

Лабораторная работа № 2. Проект – разработка простого AR-Приложения для Android-устройства (смартфон, планшет и пр.). Создание в графическом редакторе Unity 3D сцены дополненной реальности: **визуализация 2D-Изображения**.

Объекты ДР – это объекты проекта, создаваемого с помощью средств платформы Vuforia.

Введение.

Работа по созданию приложений ДР заключается в заведении проекта и объектов проекта (Контент) в **Vuforia**, а разработка 3D-сцен для объектов этого проекта осуществляется в **Unity 3D**. При этом **Vuforia** отвечает за идентификацию проекта через **License key** (см. ниже), а привязка к будущей сцене виртуального 2D- или 3D-объекта (например, 3D-модели, плоских изображений, видеоклипов и пр.) будет осуществляться через определяемую в **Vuforia** метку (**Target**). Допустимые в используемой в Лабораторном практикуме версии **Vuforia Engine** типы таргетов были подробно рассмотрены в Описании ЛР№2, Часть 1.

ВАЖНО!! → вся работа с **Vuforia** (с проектом, объектами) осуществляется через **web-интерфейс**, иными словами, **Vuforia** является облачным приложением. А работа с **Unity-3D** осуществляется непосредственно на компьютере разработчика, т.е. локально.

Связь между облачным ведением проекта (в **Vuforia**) и локальной проработкой сцен Приложения ДР должна быть выполнена за счет импорта подготовленных объектов проекта из облака **Vuforia** в среду редактора **Unity-3D**.

Рассмотрим типовую процедуру создания простого (игрового) **AR Приложения**.

Предлагается разработать приложение ДР для Android-устройств, в котором при наведении камеры устройства на реальную метку (таргет – изображение, например, на бумаге, или на дисплее) пользователь в области воспроизведения на экране мобильного устройства (МУ) увидит другое, заранее подготовленное **2D – изображение**.

Предварительные условия для начала работы:

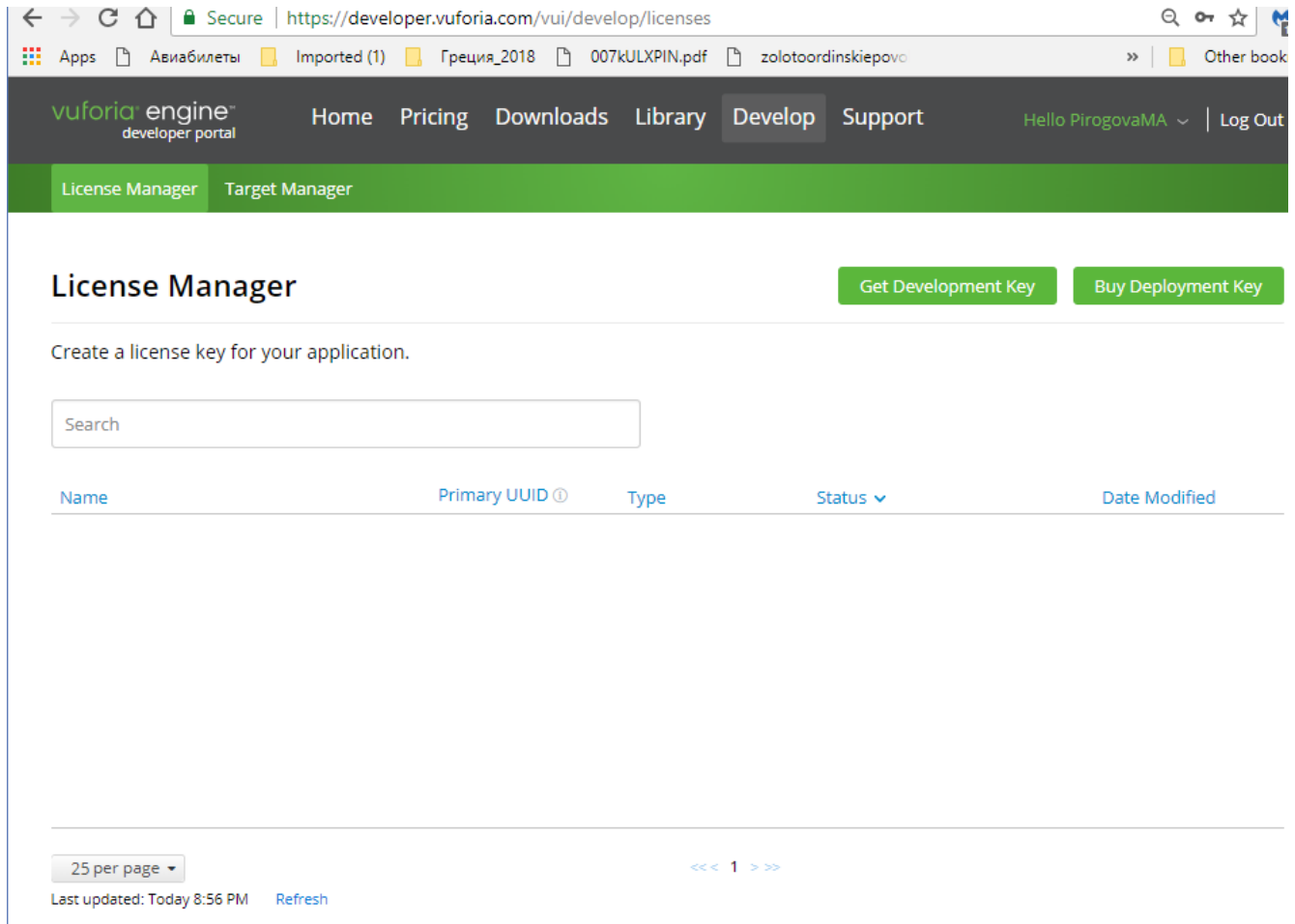
- Интернет-соединение локального компьютера;
- Наличие аккаунта пользователя **Vuforia** (результат успешного выполнения ЛР №1);
- Установленная на компьютере разработчика система **Unity 3D** (результат успешного выполнения ЛР №1);
- Заранее подготовленные изображения для метки (таргета) и контент (**2D-Изображение**);
- Кроме того, для формирования собственно приложения ДР на машине разработчика (**MSWindows 10/8.1/7**) необходимо установить **Android SDK** – набор инструментальных средств для сборки **Android-Приложений** в чужой операционной системе. Для корректной установки **Android SDK** надо предварительно установить (**если это еще не сделано**) **Java Development Kit – JDK**, предпочтительно последней версии; адреса для скачивания:

Android SDK – <https://developer.android.com/about/versions/10/setup-sdk>

JDK - <https://www.java.com/ru/download/manual.jsp>

*Дополнительные пояснения по особенностям установки **Android SDK** под **Windows** – см. **Приложение 1**: http://a0601.narod.ru/ADD01_LR_02.pdf

На сайте <https://developer.vuforia.com/> осуществляем вход с заведенными ранее логином/паролем. В результате получаем доступ к среде разработки **Vuforia** (облачное решение). Для ведения проекта средствами **Vuforia** необходимо выполнить две процедуры: получить **лицензионный код** на проект и сгенерировать метки (**Target**).



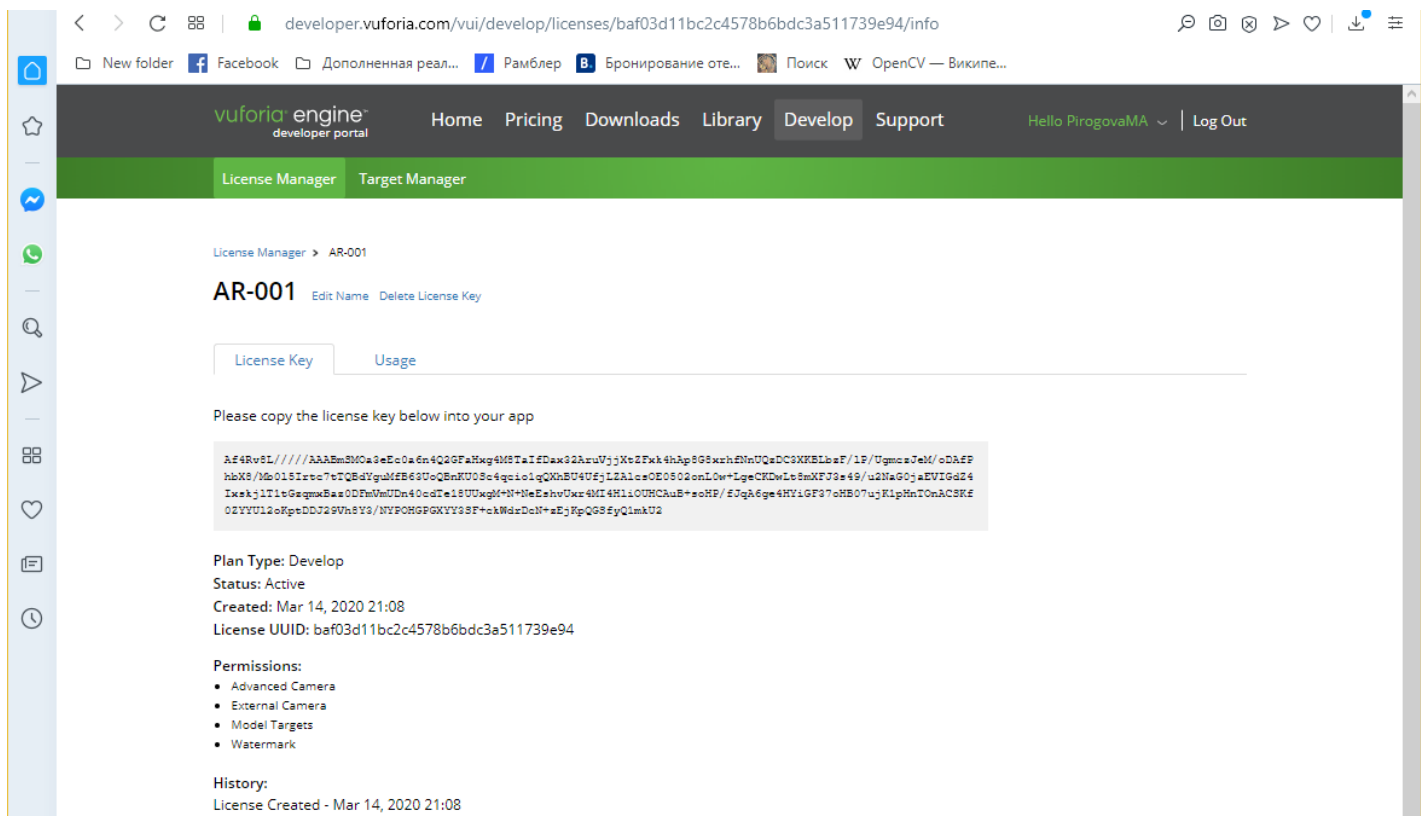
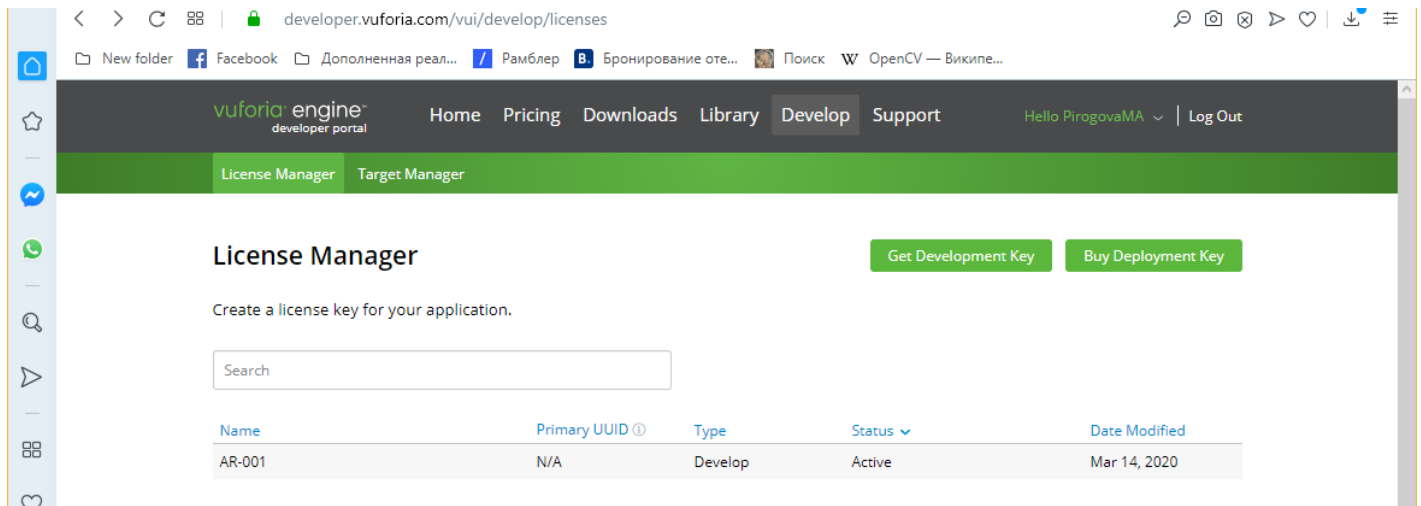
The screenshot shows the Vuforia Developer Portal interface. At the top, there is a navigation bar with links for Home, Pricing, Downloads, Library, Develop, and Support. The user is logged in as 'Hello PirogovaMA' and can click 'Log Out'. Below the navigation bar, there are two tabs: 'License Manager' (selected) and 'Target Manager'. The main content area is titled 'License Manager' and contains two buttons: 'Get Development Key' and 'Buy Deployment Key'. Below these buttons, there is a text prompt: 'Create a license key for your application.' followed by a search input field. A table with the following columns is visible: Name, Primary UUID (with an info icon), Type, Status (with a dropdown arrow), and Date Modified. At the bottom of the page, there is a pagination control showing '25 per page' and a page indicator '<<< 1 >>>'. A footer note states 'Last updated: Today 8:56 PM' with a 'Refresh' link.

В открывшемся окне доступны две вкладки для последовательного выполнения этих процедур – **License Manager** и **Target Manager**. Если ни одного проекта еще не создано, то в открывшемся окне пусто.

Шаги выполнения ЛР №2, часть 2 подобны тем же шагам в ЛР №2, часть 1, особенности и различия в основном находятся в разделе 6.

