальной. Большая часть населения трактует компьютерную игру как плохое влияние на психику человека. Но игра может быть полезной для развития логического мышления, умение выстраивать свою стратегию прохождения уровня или игры в целом.

Мне всегда было интересно попробовать себя в этой области, хоть не много погрузиться в мир информационный технологий. Пройдя пол года обучения в Академии Калашников я попробовал создать простую игру, хоть у меня нет опыта и достаточной информации.

**Объектом:** Моего исследования будет инструкцией посозданию игры

**Предметом:** Моего исследования будет игра

**Гипотеза:** допустим, что мой проект покажет, что даже не обладая особыми знаниями каждый сможет начинать создавать свои игры, которые будут интересны, при этом они будут развивать логическое мышление.

**Цель:**Рассказать, как можно создать свою игру и показать, что игры могут нести положительное влияние на человека

**Задачи:**Рассказать на примере как можно создать простую игру

**2.Теоретическая часть**

**Компьютерная игра́** — [компьютерная программа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), служащая для организации [игрового](https://ru.wikipedia.org/wiki/%D0%98%D0%B3%D1%80%D0%B0) процесса ([геймплея](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%B9%D0%BC%D0%BF%D0%BB%D0%B5%D0%B9)), связи с партнёрами по игре, или сама выступающая в качестве [партнёра](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B0%D1%80%D1%80%D0%B8%D0%BD%D0%B3).

В настоящее время, в ряде случаев, вместо *компьютерная игра* может использоваться [*видеоигра*](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D0%B8%D0%B3%D1%80%D0%B0), то есть данные термины могут употребляться как синонимы и быть взаимозаменяемыми[[3]](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0#cite_note-3)[[4]](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0#cite_note-4)[[к. 1]](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0#cite_note-video-6). В компьютерных играх, как правило, игровая ситуация воспроизводится на экране дисплея или обычного телевизора (в этом случае компьютерные игры одновременно являются и [видеоиграми](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D0%B8%D0%B3%D1%80%D0%B0)), но в то же время компьютерная игра может быть [звуковой](https://ru.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0), [телетайповой](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%82%D0%B0%D0%B9%D0%BF) и др.

Создавать игру будем на базе Unity

**Unity** — [межплатформенная](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) [среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) [компьютерных игр](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0), разработанная американской компанией [Unity Technologies](https://ru.wikipedia.org/wiki/Unity_Technologies). Unity позволяет создавать приложения, работающие на более чем 25 различных [платформах](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0), включающих [персональные компьютеры](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80), [игровые консоли](https://ru.wikipedia.org/wiki/%D0%98%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BA%D0%BE%D0%BD%D1%81%D0%BE%D0%BB%D1%8C), [мобильные устройства](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B1%D0%B8%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0), [интернет-приложения](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) и другие[[2]](https://ru.wikipedia.org/wiki/Unity_(%D0%B8%D0%B3%D1%80%D0%BE%D0%B2%D0%BE%D0%B9_%D0%B4%D0%B2%D0%B8%D0%B6%D0%BE%D0%BA)#cite_note-_2220faf84e878ba7-2). Выпуск Unity состоялся в [2005](https://ru.wikipedia.org/wiki/2005_%D0%B3%D0%BE%D0%B4_%D0%B2_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D1%85_%D0%B8%D0%B3%D1%80%D0%B0%D1%85) году и с того времени идёт постоянное развитие.

**Microsoft Visual Studio** — линейка продуктов компании [Microsoft](https://ru.wikipedia.org/wiki/Microsoft), включающих [интегрированную среду разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) программного обеспечения и ряд других инструментальных инструментов. Данные продукты позволяют разрабатывать как [консольные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) [приложения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), так и игры и приложения с [графическим интерфейсом](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F), в том числе с поддержкой технологии [Windows Forms](https://ru.wikipedia.org/wiki/Windows_Forms), а также [веб-сайты](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82), [веб-приложения](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), [веб-службы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) как в [родном](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), так и в [управляемом](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) кодах для всех платформ, поддерживаемых [Windows](https://ru.wikipedia.org/wiki/Windows), [Windows Mobile](https://ru.wikipedia.org/wiki/Windows_Mobile), [Windows CE](https://ru.wikipedia.org/wiki/Windows_CE), [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework), [Xbox](https://ru.wikipedia.org/wiki/Xbox), [Windows Phone](https://ru.wikipedia.org/wiki/Windows_Phone) [.NET Compact Framework](https://ru.wikipedia.org/wiki/.NET_Compact_Framework) и [Silverligh](https://ru.wikipedia.org/wiki/Silverlight)

**1.Основы программирования**

**Алгоритм -** конечная последовательность шагов, приводящих от исходных данных к искомому результату

**Свойства алгоритма:**

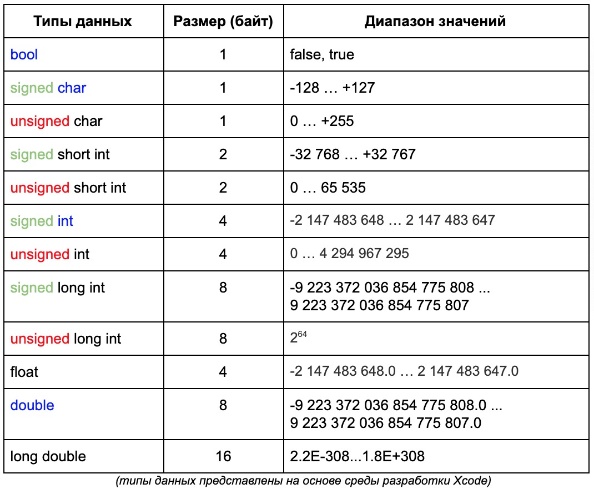
***Дискретность*** — алгоритм должен представлять процесс решения задачи как последовательное выполнение простых шагов.

