ГУАП

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Д. А. Булгаков |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ПОДГОТОВКА МАРКЕРНАЯ ДОПОЛНЕННАЯ РЕАЛЬНОСТЬ НА БАЗЕ ARCORE |
| по курсу: Разработка виртуальной и дополненной реальности |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4941 |  |  |  | Н.С. Горбунов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

**Цель работы:** знакомство с пакетом разработки Google ARCore для дополненной реальности. Подготовка проекта Unity для работы с ARCore, создание простого приложения с использованием технологии маркерной дополненной реальности и реализация простейшего взаимодействия с объектами сцены через экран смартфона.

Вариант:

|  |  |
| --- | --- |
| **Вариант** | 7 |
| **Примитив** |  |
| Цилиндр | + |
| **Действие** |  |
| Изменение цвета | + |

**Подготовка к работе:**

Установка пакета дополнений ARCore. Скачав свежую версию пакета необходимо установить его в проект Unity через Package Manager.

В пакет входят такие плагины как:

* XR Plugin Management
* XR Legacy Input Helpers
* AR Foundation
* ARCore XR Plugin

Настройка проекта. В Build settings необходимо переключить платформу с ПК на Android. (рис. 1)

В Player Settings необходимо выставить настройки связанные с работой приложения. К примеру, выставить минимальную версию Android 7.0 Nougat, включить поддержку ARM64. (рис. 2)

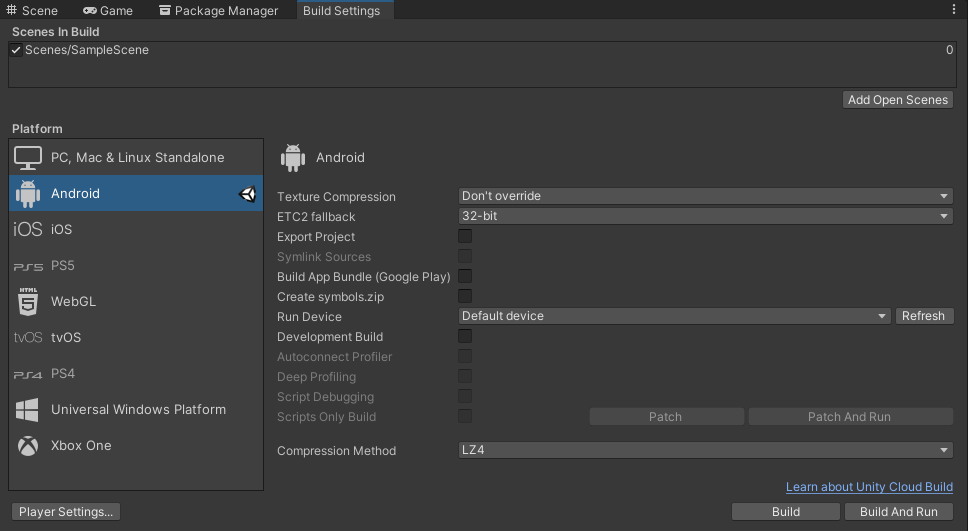


Рисунок 1. Build Settings

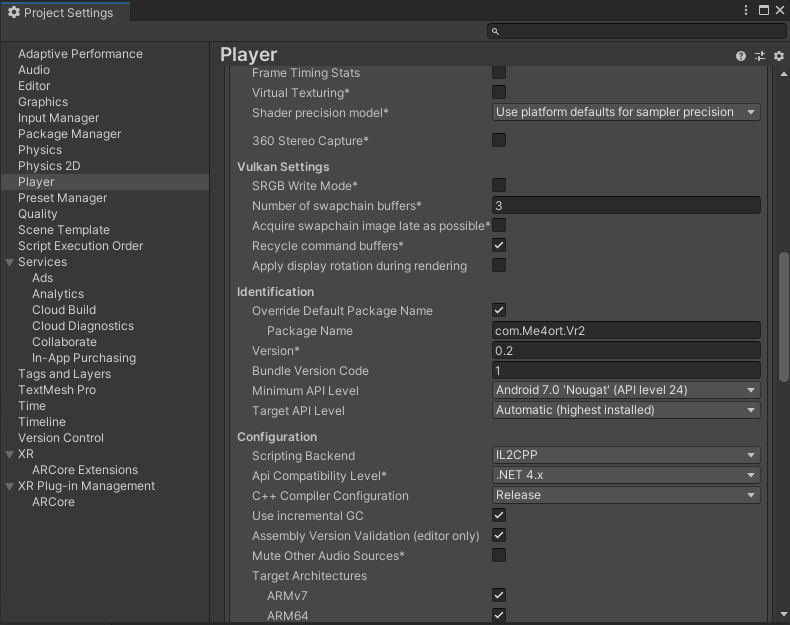


Рисунок 2. Player settings

**Выполнение работы**

Первым шагом к результату будет добавление специализированных объектов *AR Session Origin*  и *AR Session*. *AR Session Origin* устанавливает начало координат в дополненном окружении, а *AR Session* инициализирует сессию дополненной реальности. (рис. 3)