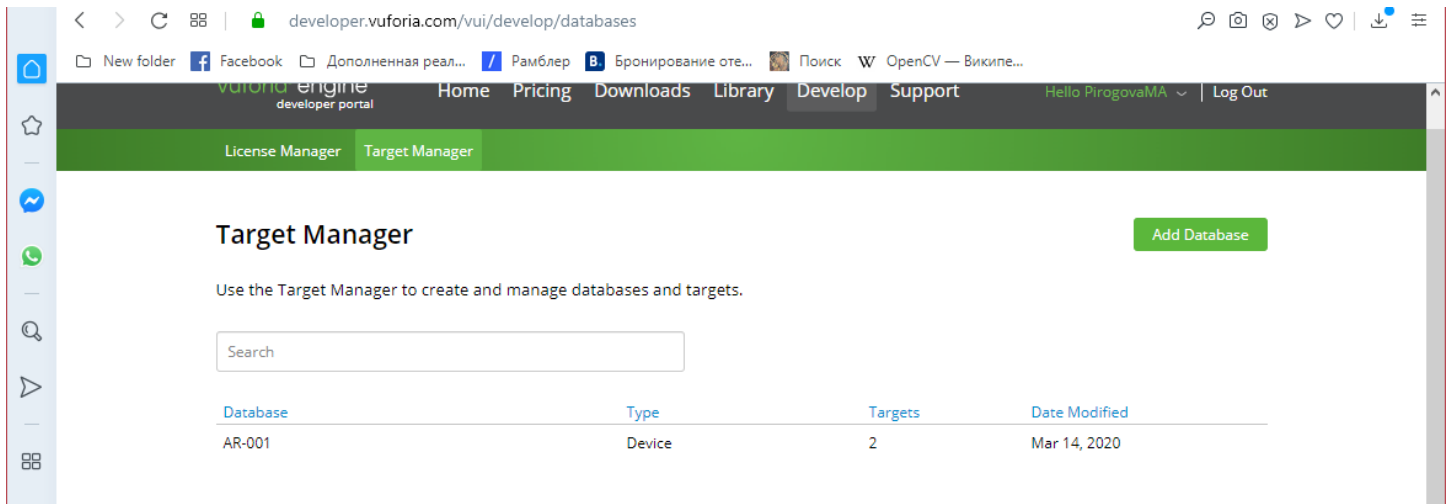
1. Переходим в **License Manager**. Если вы хотите продолжить ранее созданный Проект (в нашем случае – **AR-001**), то воспользуйтесь уже сгенеренной лицензией → выберите проект → в открывшемся окне видите лицензию → сохраняете ее на локальной машине для дальнейшего использования. Если проекта еще нет, или вы хотите работать в новом проекте – повторите процедуру создания и получения лицензии, выполненную в предыдущей ЛР.

ВАЖНО!!! Одна лицензия Проекта в **Vuforia Engine** не запрещает разрабатывать несколько независимых Проектов в **Unity** под этой лицензией. В данном описании ЛР мы продолжаем работать в **Vuforia** - проекте **AR-001**. Вы можете создать новый с новым лицензионным ключем.

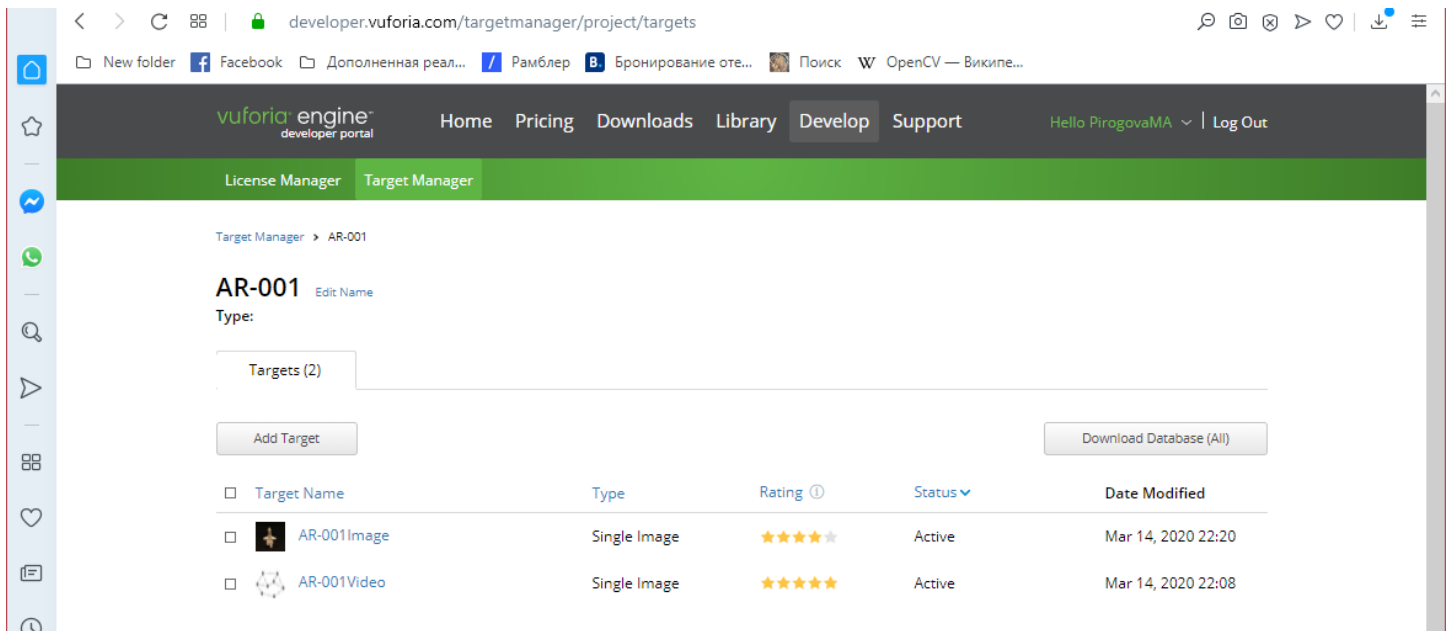


НАПОМИНАНИЕ → Полученный лицензионный ключ необходимо скопировать и сохранить в любом текстовом редакторе (хороший стиль - не **Word!**) для дальнейшего его использования в среде разработки (редакторе) **Unity 3D** на вашем локальном компьютере.

2. Переходим в менеджер меток – **Target Manager**:



База данных меток для **Vuforia**-проекта **AR-001** уже создана:



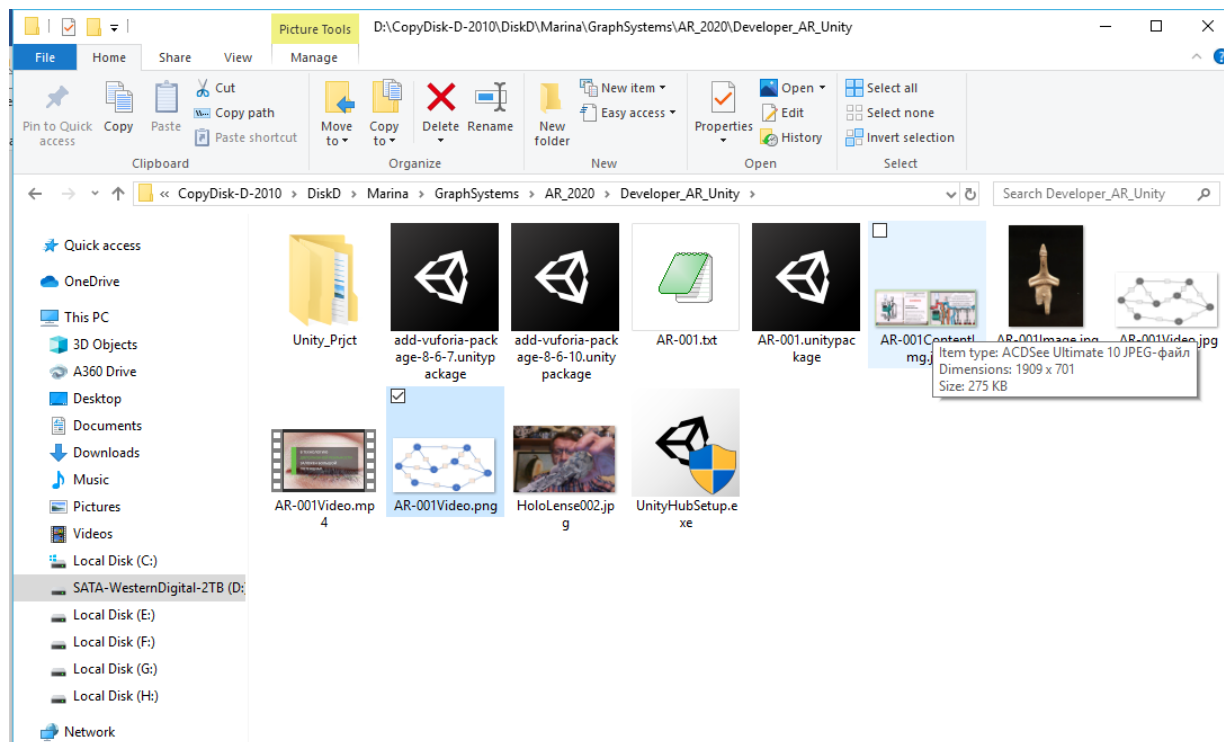
Можно использовать в этой ЛР уже созданные таргеты или добавить новый. В предыдущей ЛР в проекте **AR-001** мы уже создали таргет для второй части ЛРН№2. Это файл - **AR-001Image**. Если вы хотите добавить еще один таргет – используйте кнопку **Add Target** и заново выполните загрузку БД таргетов (**Download Database (ALL)**), как это описано в ЛР№2, часть1.

Если вы создаете БД таргетов заново или добавляете в уже имеющуюся БД еще один таргет – см. Описание Предыдущей ЛР.

Не забывайте, что в результате описываемых процедур (на каком бы шаге вы их не делали – здесь, или в предыдущей ЛР), формируется образ Базы Данных **Vuforia**-Проекта (у нас - **AR-001**) для работы в среде **Unity 3D** в специальном формате - **.unitypackage**, который выгружается из облака **Vuforia** для сохранения в локальной файловой системе.

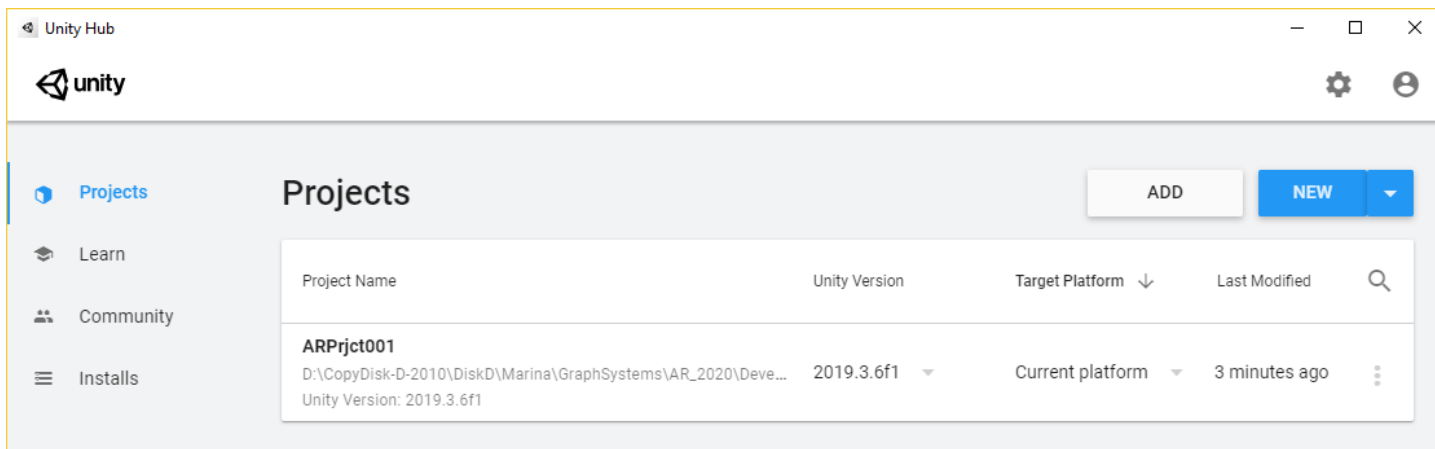
В результате выполнения двух описанных процедур мы получили из облака **Vuforia** для локальной работы в **Unity 3D** следующие ресурсы:

- Лицензионный ключ;
- Базу данных меток - два таргета: **AR-001Video.jpg**, **AR-001Image.jpg**;
- Заранее подготовленные объекты контента для воспроизведения их на экране **Android**-устройства в Приложении ДР: на данном этапе это **2D-Изображение** в формате **.jpg** – файл **AR-001ContentImg.jpg**.



3. Стартуем приложение **Unity 3D**.

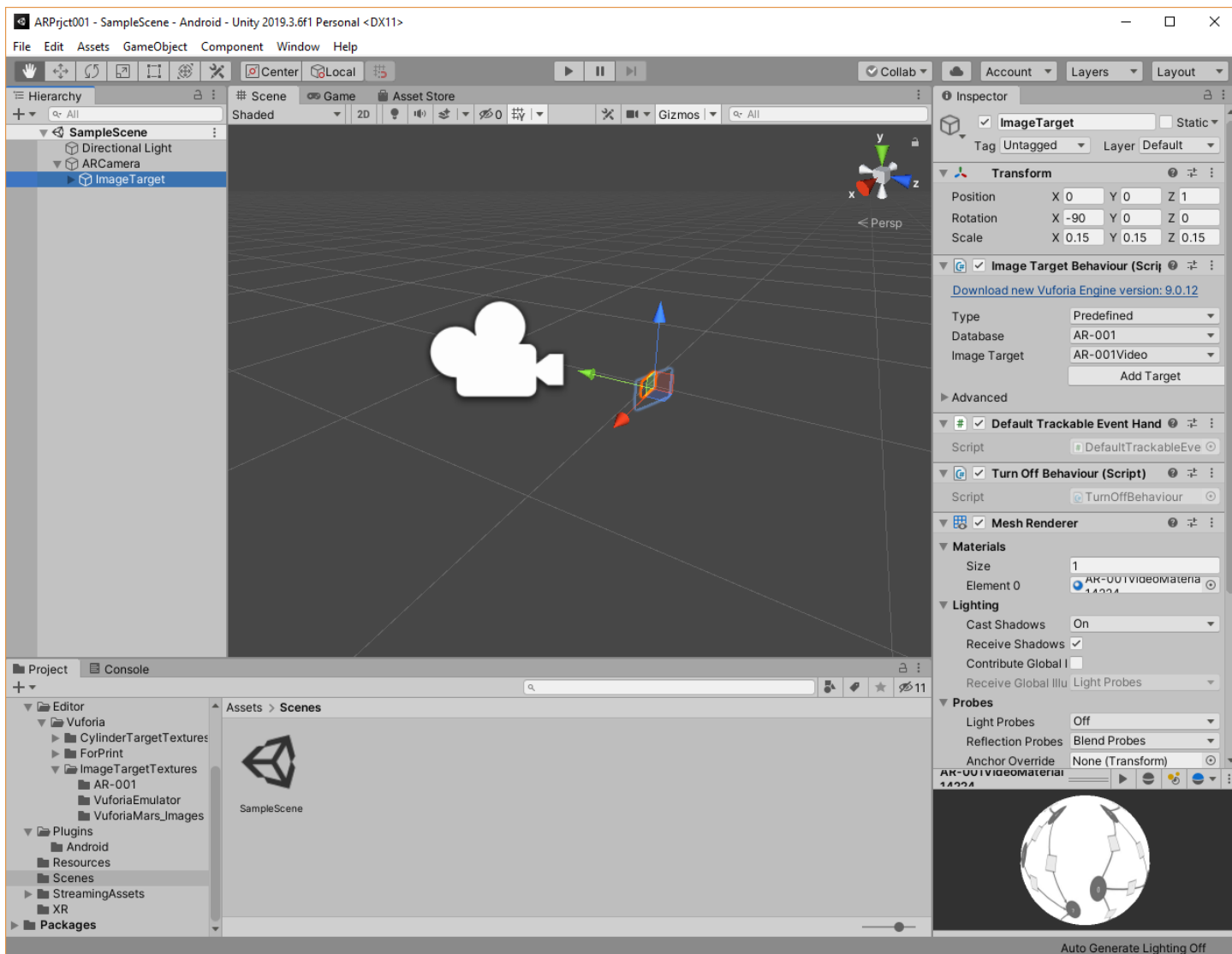
Вы увидите в открывшемся окне заведенный в первой части ДР№2 ранее проект.