***Определенность***— каждое правило алгоритма должно быть четким, однозначным.

***Результативность*** — алгоритм должен приводить к решению задачи за конечное число шагов.

***Константа*** *—* это величина, которая при выполнении программы остаётся неизменной.

***Переменная*** *—* это ячейка памяти для временного хранения данных. Предполагается, что в процессе выполнения программы значения переменных могут изменяться.



**2.Процесс компиляции**

**Исходный C++ файл** — это всего лишь код, но его невозможно запустить как программу или использовать как библиотеку. Поэтому каждый исходный файл требуется скомпилировать в исполняемый файл, динамическую или статическую библиотеки

**Этапы компиляции:**

***1.Препроцессор*:** подстановка заголовочных файлов и макросов.

***2.Лексический анализ*:** преобразование последовательностей символов в лексемы программы (выявление «орфографических» ошибок).

***3.Синтаксический анализ*:** структурирование последовательности лексем во внутреннее представление программы в виде дерева синтаксического разбора.

***4.Генерация кода*:** преобразование внутреннего представления в объектный (целевой) код.

**5.Ассемблирование:** преобразование автокода в машинный код.

**6.Компоновка:** сборка всех объектных модулей и модулей, содержащих системные вызовы данной ОС.

7.**Исполнение:** выполнение инструкций объектного языка.

**Ввод информации**

**Алфавит** -символы для записи конструкций языка программирования. Алфавит языка состоит из букв, цифр и лексем.

**Лексема** - это наименьшая единица языка имеющая самостоятельный смысл.

К лексемам относятся:

* + спецсимволы (==, !=, ++, =, >, < и т.д.);
  + ключевыеслова (main, switch, if, while ит.д.).

**Синтаксис**— набор правил, описывающий комбинации символов алфавита, считающиеся правильно структурированной программой (документом) или её фрагментом.

Если последовательность символов принадлежит языку, то она считается синтаксически правильной. Для программы это означает, что транслятор на ней не выдает ошибки. Но синтаксическая правильность не гарантирует даже осмысленности программы.

**Семантика**— это соответствие между синтаксически правильными программами и действиями абстрактного исполнителя, то есть это смысл синтаксических конструкций.

Семантика определят то, как именно программа будет восприниматься компилятором.

**Структура программы**— набор функциональных частей программы, выстроенных в соответствии с правилами используемого языка программирования

— искусственно выделенные программистом взаимодействующие части программы

**3.Структура программы С++**

**Компьютерная программа — это последовательность инструкций,** которые сообщают компьютеру, что ему нужно сделать.

Базовым строительным блоком программы являются инструкции (statement). Инструкция представляет некоторое действие, например, арифметическую операцию, вызов метода, объявление переменной и присвоение ей значения.

Компилятор также способен обрабатывать выражения. Выражение (англ. «expression») — это математический объект, который создается (составляется) для проведения вычислений и нахождения соответствующего результата

**Библиотека** — это набор скомпилированного кода (например, функций), который был «упакован» для повторного использования в других программах. С помощью библиотек можно расширить возможности программ. Например, если вы пишете игру, то вам придется подключать библиотеки звука или графики (если вы самостоятельно не хотите их создавать).

Язык C++ не такой уж и большой, как вы могли бы подумать. Тем не менее, он идет в комплекте со **Стандартной библиотекой С++**, которая предоставляет дополнительный функционал. Одной из наиболее часто используемых частей Стандартной библиотеки C++ является **библиотека iostream**, которая позволяет выводить данные на экран и обрабатывать пользовательский ввод.

В языке C++ стейтменты объединяются в блоки — **функции**. **Функция** — это последовательность стейтментов для выполнения определенного задания. Каждая программа, написанная на языке C++, должна содержать главную функцию ***main()***. Именно с первого стейтмента, находящегося в функции *main()*, и начинается выполнение всей программы.

Функции, как правило, выполняют конкретное задание. Например, функция *max()* может содержать стейтменты, которые определяют большее из заданных чисел, а функция *calculateGrade()* может вычислять среднюю оценку студента по какой-либо дисциплине.

**Вызов функции** — это выражение, которое указывает процессору прервать выполнение текущей функции и приступить к выполнению другой функции.

Для возвращения результата из функции или немедленного прерывания ее работы применяется оператор **return**.