Для удобства работы нужно создать папки в директории Assets, для хранения элементов в процессе работы. (рис. 4)

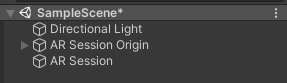


Рисунок 3. Окно иерархии с созданными объектами



Рисунок 4. Структура размещения файлов

Следующим шагом будет создание префаба, который будет появляться на метке. По заданию варианта — это примитив цилиндр. К нему необходимо применить материал primitive\_mat (рис. 5). Выставить размеры, приближенные к реальным (к примеру, выставить size по всем осям 0.2) и сохранить объект как префаб (рис. 6)

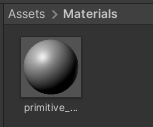


Рисунок 5. Материал



Рисунок 6. Созданный префаб

Этап создания маркера. Маркер будет распознаваться камерой, на нём программа будет проецировать выбранный 3D объект.

В маркер может быть преобразована любая картинка. Удовлетворяющая простым условиям: разрешение не менее 300x300 пикселей; отсутствие осевой симметрии; имеется достаточное количество неповторяющихся деталей. Важно отметить, что не стоит использовать картинки, в которых сложно однозначно определить их положение и ориентацию в реальном мире.

Для использования маркера необходимо воспользоваться компонентом *AR Tracked Image Manager*, который добавляется к AR Session Origin. В папке Markers надо создать *Reference Image Library*, что и будет являться маркером. В него помещается выбранная картинка. К примеру рис. 7



Рисунок 7. Изображение для маркера

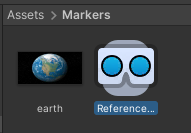


Рисунок 8. Маркер и изображение для него

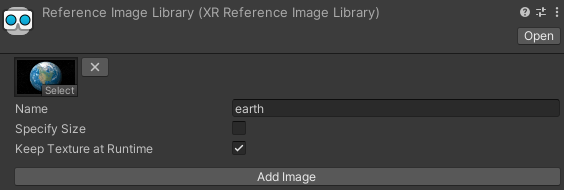


Рисунок 9. Настройки маркера

Поле «Keep Textures at Runtime» позволяет приложению запоминать маркеры во время выполнения, не выгружая их из памяти телефона.

По заданию при касании экрана на объекте должен изменится цвет этого объекта. Для этого необходимо добавить скрипт (листинг 1) на *AR Session Origin*, привязав к этому скрипту префаб объекта и *AR Camera*. Перед этим надо изменить префаб, добавив к нему Rigidbody комонент и тэг InteractableObject. (рис. 10)

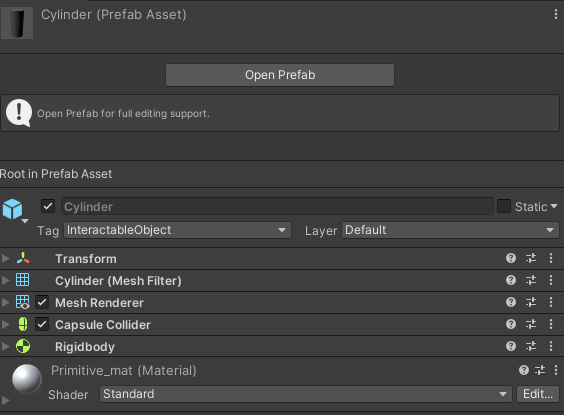


Рисунок 10. Новый префаб

Листинг 1. Скрипт обработки касания на объекте

public class ObjectRaycast : MonoBehaviour

{

//--------------ПЕРЕМЕННЫЕ----------------------------

// Объект, который будет реагировать на рейкаст

public GameObject activeObject;

// Определяем камеру, из которой будут исходить лучи

public Camera cameraAR;

bool colorFlag = true;

//----------------------------------------------------

void Update()

{

// Поиск объекта в сцене по тэгу

activeObject = GameObject.FindGameObjectWithTag("InteractableObject");

// Получаем компонент Рендерер активного объекта (куба)

var CulinderRenderer = activeObject.GetComponent<Renderer>();

// Если пользователь хотя бы раз нажал на экран

if (Input.touchCount > 0)

{

// Записываем в переменную информацию о касании экрана

Touch touch = Input.GetTouch(0);

// Смотрим фазу касания – она должна быть начальной

if (touch.phase == TouchPhase.Began)