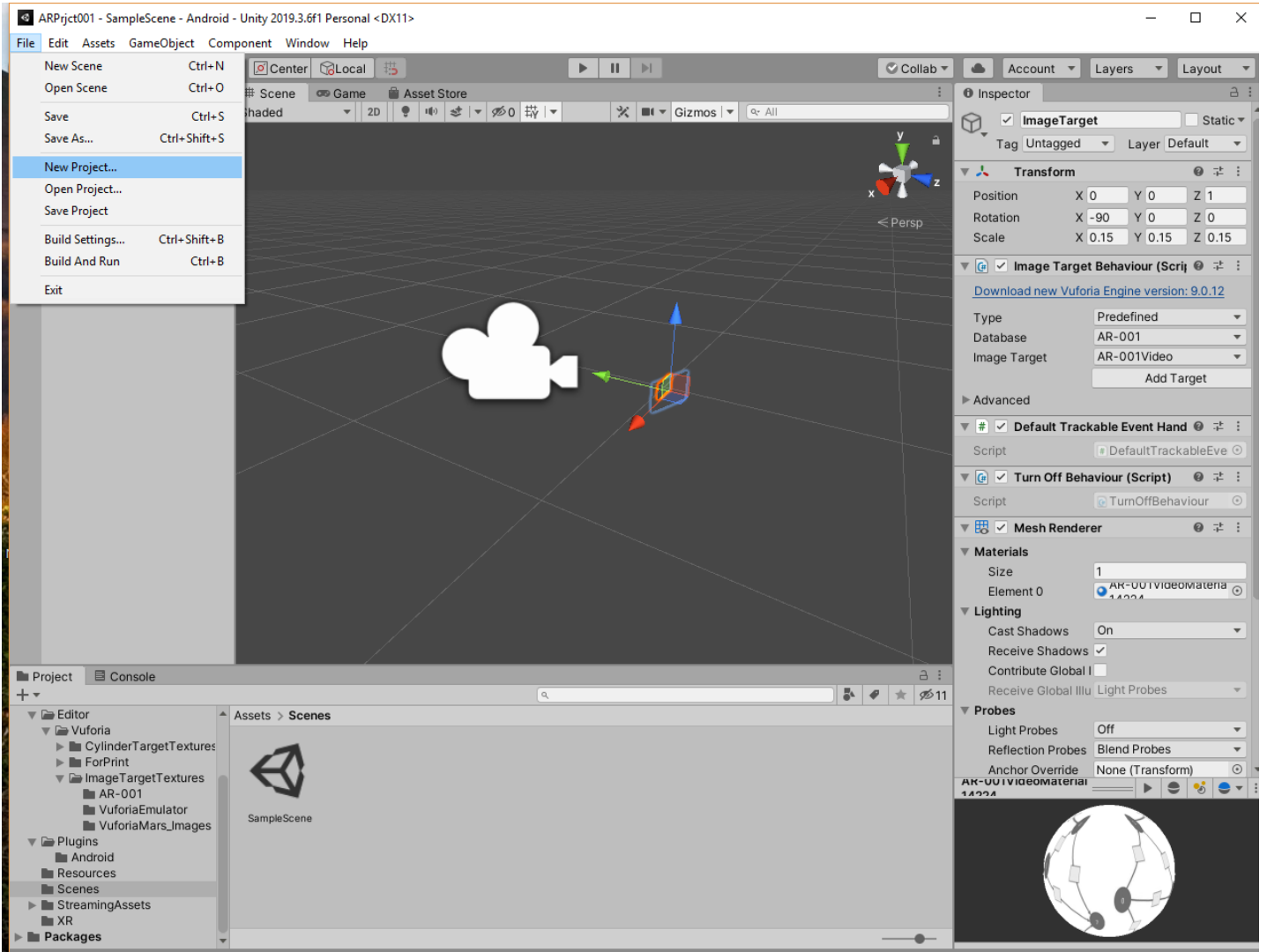
На данном этапе возможно:

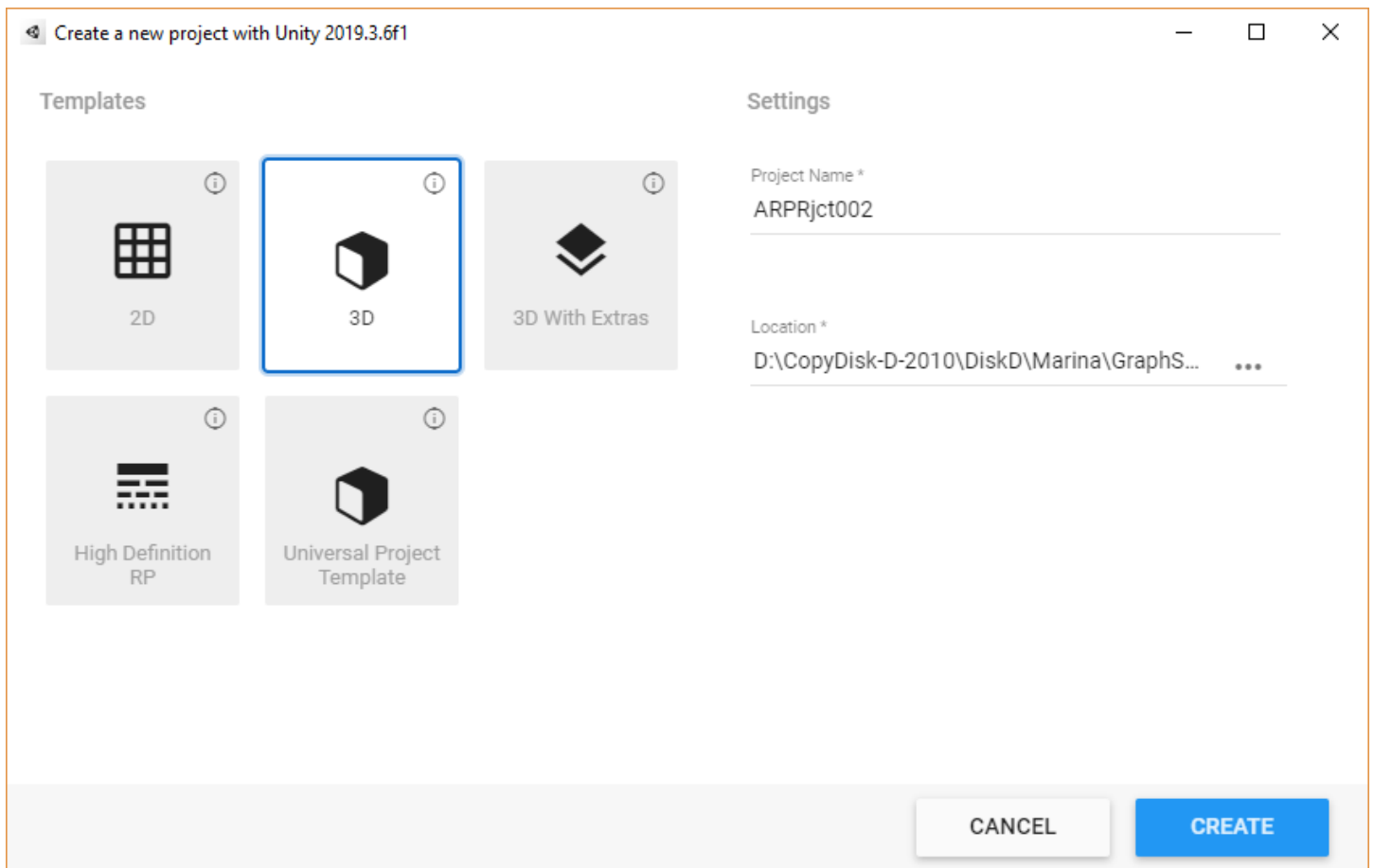
- Продолжать работу в созданном ранее проекте **ARPRjct001**;
- Создать новый проект нажатием клавиши **NEW** – см. описание ЛР №2, часть 1;
- Создать новый проект в рамках работы в уже созданном проекте (**ARPRjct001**).

В данном случае стартуем проект **ARPRjct001** (клик левой клавишей по имени в списке) и для выполнения ЛР №2, Часть 2, создадим новый проект **ARPRjct002**:

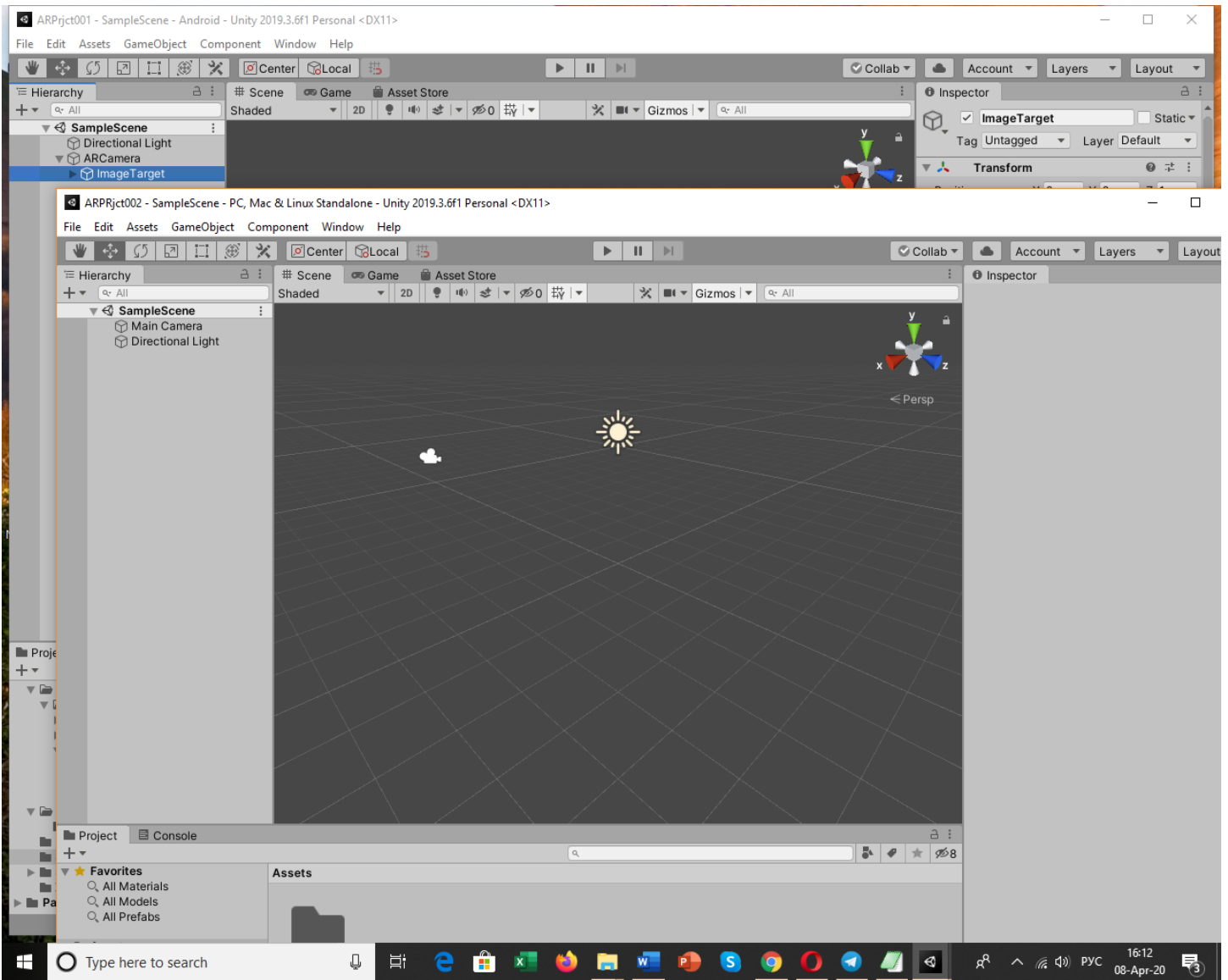


В ленте меню, в закладке **File**, выбираем **New Project**:





В появившемся поле заводим новое имя проекта **ARPRjct002** → **CREATE**.