**3.Практичекая чать**

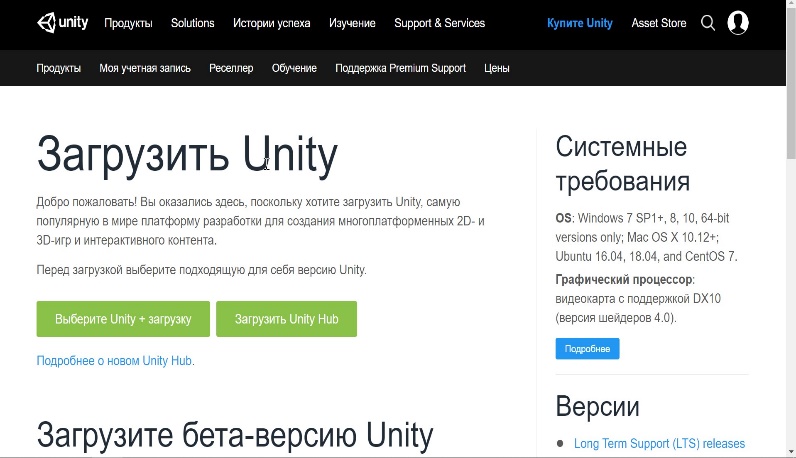
После обучения в Аккадемии я захотел попробовать создатьт игру сам при помощи интерент источников.

В интеренте я увидел одну интересную идею игры и захотел попробовать ее реализовать.

**Поэтому я расскажу все этапы создания этой и гры**

1. Если вы до этого не работали в unity или вообще не делали игры, то этот проект, будет самое то для начала. Суть игры вытолкнуть розовый блок за пределы коробки одновременно можно двигать только горизонтальные и вертикальные блоки.

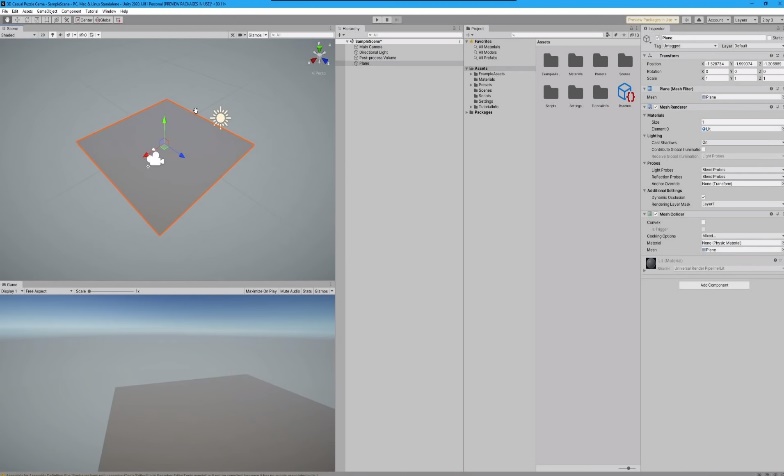
2 Все что нам потребуется для работы это, игровой движок unity, никакого рисования графики никаких других программ только сам движок его вы можете установить бесплатно с официального



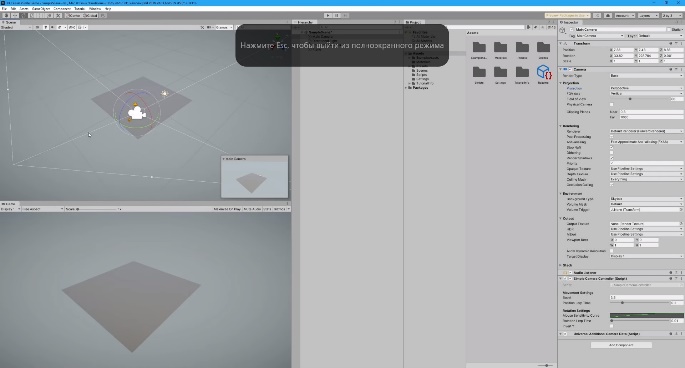
3 Перешли по ссылке на сайт unity нажимаете загрузить unity хаб, когда unity хаб скачался и вы егооткрыли переходите во вкладку instance, здесьвыбираете самую новую версию это важно. Далее нажимаете next и тут выбирать, если вам надо что-то поставить дополнительно, например, если будете делать игры на android выбирайте android галочку если на ios или другие системы, планируете делать соответственно тоже, галочкиставьте тут, как захотите теперь ждать пока версияскатается.

4.Переходим во вкладку проекты и нажимаем (new) сюдавбивайте название проекта и тип проекта, я выберу не просто 3d а you neversurrenderpipelineи нажимаем create, ждем запуска unity