{

// Если касание есть, из камеры посылается луч в место касания

Ray ray = cameraAR.ScreenPointToRay(touch.position);

// Сюда записываются объекты, в которые попал луч

RaycastHit hitObject;

// Если луч куда-то попал, можно выполнить действие над этим объектом

if (Physics.Raycast(ray, out hitObject))

{

// Если луч попал в куб

if (hitObject.transform.gameObject == activeObject)

{

if (colorFlag)

{

// Меняем цвет куба на красный

CulinderRenderer.material.SetColor("\_Color", Color.red);

colorFlag = !colorFlag;

}

else

{

// Меняем цвет куба на белый

CulinderRenderer.material.SetColor("\_Color", Color.white);

colorFlag = !colorFlag;

}

}

}

}

}

}

После всего этого можно собирать приложение. В итоге получается \*.apk файл, который является пакетным для системы Android.

Билд запускался на Samsung A30:

* Процессор Exynos 7904 8 ядер 1,8 ГГц
* ОС Android 11

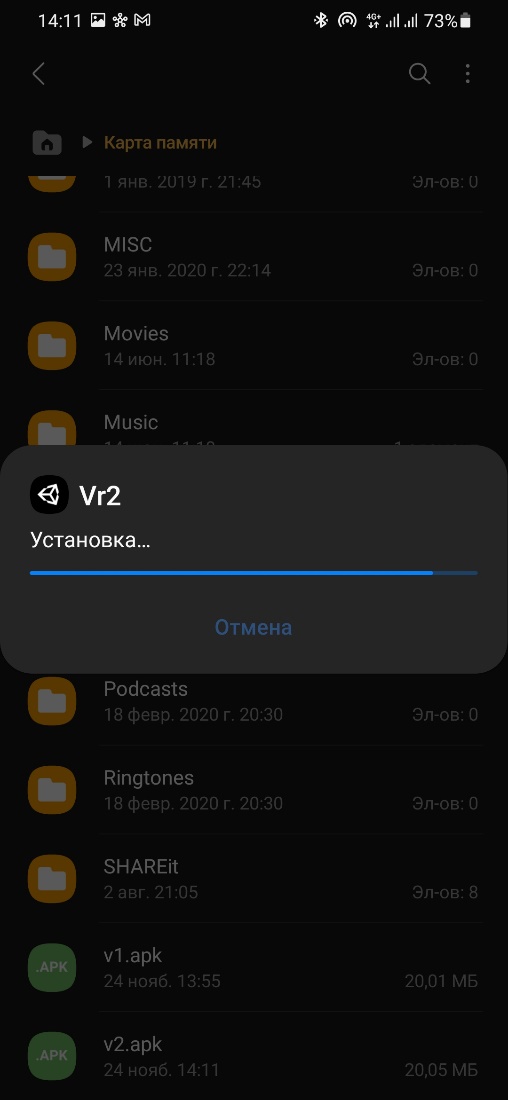


Рисунок 11. Установка приложения



Рисунок 12. Отображение объекта на метке

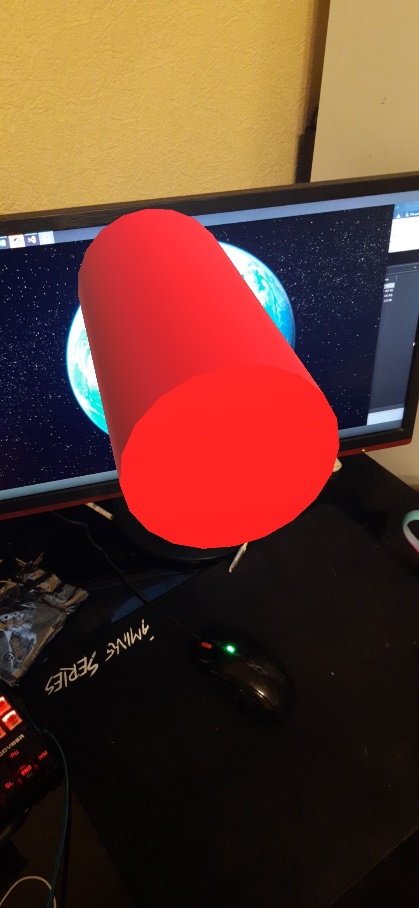


Рисунок 13. Изменение цвета после касания

**Вывод:** в данной работе рассмотрели простейшие инструменты для AR разработки, доступные в пакете *ARCore Extensions*. Воспользовались модулем, распознающим маркеры в реальном мире. Рассмотрели обработку нажатий на экране пользователем. В результате выполнения работы получили приложение, запускаемое на платформе Android, которое реагирует на метку в реальном мире и создаёт виртуальный объект-цилиндр. При нажатии на цилиндр он меняет цвет на красный.