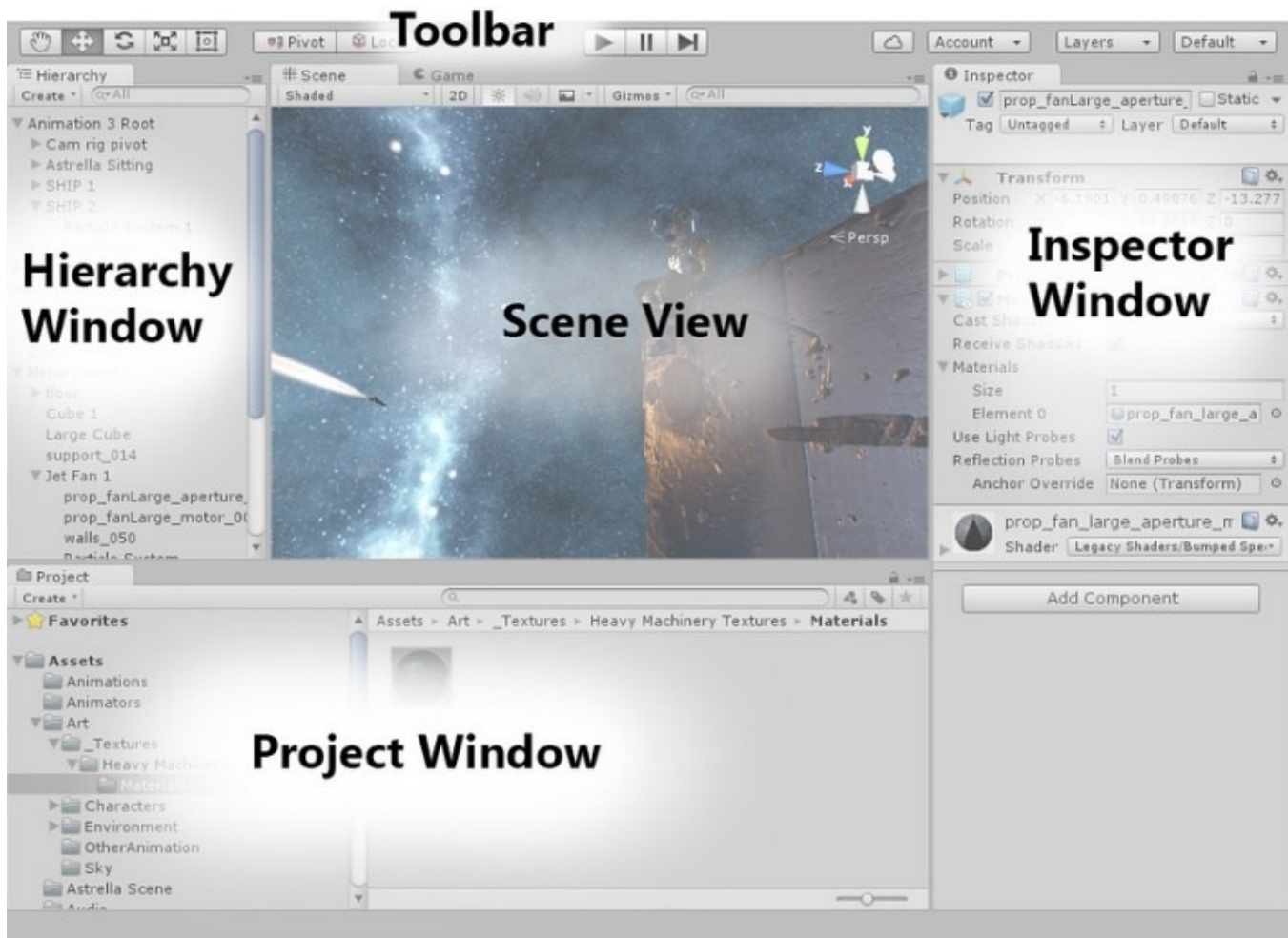


Не забудьте закрыть старый проект **ARPRjct001**.

ВАЖНО!! Можно было в рамках старого проекта **ARPRjct001** стартовать новую сцену: **File**→**New Scene**. При этом объем проекта естественно будет возрастать.

Документацию по работе в редакторе – см. по ссылке:

<https://docs.unity3d.com/Manual/LearningtheInterface.html>



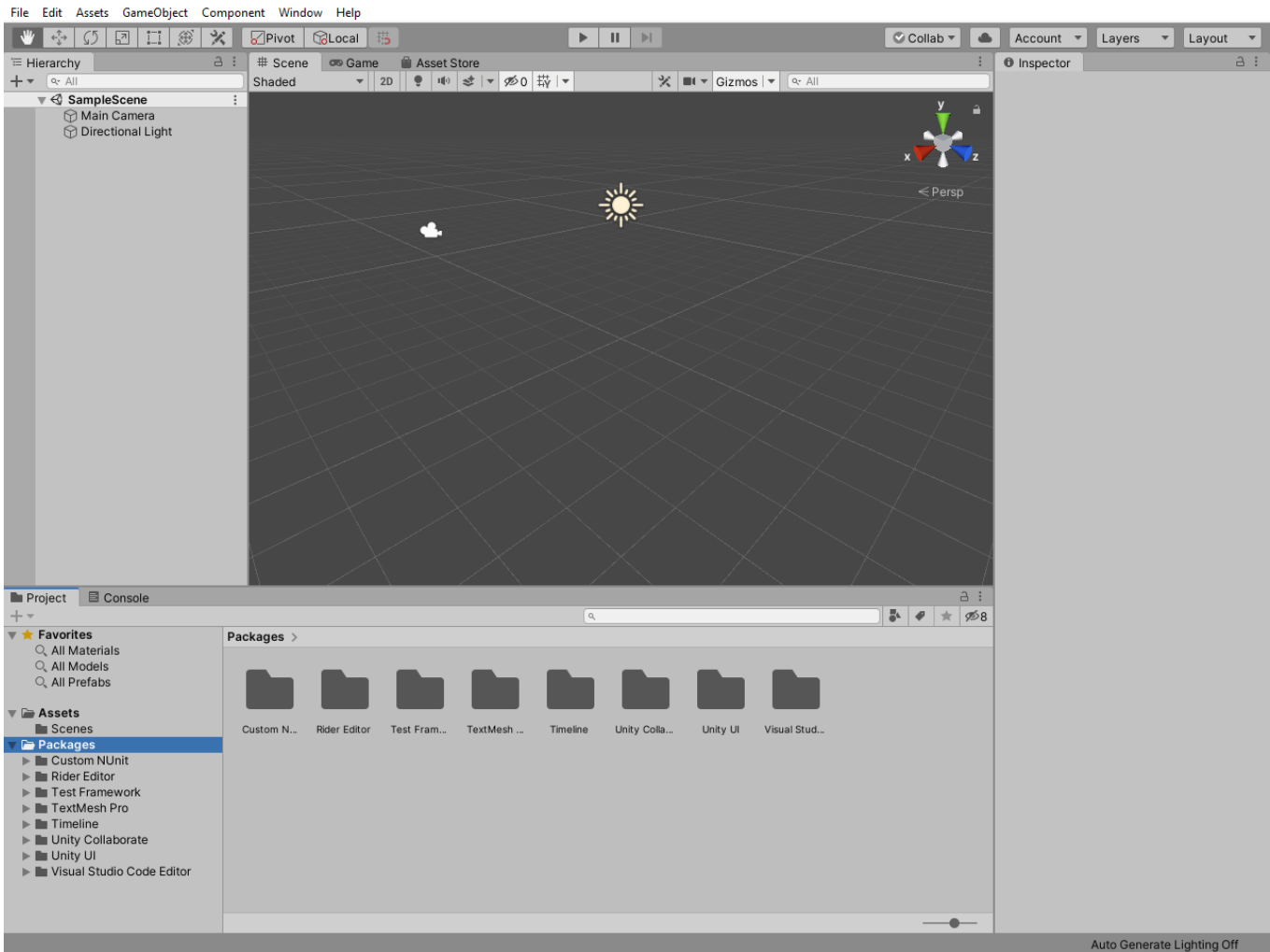
Обратите внимание - начальные действия по организации среды игрового движка **Unity 3D** для нашего проекта (Приложение ДР) происходят с использованием закладки **Scene** (область **Scene View**) и группы элементов линейки **Toolbar**, объединенных в группу с названием **GameObject**.

Без дополнительных настроек редактор **Unity 3D** работает в режиме виртуальной реальности. По условиям Задания ЛР наше **Android**-устройство в Проекте должно работать в режиме **Дополненной реальности**. Как отмечалось выше, за режим ДР отвечает **Vuforia**. Это означает, что в список объектов **Unity 3D** необходимо добавить специфические объекты **Vuforia**. А работу с ними поддержать действующими лицензиями **Vuforia**.

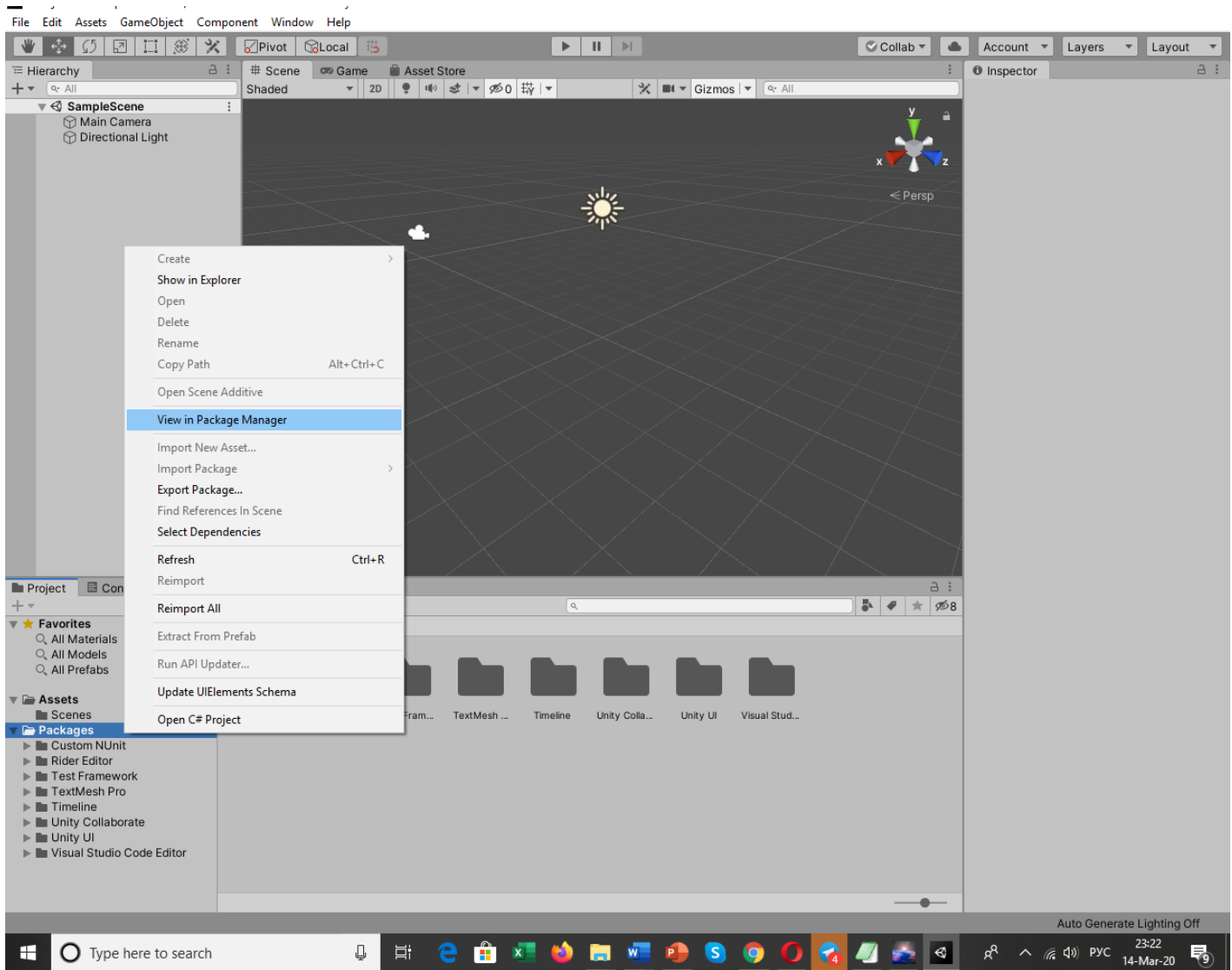
Для этого необходимо файлы типа **.unitypackage** загрузить в **Unity 3D** с помощью входящего в ее состав средства **Package Manager**. Эти файлы вы заранее подготовили (в ЛР2) и загрузили из облака **Vuforia** на локальную машину (см. ЛР1).

Использование **Package Manager** для подключения **Vuforia** предполагает выполнение следующих шагов:

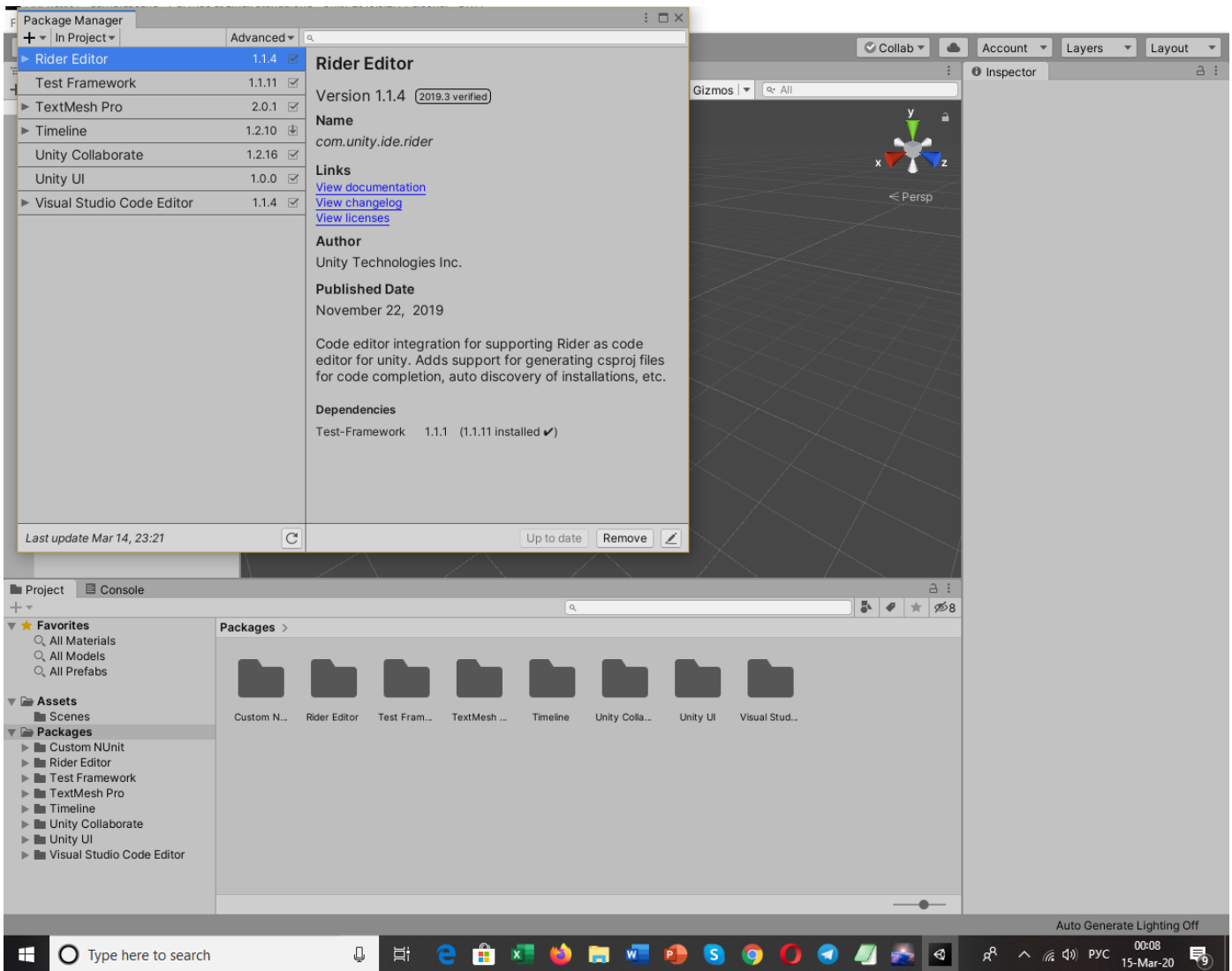
- В области **Project Window** в строке **Packages** → кликаем правой клавишей мыши (**RMB**)