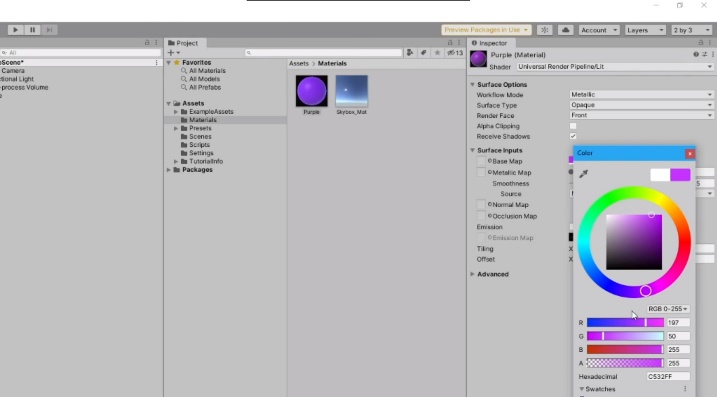
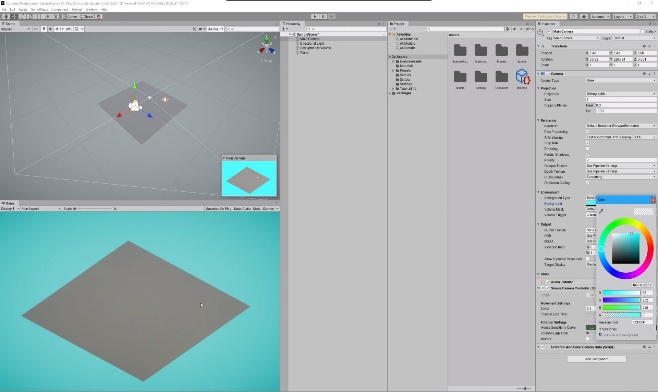
5.Когда движок открылся, мы можем увидеть перед собой непонятную картину, я нажму день, layout и поменяю расстановку на 2: 3, так мне удобнее работать кратко по интерфейсу: слева сверху эту сцену наше рабочее пространство, а снизу это то что мы будем видеть в самой игре здесь все объекты на нашей сцене, справа инспектор в нем свойства каждого объекта все что у нас сейчас есть на сцене это главная камера, источник света, так что здесь я нажимаю правой кнопкой мыши 3d объект и выбираю plainу нас устанавливается панель, справа в инспекторе надо нажать три точки и reset чтобы у нее сбросилась позиция снизу можем видеть как это панель сейчас выглядит в игре и от не совсем то что нам нужно.

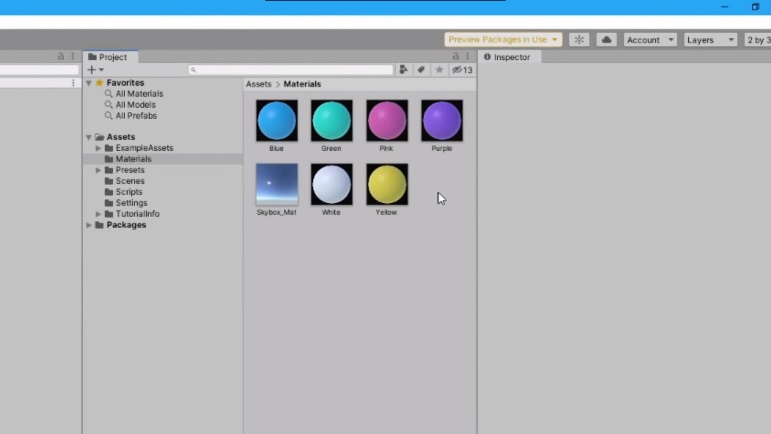


6.Я выбираю нашу камеру и за зеленую стрелочку тащу ее вверх, за синюю и красную также чуть отодвигаю в итоге нам нужно чтоб в игре панель была видна полностью наверху выбрав инструмент вращения мы можем развернуть нашу камеру и еще немного ее подвинуть так ,чтобы она смотрела четко на панель,режим камеры в инспекторе надо поменять перспектив график и теперь в игре, мы видим ее плоской как надо.

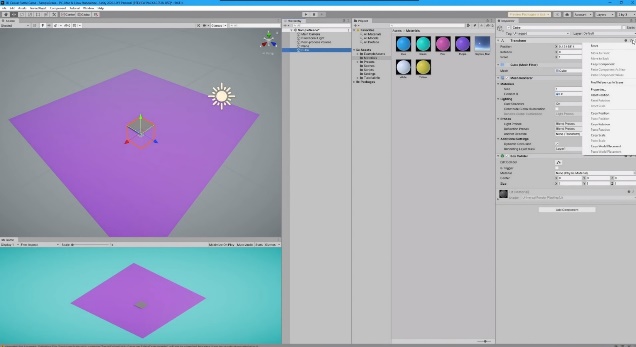


Также задний фон камеры меняем в какой вы хотите цвет, чтобы на заднем фоне был голубой, а чтоб сменить цвет самой панели и она не была такой серой и скучной надо здесь перейти в папочку с материалами и на правую кнопку мыши , новый материал, называйте его как хотите, я назвал purple потому, что в настройках материала хочу поставить фиолетовый цвет, материал теперь фиолетовый, его нам нужно просто взять и прямо отсюда перетащить на объект, и панель наши в игре теперь стала фиолетовой, таким же образом можете создать еще разных материалов других цветов цвета для материалов можно брать просто рандомно либо с каких-нибудь сайтов с палитрами.