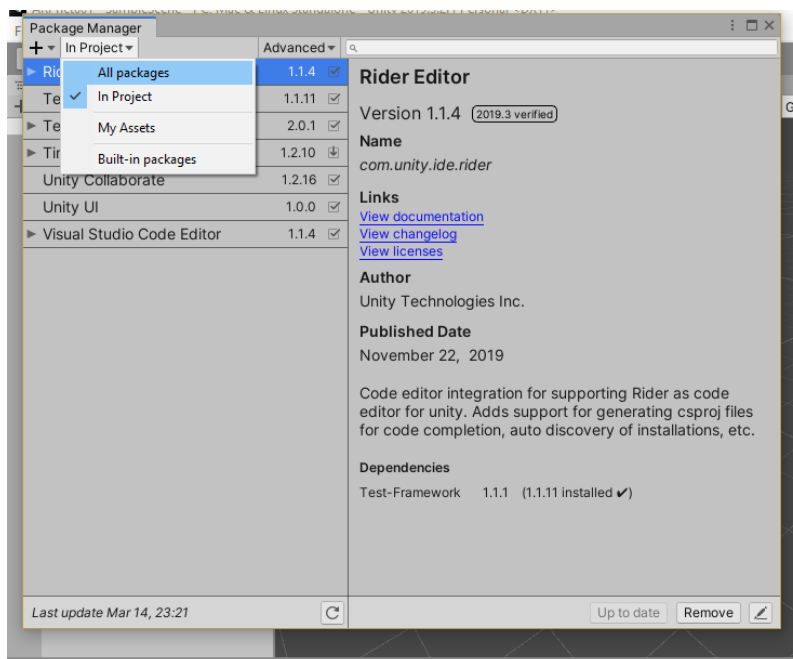
- В списке возможных действий выбираем **View in Package Manager**:



- В результате появится окно **Package Manager**:



- В окне **Package Manager** в падающем меню необходимо выбрать режим **All packages**, в котором возможно подключение пакетов Vuforia:



- В результате в списке **All packages** выбираем инсталляцию (кнопка **Install**) **Vuforia Engine AR**

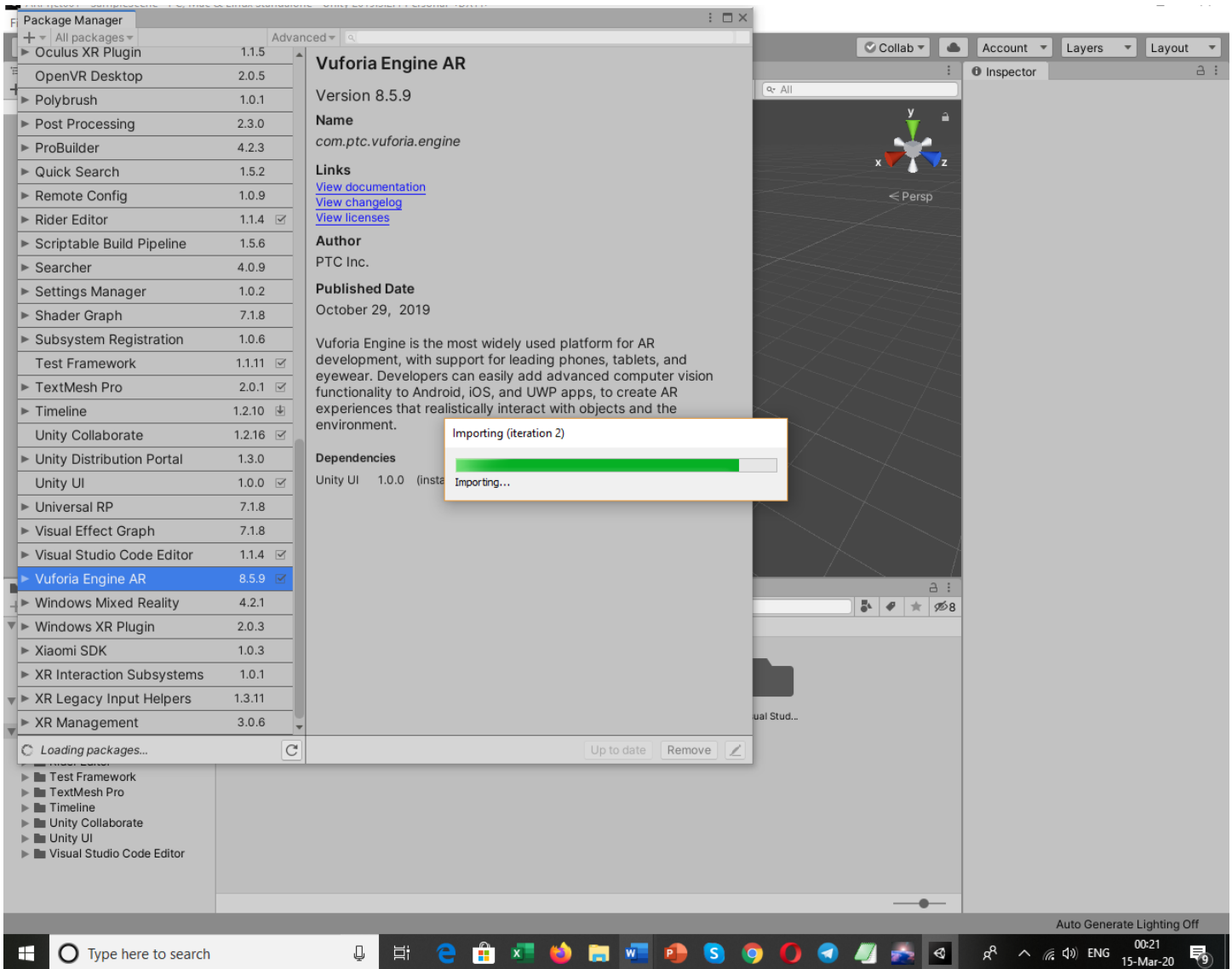
The screenshot shows the Unity Package Manager window. On the left, a list of packages is displayed with columns for package name and version. The package 'Vuforia Engine AR' with version '8.5.9' is highlighted in blue. On the right, the details for 'Vuforia Engine AR' are shown, including its version (8.5.9), name, author (PTC Inc.), published date (October 29, 2019), a description, and a list of dependencies (Unity UI 1.0.0, which is installed).

Package Name	Version
Oculus Desktop	2.38.4
Oculus XR Plugin	1.1.5
OpenVR Desktop	2.0.5
Polybrush	1.0.1
Post Processing	2.3.0
ProBuilder	4.2.3
Quick Search	1.5.2
Remote Config	1.0.9
Rider Editor	1.1.4 <input checked="" type="checkbox"/>
Scriptable Build Pipeline	1.5.6
Searcher	4.0.9
Settings Manager	1.0.2
Shader Graph	7.1.8
Subsystem Registration	1.0.6
Test Framework	1.1.11 <input checked="" type="checkbox"/>
TextMesh Pro	2.0.1 <input checked="" type="checkbox"/>
Timeline	1.2.10 <input type="checkbox"/>
Unity Collaborate	1.2.16 <input checked="" type="checkbox"/>
Unity Distribution Portal	1.3.0
Unity UI	1.0.0 <input checked="" type="checkbox"/>
Universal RP	7.1.8
Visual Effect Graph	7.1.8
Visual Studio Code Editor	1.1.4 <input checked="" type="checkbox"/>
Vuforia Engine AR	8.5.9
Windows Mixed Reality	4.2.1
Windows XR Plugin	2.0.3
Xiaomi SDK	1.0.3
XR Interaction Subsystems	1.0.1
XR Legacy Input Helpers	1.3.11

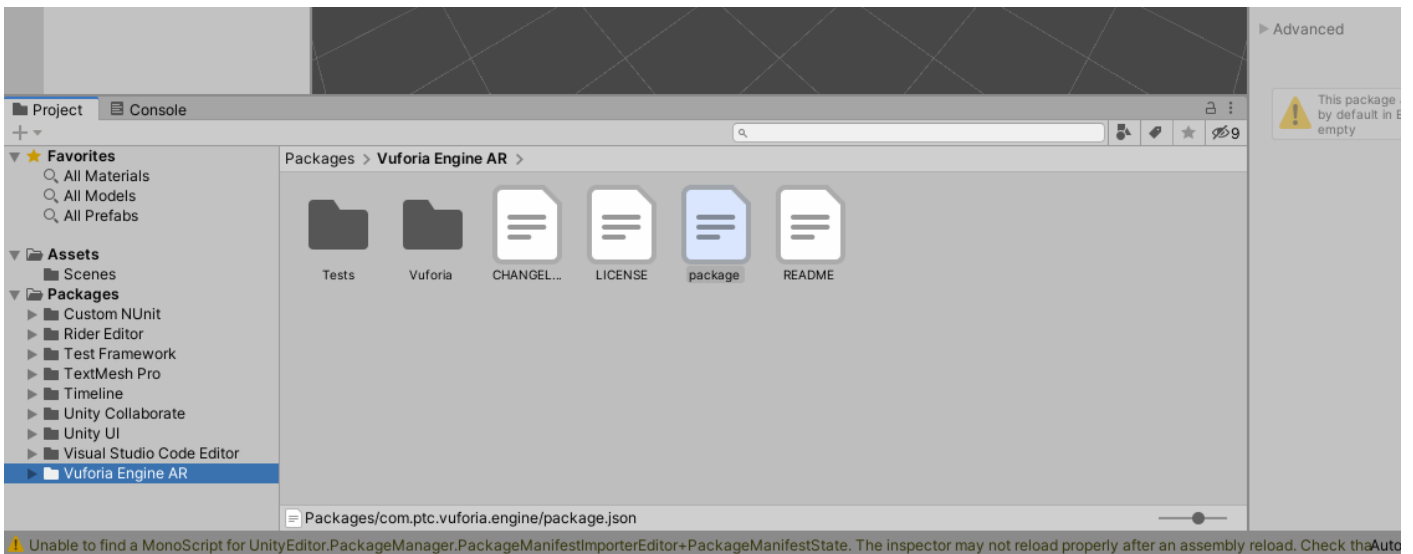
Vuforia Engine AR
Version 8.5.9
Name
com.ptc.vuforia.engine
Links
[View documentation](#)
[View changelog](#)
[View licenses](#)
Author
PTC Inc.
Published Date
October 29, 2019
Vuforia Engine is the most widely used platform for AR development, with support for leading phones, tablets, and eyewear. Developers can easily add advanced computer vision functionality to Android, iOS, and UWP apps, to create AR experiences that realistically interact with objects and the environment.
Dependencies
Unity UI 1.0.0 (installed)

Last update Mar 14, 23:21

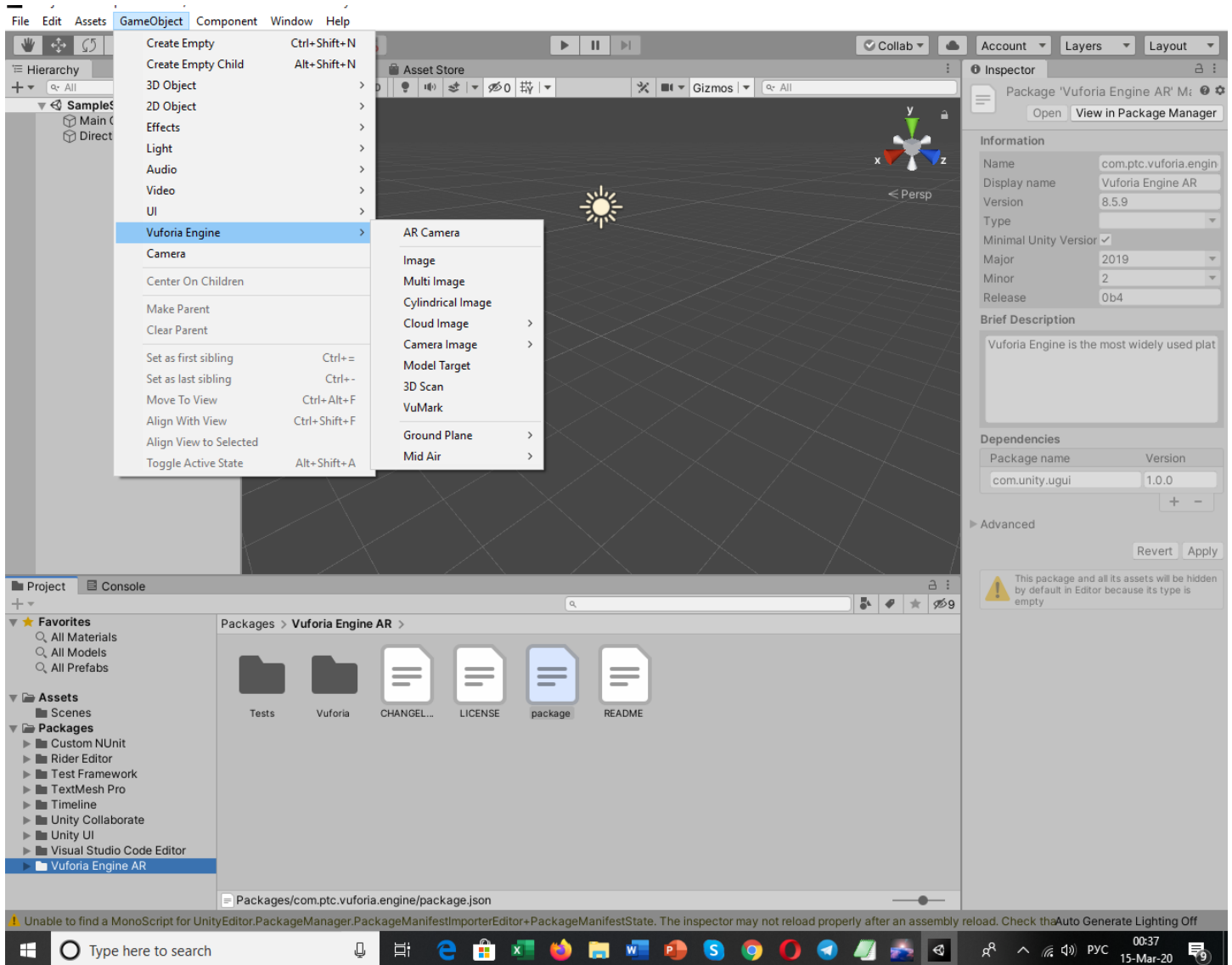
- Происходит инсталляция пакета **Vuforia**:



- В результате мы получаем обновленный список пакетов в области **Project Window** редактора **Unity 3D**:



- Убедимся, что у **Unity 3D** появились новые **Game Object** из пакета **Vuforia**:



На этом работа с **Package Manager** завершена. Базовые настройки для работы **Unity 3D** с **Vuforia** выполнены.

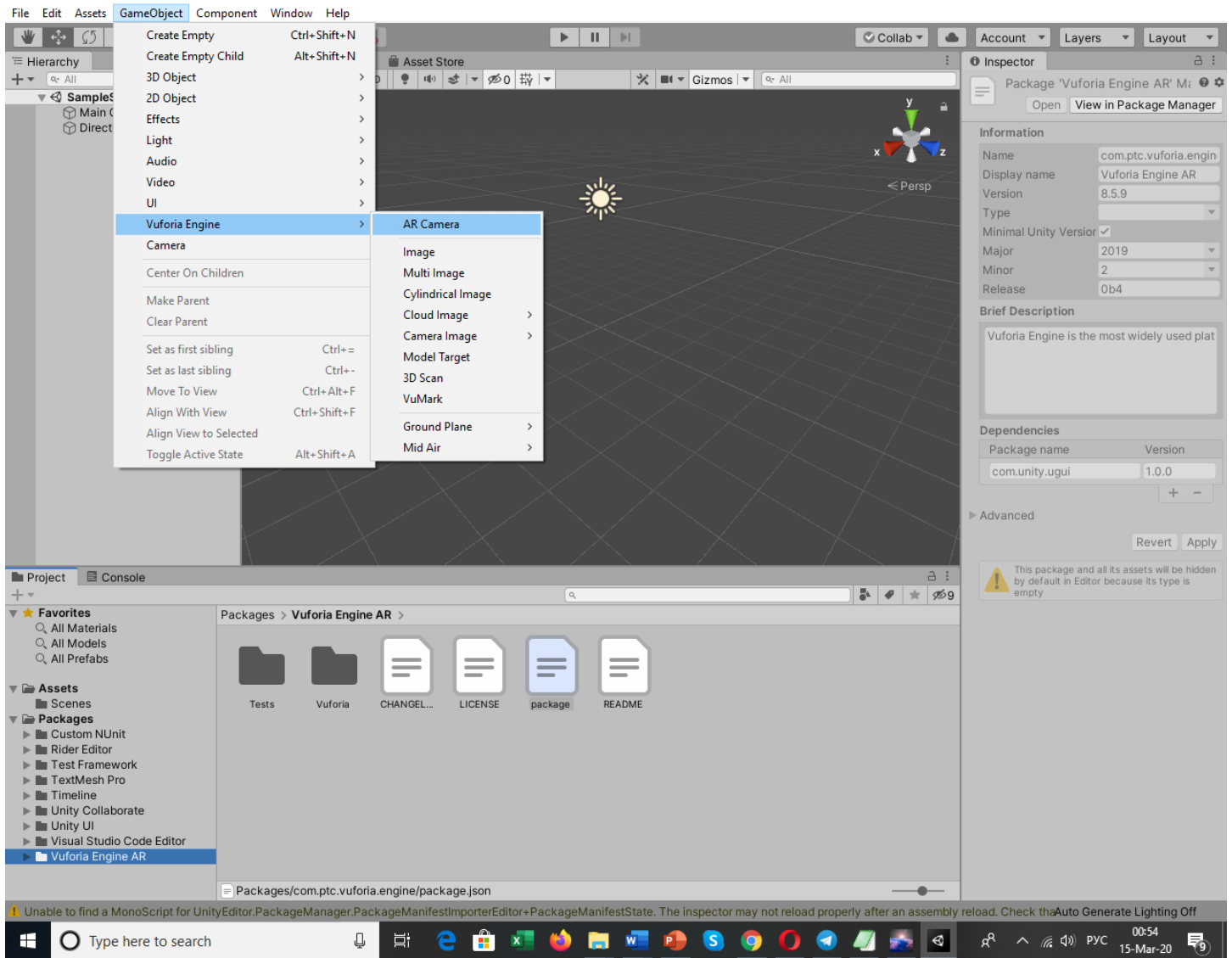
4. Создание Приложения ДР в среде **Vuforia + Unity 3D** для просмотра **2D-Изображения**.

В разрабатываемом Приложении ДР **Main Camera** редактора **Unity 3D** не позволяет выполнять визуализацию реального окружения, а работает только с **виртуальными объектами сцены в виртуальном пространстве**.

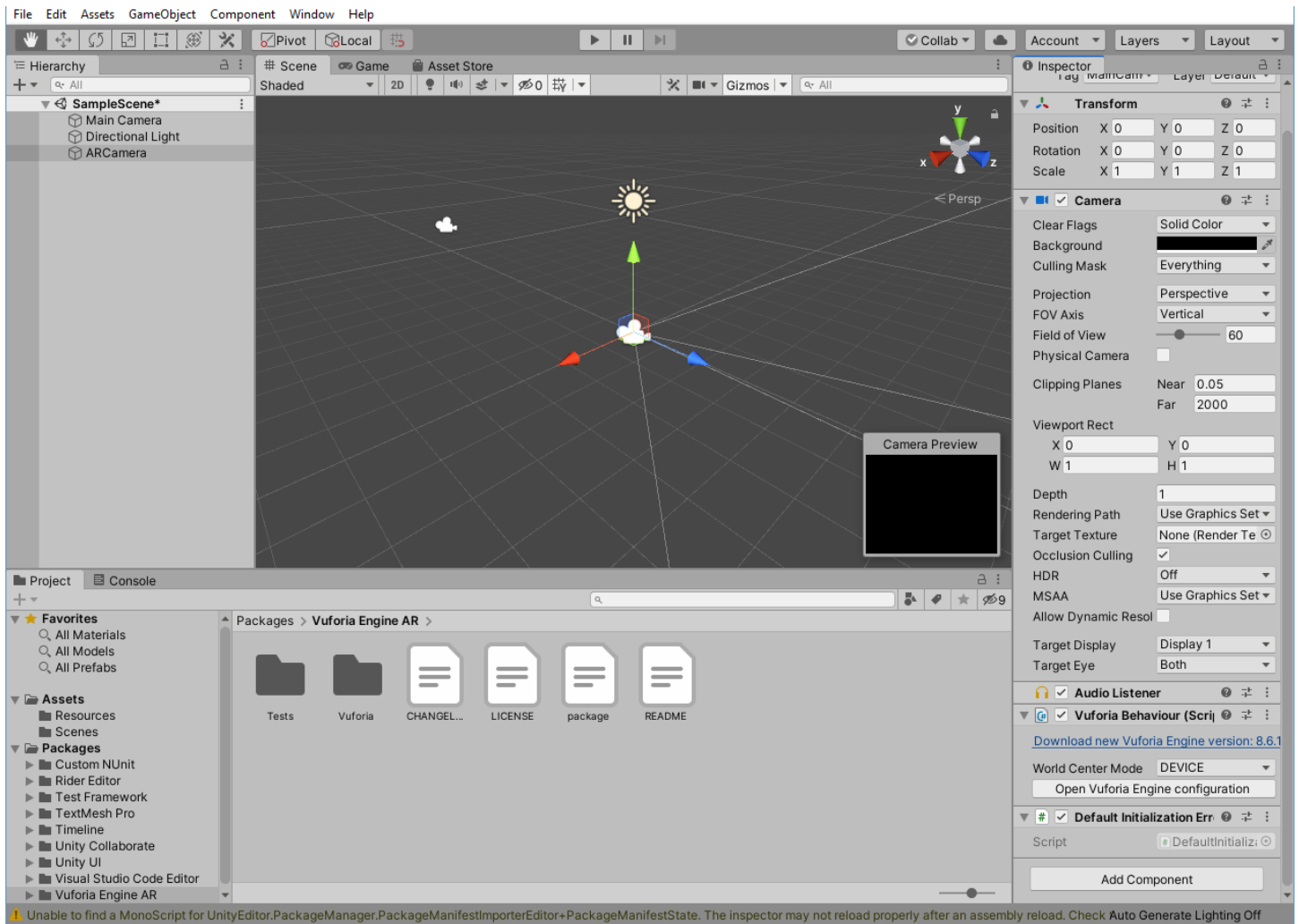
Для работы в режиме ДР необходима «камера», способная выполнять визуализацию виртуальных объектов сцены на фоне реального окружения. Для переключения нашего проекта из режима виртуальной реальности в режим Дополненной реальности заменим в наборе иерархических объектов (область экрана редактора **Hierarchy**) камеру **Main Camera** на камеру дополненной реальности из набора настроек **Vuforia**.

Для этого выполним следующие вызовы:

- **Game Object** → **Vuforia** → **AR Camera**

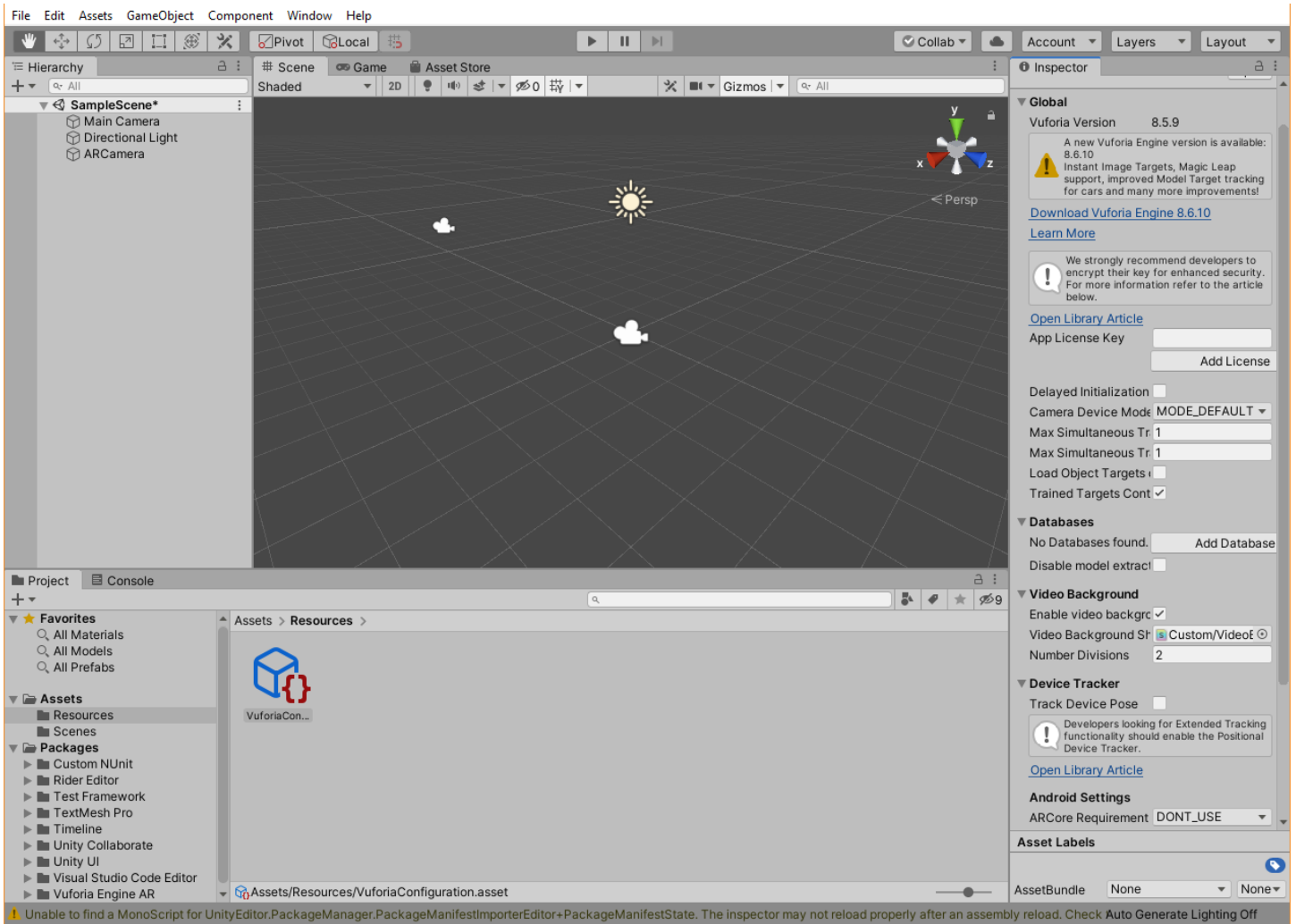


В области **Hierarchy** появляется **Vuforia**-объект **ARCamera**.

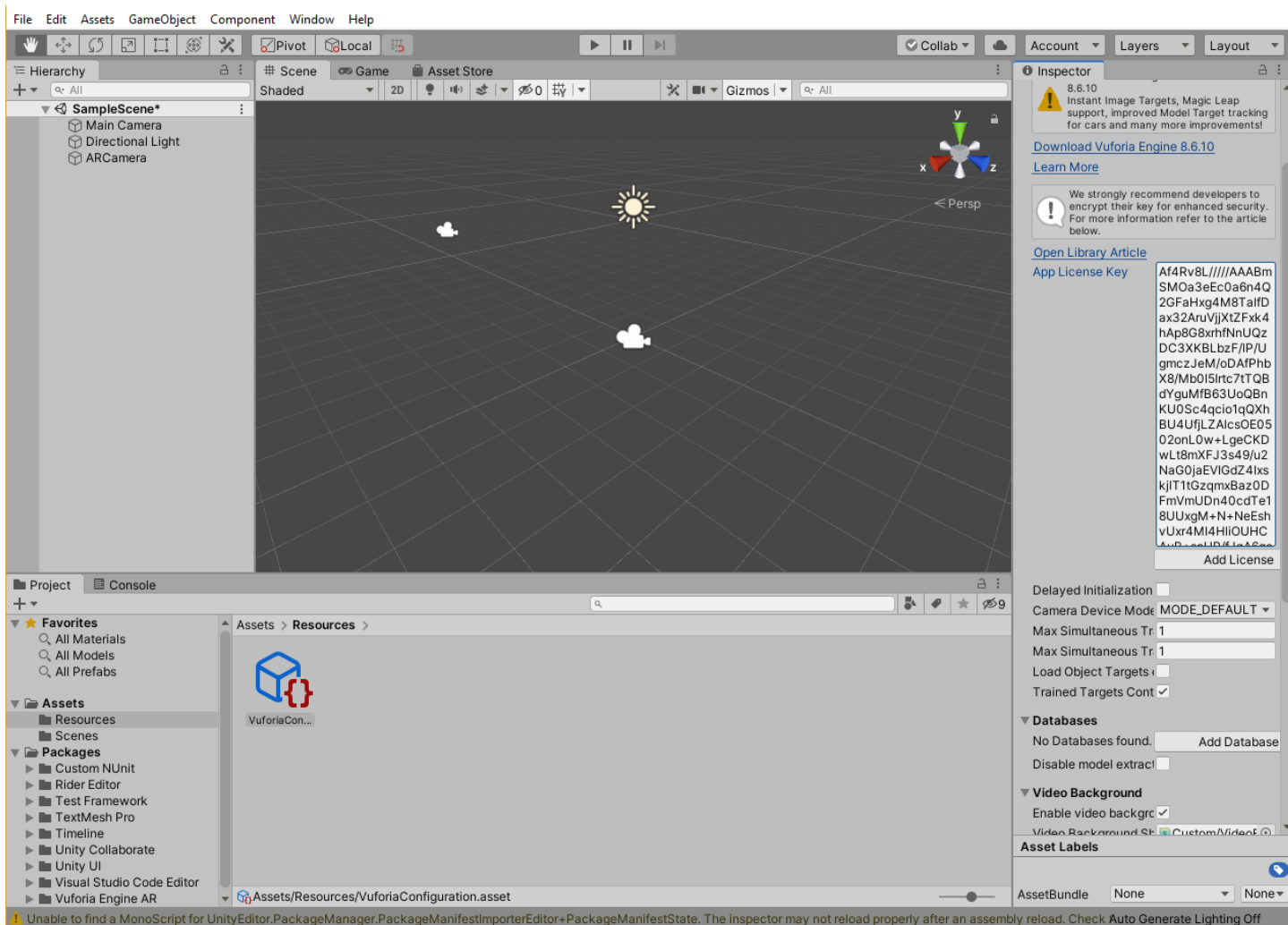


Для работы с объектами **Vuforia** необходимо перевести их под действие сгенерированной вами выше лицензии. Делать мы это будем через верхний уровень объектов **Vuforia**. Установленный объект **ARCamera** является верхним уровнем иерархии всех объектов **Vuforia** разрабатываемой сцены:

- В области **Hierarchy** выбираем **ARCamera** → в области **Inspector** находим и выбираем поле **Open Vuforia Engine configuration**. Теперь в **Inspector**'е появилось окно ввода лицензии **App License Key**:



- Ранее на локальной машине вы сохранили текст лицензии (**license key**). Находим этот текст и копируем в это поле.