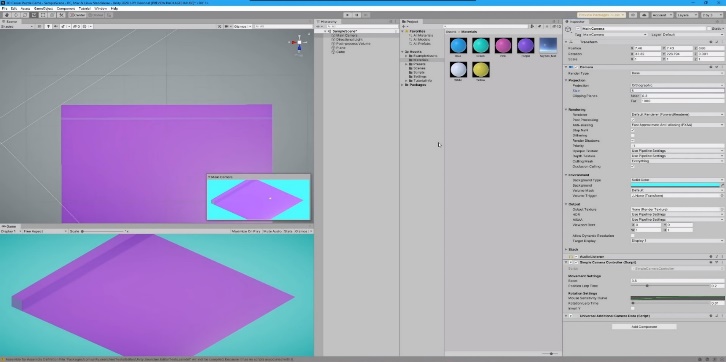




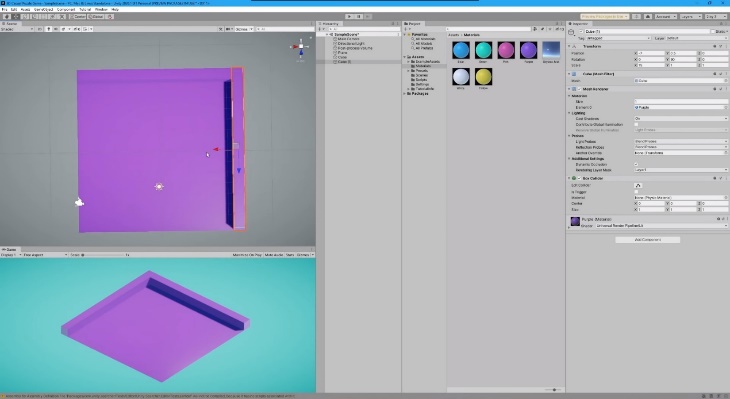
7.Далее я создан новый 3d объект, на этот раз просто куб, у него мы также должны сбросить позицию и немножко поднять чтоб он все таки не был застрявшим в панели ,совет( вы можете не только просто его поднимать, и опускать но и делать это точнее, зажав контуром когда, я пытаюсь его поднять зажатым контролом, он поднимается обрывисто только на полклетки так что потянув вверх два раза сзажатым контролом я его подниму безвсякой погрешности) на сам куб , я также передвину фиолетовый материал чтобы он был такого же цвета, и панель все-таки сделал немного по шире, поставить в 1.5 , теперь она достаточнобольшая.



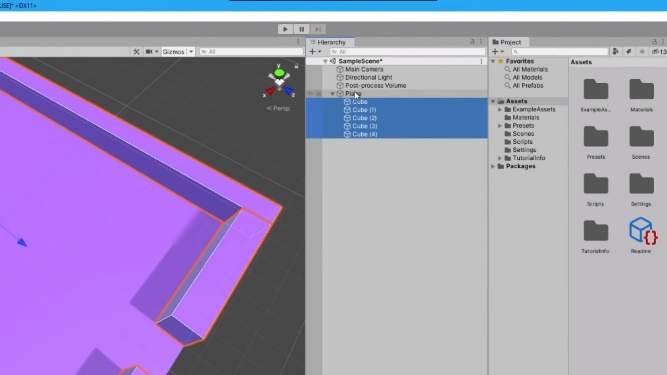
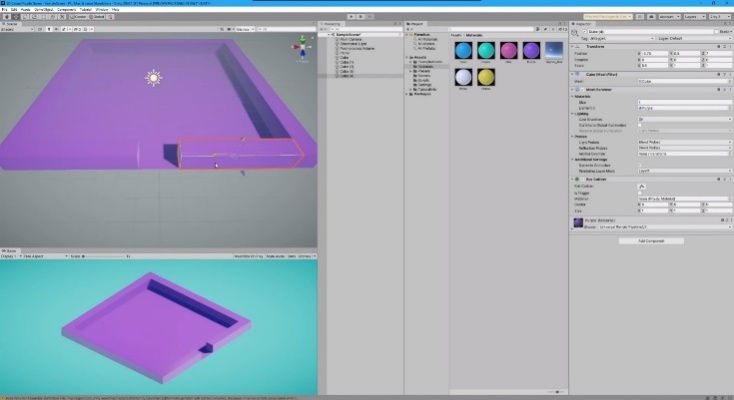
Этот куб и я также с зажатым контролем перетягиваю к верхней границе выбираем инструмент растягивание, также зажатым контролем его растягиваю, в итоге он должен быть растянут ровно на стенку, можете менять размер камеры далее этот куб копирую, нажимаю контролемb разворачиваю его на 90 градусов и устанавливая сбоку.



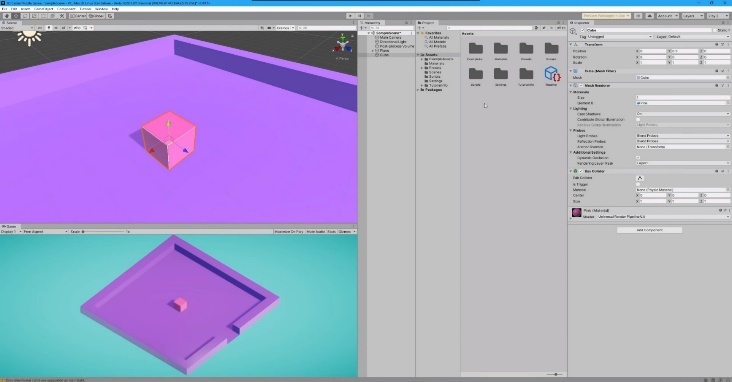
нажимаем сontrol и еще раз и копирую стену на другую сторону.



теперь выберу самую первую стену еще раз нажму control-d перетяну вниз на длину и сделаю его меньше так что бы с двух сторон его поставить и осталось место.



Созданием теперь новый 3d кубтак же сбросим позицию и поднимем ему дано этот раз розовый цвет это и будет наш главный куб, который должен выбраться отсюда и чтоб он двигался пришло времянаписать ему контроллер нажимаю создать скрипт с# скрипт назову Player.



8) Переходим в редактор кода visualstudio, мы хотим, чтоб игрок у нас двигался в двух направлениях, вверх и вниз ну либо влево, вправо если горизонтально, тонам нужно задать две клавиши на клавиатуре при нажатии которых он будет двигаться. Пишем вот этот скрипт.