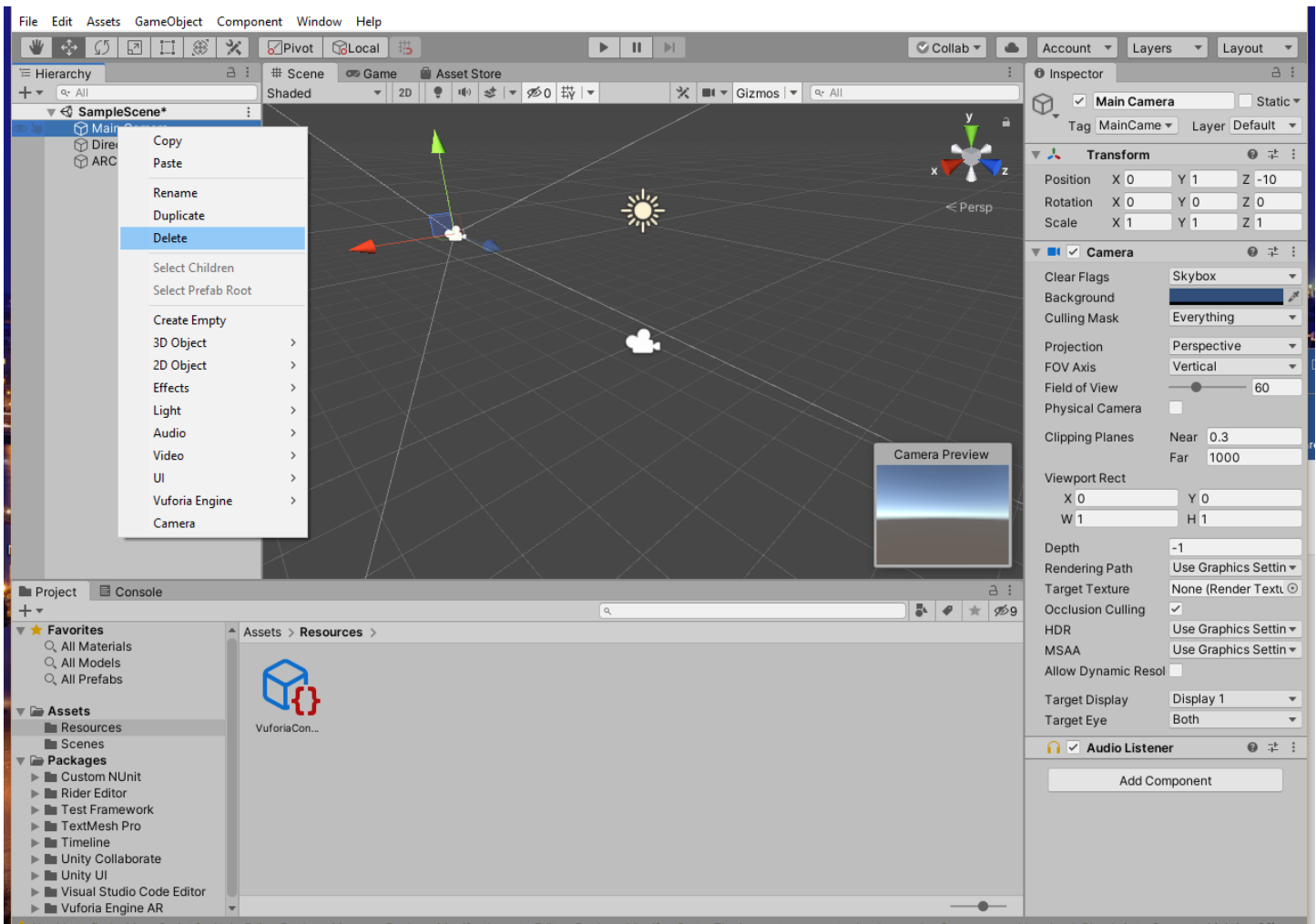


ВАЖНО!! Если Вы выберете на этом этапе поле **Add License**, то вновь попадете на этап ее генерации, который в данной ЛР был выполнен в первую очередь. Иными словами, лицензия автоматом не устанавливается при конфигурировании, ее необходимо скопировать в поле **App License Key**, предварительно сгенерив.

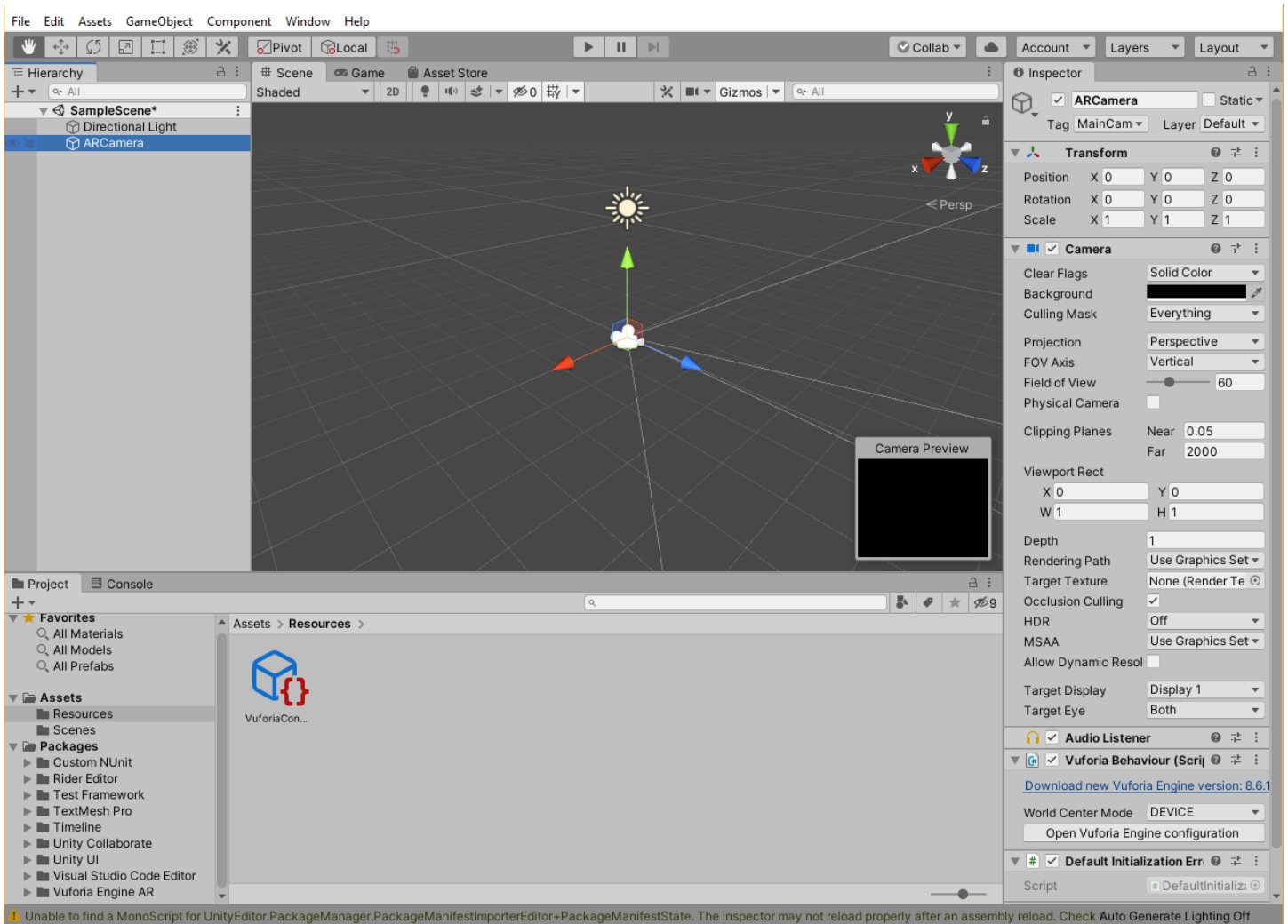
Но если вы забыли это сделать в свое время – здесь у вас есть возможность исправить эту ошибку.

Итак, лицензию вы установили.

Теперь выполняем замену камеры: оставляем **AR Camera** и удаляем **Main Camera**. **Main Camera** должна быть удалена обычным способом – маркируем **Main Camera** в меню иерархии и выбираем в падающем контекстном меню (**RMB**) функцию **Delete**.

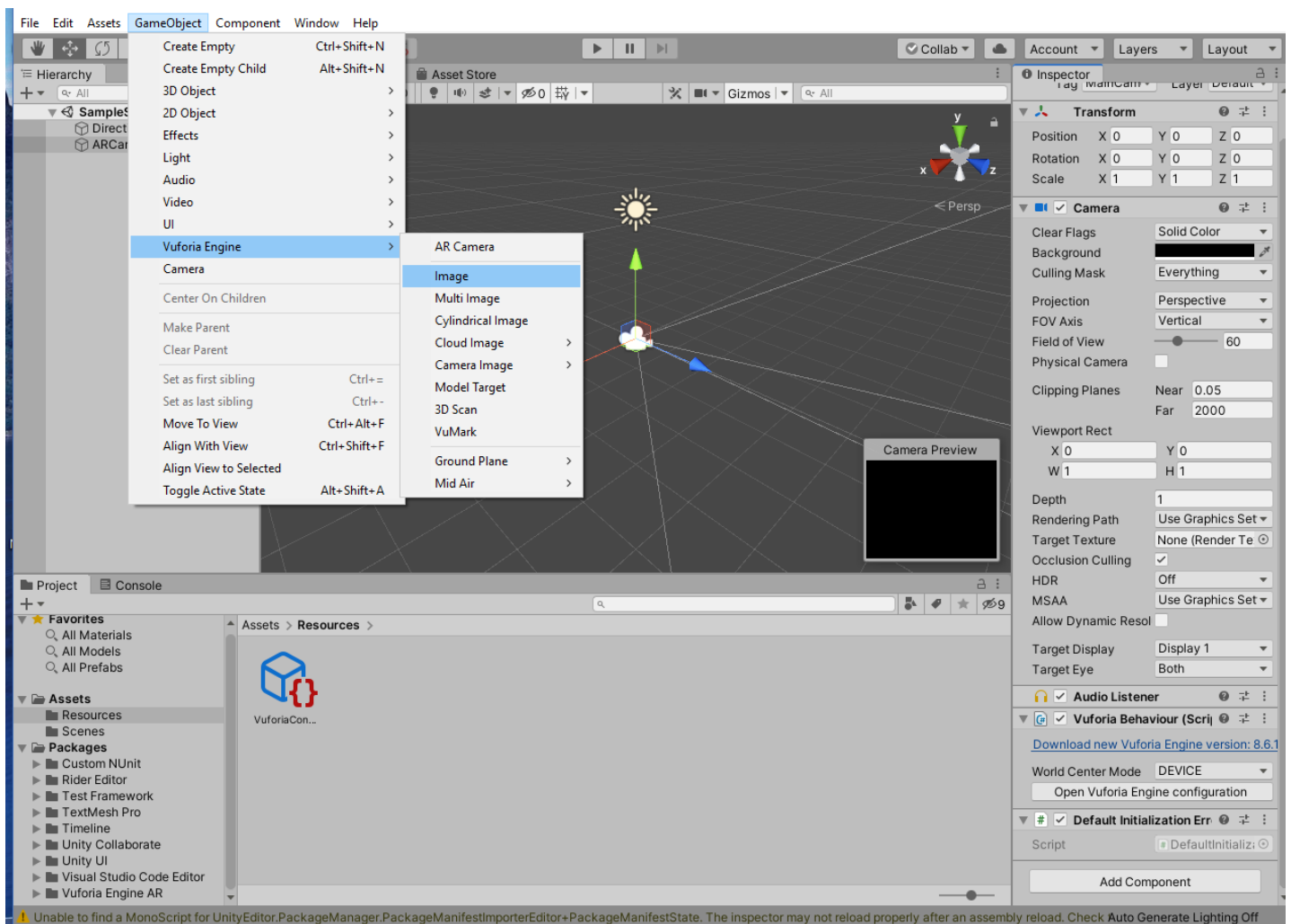


В результате в **Hierarchy** остается только одна камера, свойства которой отображены в области **Inspector**.

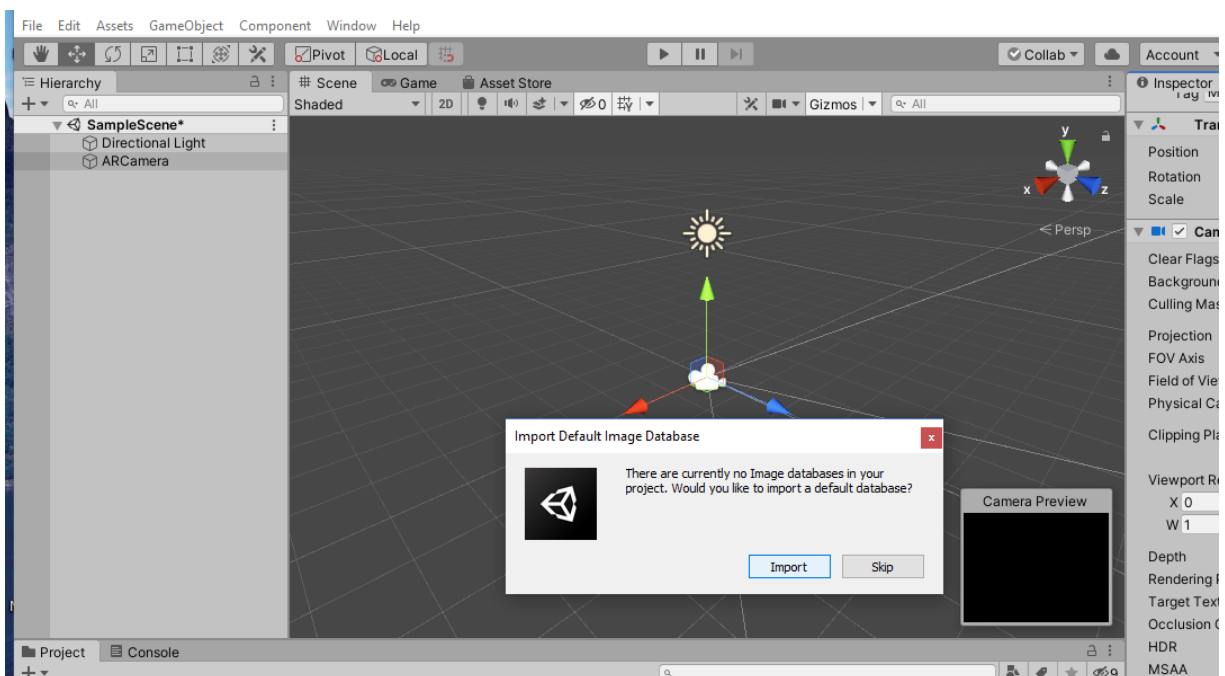


5. Далее необходимо загрузить Базу данных таргетов.