**usingSystem.Collections;//** Добавляем необходимые библиотеки

**usingSystem.Collections.Generic; //** Добавляем необходимые библиотеки

**using UnityEngine; //** Добавляемнеобходимыебиблиотеки

**public class Player :MonoBehaviour**

**{**

**[SerializeField] KeyCodekeyOne; //**добавляемкнопку 1

**[SerializeField] KeyCodekeyTwo;//** добавляемкнопку 2

**[SerializeField] Vector2 moveDirection;//** добавляем вектора

И далее описываем движение по нажатию кнопки

**private void FixedUpdate()**

**{**

**if (Input.GetKey(keyOne))**

**{**

**GetComponent<Rigidbody>().velocity += moveDirection; //скоростьвверх**

**}**

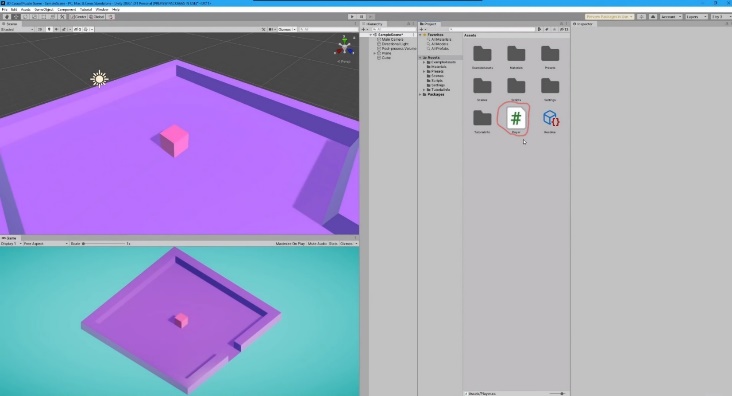
**if (Input.GetKey(keyTwo))**

**{**

**GetComponent<Rigidbody>().velocity -= moveDirection; //скоростьвниз**

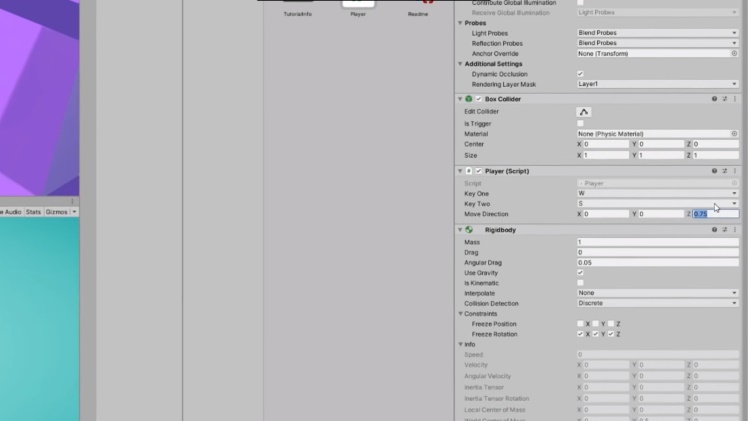
**}**

9)Этот скрипт пока что можем сохранить (распространенная ошибка если у вас скрипт не будет работать проверьте что его обычное название совпадать с названием класс, то есть и здесь playerи здесь player потому что если вы что-то переименовывали может выдать ошибку).

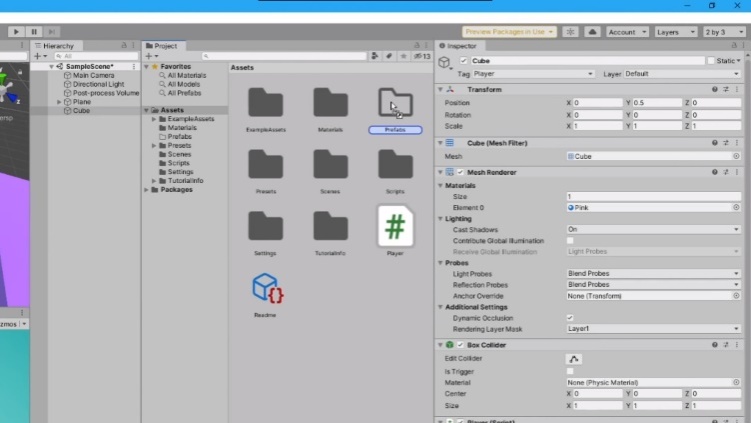


Что ж выбираем наш куб и перетягиваем на него скрипт и тут если вы все правильно сделали мы сами можем задать две клавиши ,это может быть любая абсолютно клавиши. Также добавим новый компонент rigidbody тут мы откроем constraints и поставим три галочки чтобы, куб у нас не вращался как попало также перед тем как будем тестить надо убрать у главной камеры вот этот компонент single camera controller удалить его.

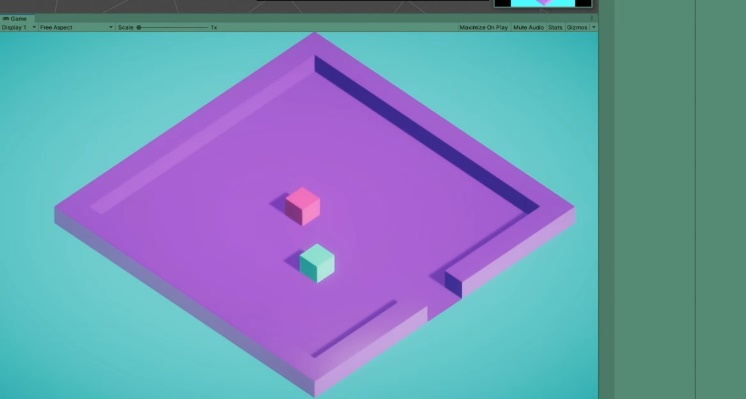
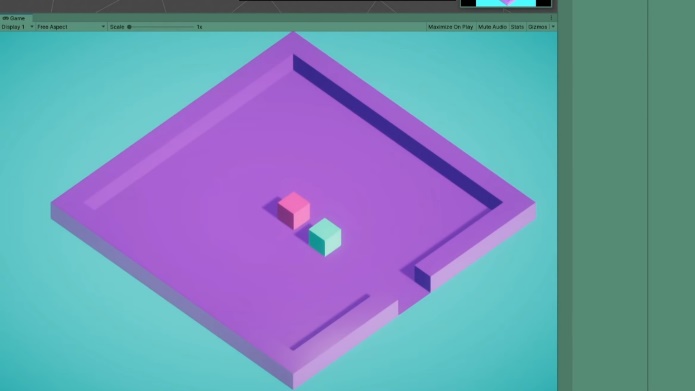
Вернемся опять кубу тут надо задать ему еще направлении вот пускаем будет двигаться по оси z со скоростью 0 75 до тут чем больше вы кажете значения тем больше будет его скорость и теперь нажимаем кнопочку play куб у нас отлично движется при нажатии на соответствующей клавиши который мы задали.



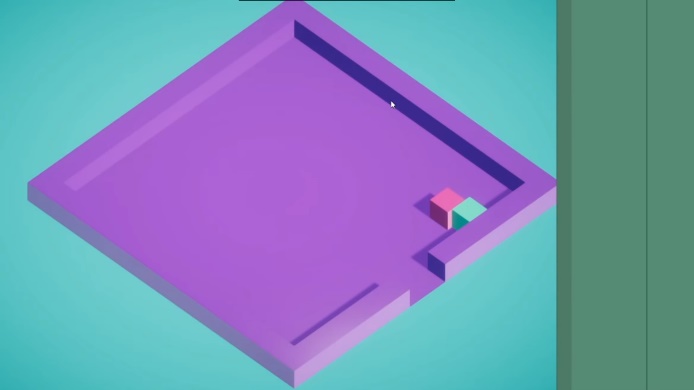
10)Создадим тут новую папку Prefabs и наш куб перетянем в неё теперь наш куб сохранён как Prefab, то есть мы можем его вставить в любое место в любое время и не надо все переделывать заново, впрочем, устанавливаем второй куб, и чтобы он двигался уже вправо-влево, меняем ему клавиши на 0,75 будет у него не по z у, а по x, теперь вернемся и можем выбирать цвет куба.



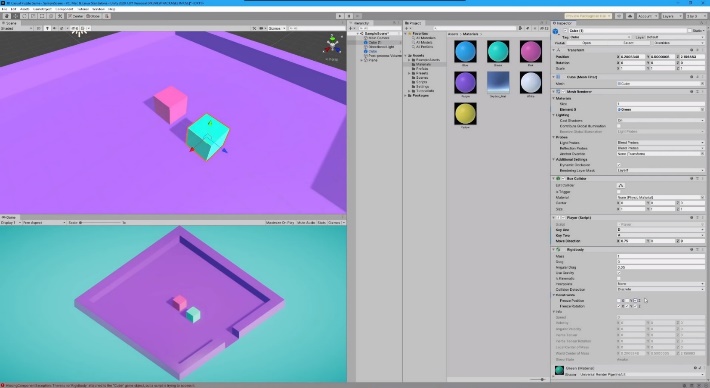
Теперь один куб двигается в одном направлении нажимая две кнопки и нажимая другие 2 кнопки второй куб двигается в противоположном направлении.( приложение 16-18)



10.Но есть одна проблема если, эти кубы случайно столкнуться то начнут, ехать вместе непонятно как, чтоб такого не было,и они держались строго на своих осях в ridgid body у розового куба замораживаем ось x, а у зеленого куба замораживаю ось z. Зеленый куб тоже перетаскиваем в отдельный Prefab.



11. Теперь можем еще ставить несколько кубов менять их цвет на синий, и он будет двигаться в одну сторону с розовым.

****

12.Теперь надо сделать так что мы смогли победить и пройти на следящий уровень. Берем скопированный блок и ставим его за предел нашего игрового поля, и при столкновении нашего основного куба с этим кубом, мы должны переходить на следящий уровень. выключаем (меж render) чтобы она была невидимой и галочку istrigger тоже нужно поставить для этой невидимой стены. Надо добавить текст финиш для нашего невидимого куба.

13.Теперь нам нужно написать код для финиша (но сколько я не пытался его реализовать программа не давала мне его использовать)

1f (this.CompareTag ("Player") && other.CompareTag ("Finish"))

SceneManager.LoadScene (SceneManager.GetActiveScene (). BuildIndex + 1);

} if (Input.GetKey (KeyCode.R))

{SceneManager. LoadScene (SceneManager.GetActiveScene (). BuildIndex);

{

if (Input.GetKey (KeyCode.Q))  
{

{SceneManager.LoadScene (SceneManager.GetActiveScene (). buildIndex + 1);

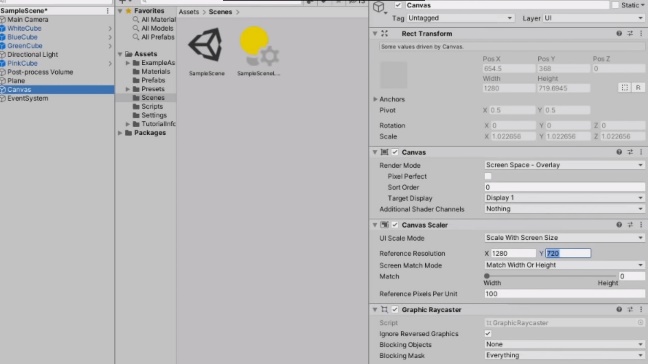
}

После того как мы добавим этот код в наш основной у нас должны работать переход между сценами, но для начала нам нужно создать сами сцены(уровни).

Для этого нужно скопировать нашу сцену и добавить новую. Теперь при прохождении уровня игра переходит на следящий уровень. На втором уровне изменяем расположение кубов что бы уровни отличались друг от друга. Так же для понятности можно добавить текст и написать на нём порядок уровня и обозначения букв R-играть заново, Q-перейти на следующий уровень.

12)Так же если будем менять разрешение экрана, то интерфейс может слетать чтобы такого не было надо у объекта Canvas поставь ScaleWithScreenSize и поставить разрешение экрана на ваше к примеру 1280\*720. Теперь при смене масштаба экрана, игра не будет искажаться.

Теперь остается только создавать новые уровни и усложнять их.

****

**4.Руководство по пользованию**

1.Запустить программу Unity,

2. нажимаем кнопку Play, находится с верху на панели управления.

3.Есть 4 действующие кнопки — это движение вперед по нажатию W( для главного куба )

Движение назад по нажатию кнопки S.(для главного куба )

Движение вправо по нажатию кнопки D.

Движение влево по нажатию кнопки A.

Цель игры:

Ход игры:

Так же в дальнейшем будут созданы кнопки (начать уровень заново) и (кнопка перехода на следующий уровень).

**5.Заключение**

**Итог:** Моя игра пока ещё требует завершения ведь создание игры требует терпения и необходимых знаний в написании кода, но пробуя создать эту игру, в первую очередь я хотел проверить свои умения в этой сфере и применения полученных знаний, так же на своем примере я хочу показать, что по моей инструкции игру может создать каждый. В дальнейшем я планирую доработать игру и добавить несколько уровней. Эта игра сможет показать, что игры могут быть и полезными для человека, развивать логическое мышление.

**6.Приложение**

Приложение 1

Разминка перед кодом

Обсуждение

Что такое голубь?

Что такое константа Какие типы данных вы знаете?

Что такое объект?

Что такое метод?

Давайте вспомним на практике ШАГ 1. Откройте СodeBlocks и создайте проект.

Не забудьте проверить, установлен ли компилятор! Для этого откройте Менеджер установки MinGW и проверьте наличие компилятора.

Задачи Запрограммируйте следующее выражение: (a + b - f // a) + f \* a \* a - (a + b) Числа a, b, f вводятся с клавиатуры. Напишите программу, которая требует ввода двух чисел. Если первое число больше второго, программа печатает число больше. Если первое число меньше второго, программа печатает Число меньше. И если числа равны, программа выведет сообщение Эти числа равны. По введенному номеру дня недели отобразите его название с помощью оператора множественного выбора. Нумерация дней недели начинается с 1 и т. Д. Понедельник, 2 - Вторник.

Задачи 1. Удалить все числа в массиве, повторяющиеся более двух раз. 2. Введите одномерный целочисленный массив. Найдите самый большой нечетный элемент. Далее трижды выполнить циклический сдвиг влево элементов вправо от найденного максимума и один раз - сдвиг элементов вправо, влево от найденного максимума. 3. Найдите сумму отрицательных элементов массива. 4. Найдите произведение элементов массива с нечетными номерами.

Задачи 1. Найдите сумму элементов массива между первыми двумя нулями. Если в массиве нет двух нулей, то вывести ноль. 2. Найдите самый большой элемент в массиве. 3. Найдите наименьший четный элемент в массиве. Если его нет, выведите первый элемент. 4. Преобразуйте массив так, чтобы сначала шли нулевые элементы, а затем все остальные. 5. Найдите сумму чисел минимального и максимального элементов. 6. Найдите элемент массива по модулю.

Приложение 2

****

**7.Интернет ресурсы**

[**https://www.youtube.com**](https://www.youtube.com)

[**http://edu.kalashnikov.academy/course/view.php?id=44**](http://edu.kalashnikov.academy/course/view.php?id=44)

[**https://docs.unity3d.com/ru/530/Manual/UnityManual.html**](https://docs.unity3d.com/ru/530/Manual/UnityManual.html)

[**https://unity3d.com/ru/get-unity/download**](https://unity3d.com/ru/get-unity/download)

[**https://ru.wikipedia.org/wiki/**](https://ru.wikipedia.org/wiki/) **и тд.**