Для этого осуществляем вызовы в закладке **Game Object** → **Vuforia** → **Image**:

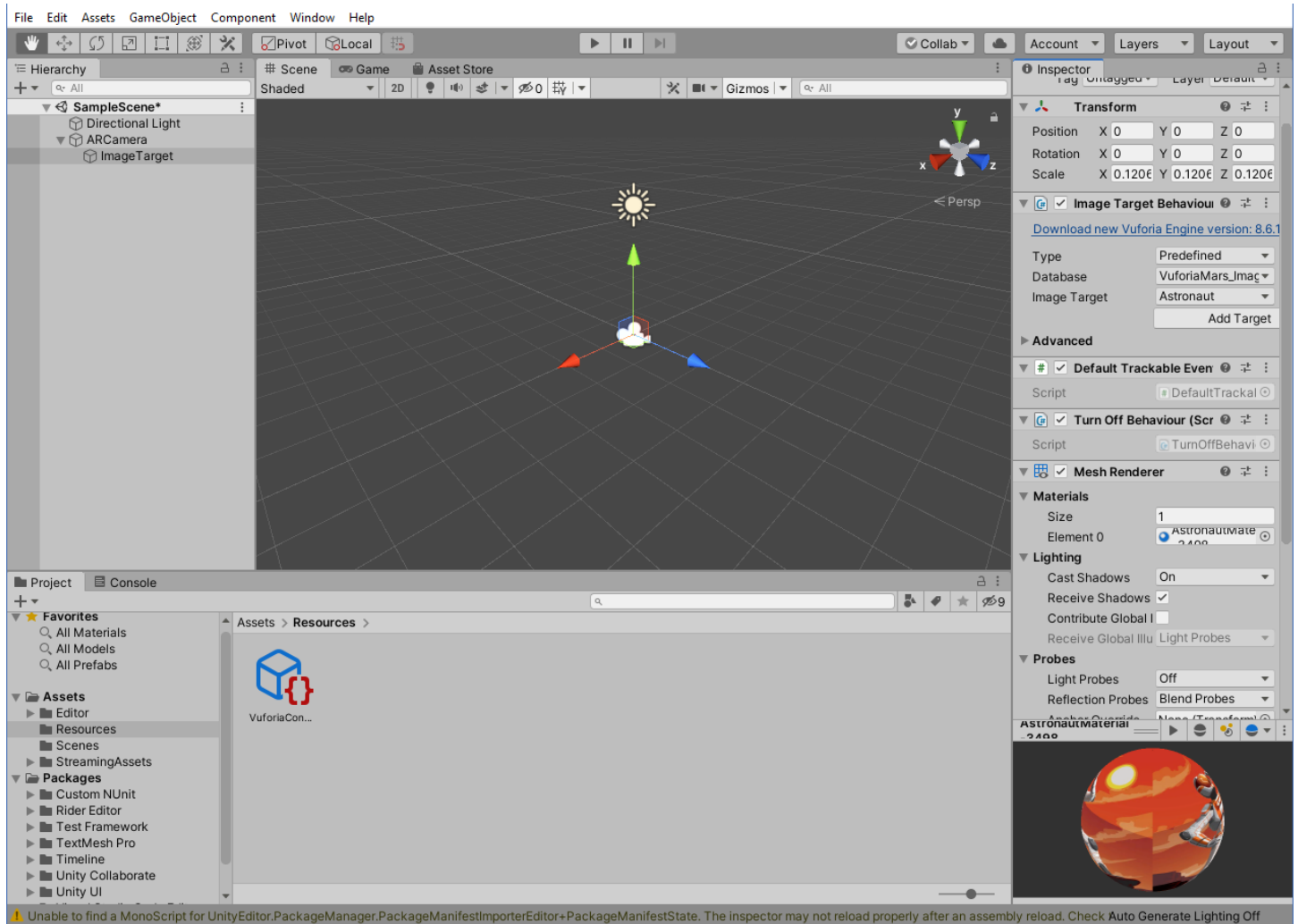


Получаем сообщение:



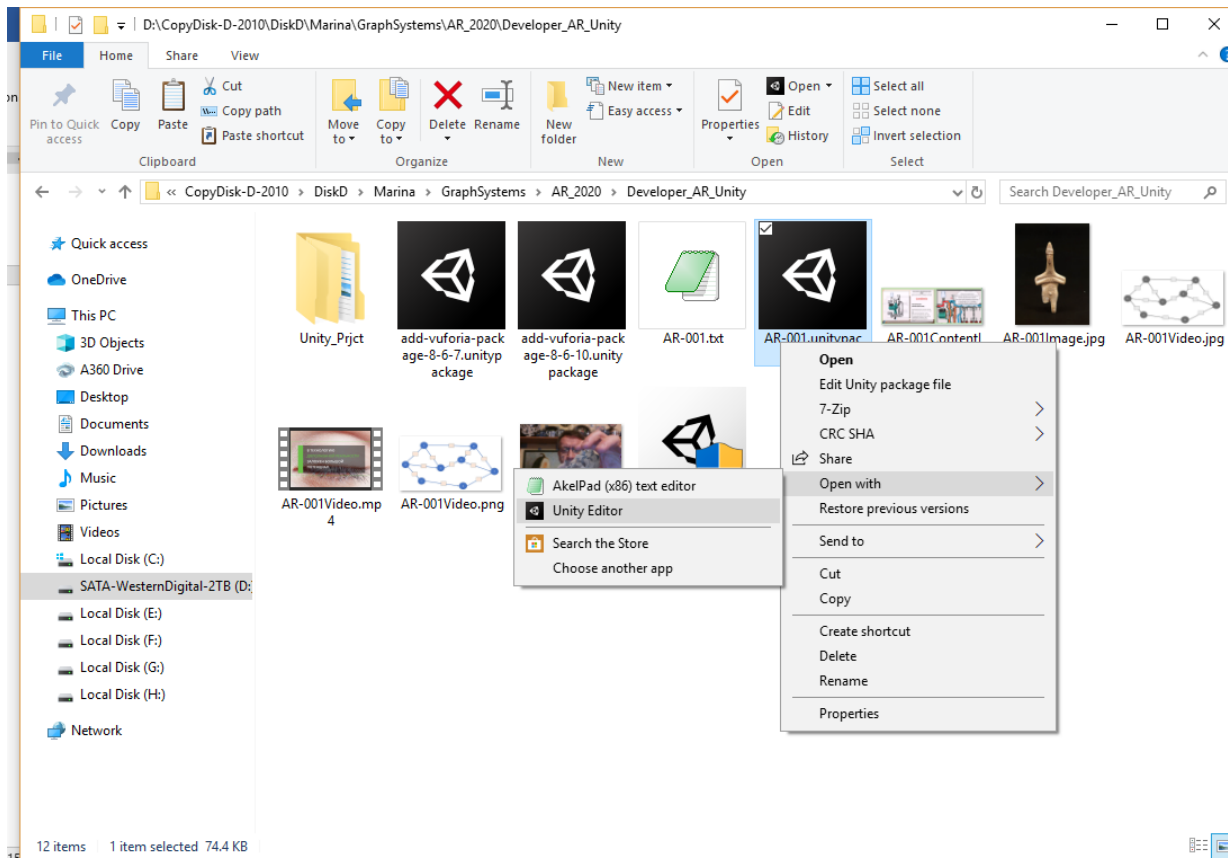
В появившемся запросе нажмите кнопку **Import**.

В результате загружается БД таргетов по умолчанию («Астронавт»).

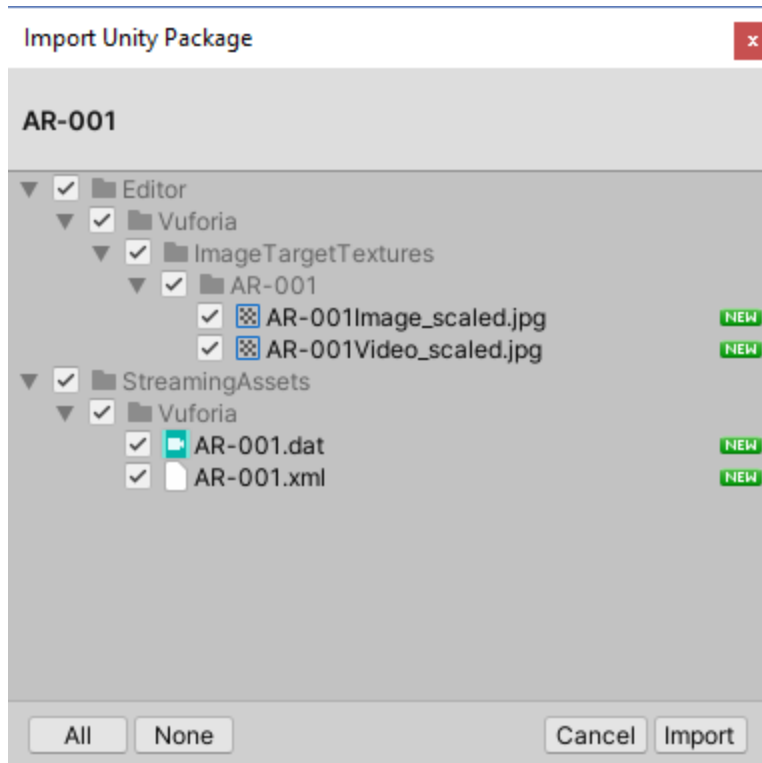


Добавляем подготовленную заранее БД таргетов в сцену нашего проекта **AR-001**.

Для этого в локальной файловой структуре находим эту ранее подготовленную БД таргетов (в конце шага 2 - **AR-001.unitypackage**), по правой клавише мыши вызываем контекстное меню работы с БД и выбираем действие (**ВОЛШЕБСТВО!!**) **Open with → Unity Editor**:

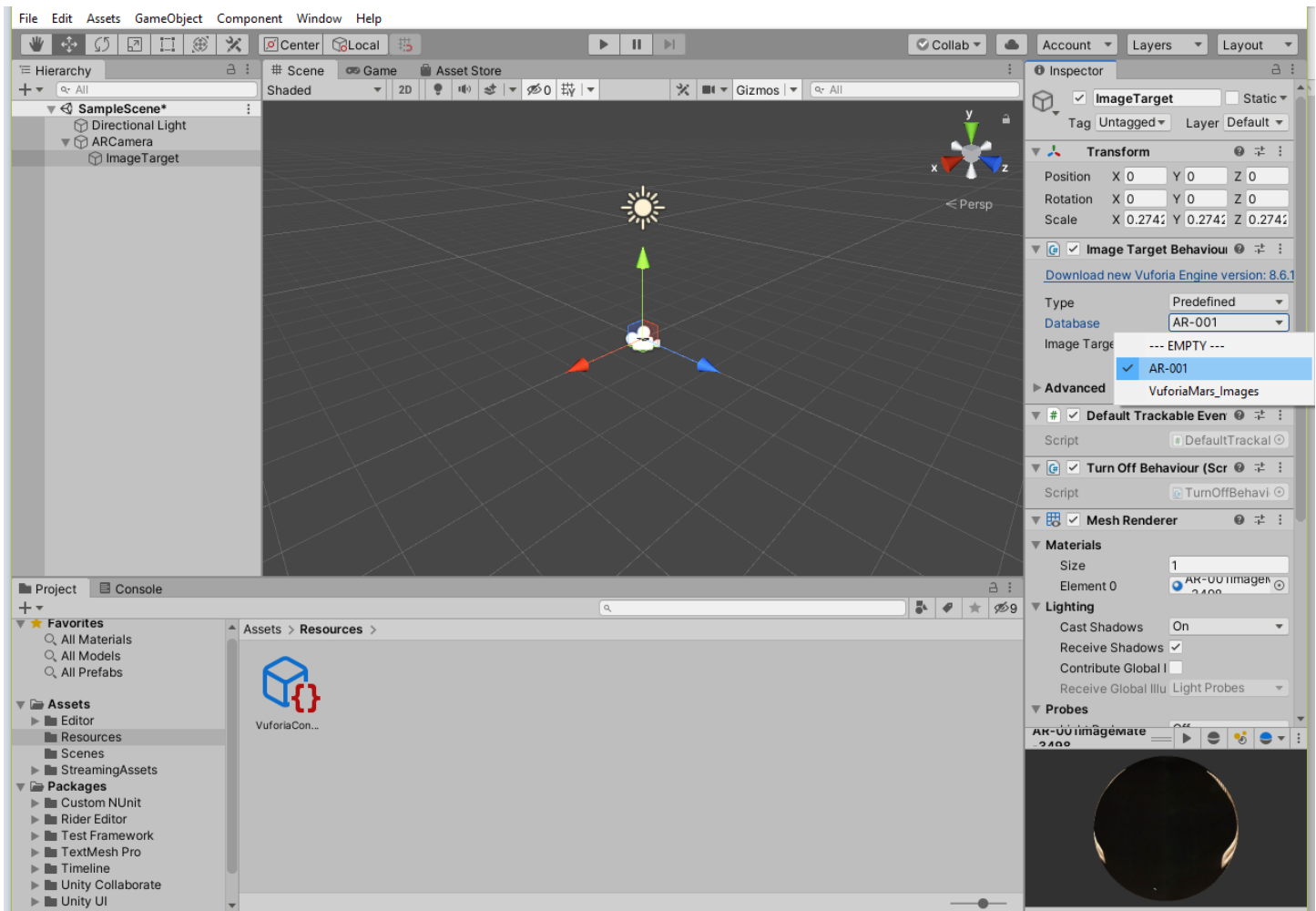


В результате в окне среды **Unity 3D** появляется диалоговое окно импорта БД таргетов:

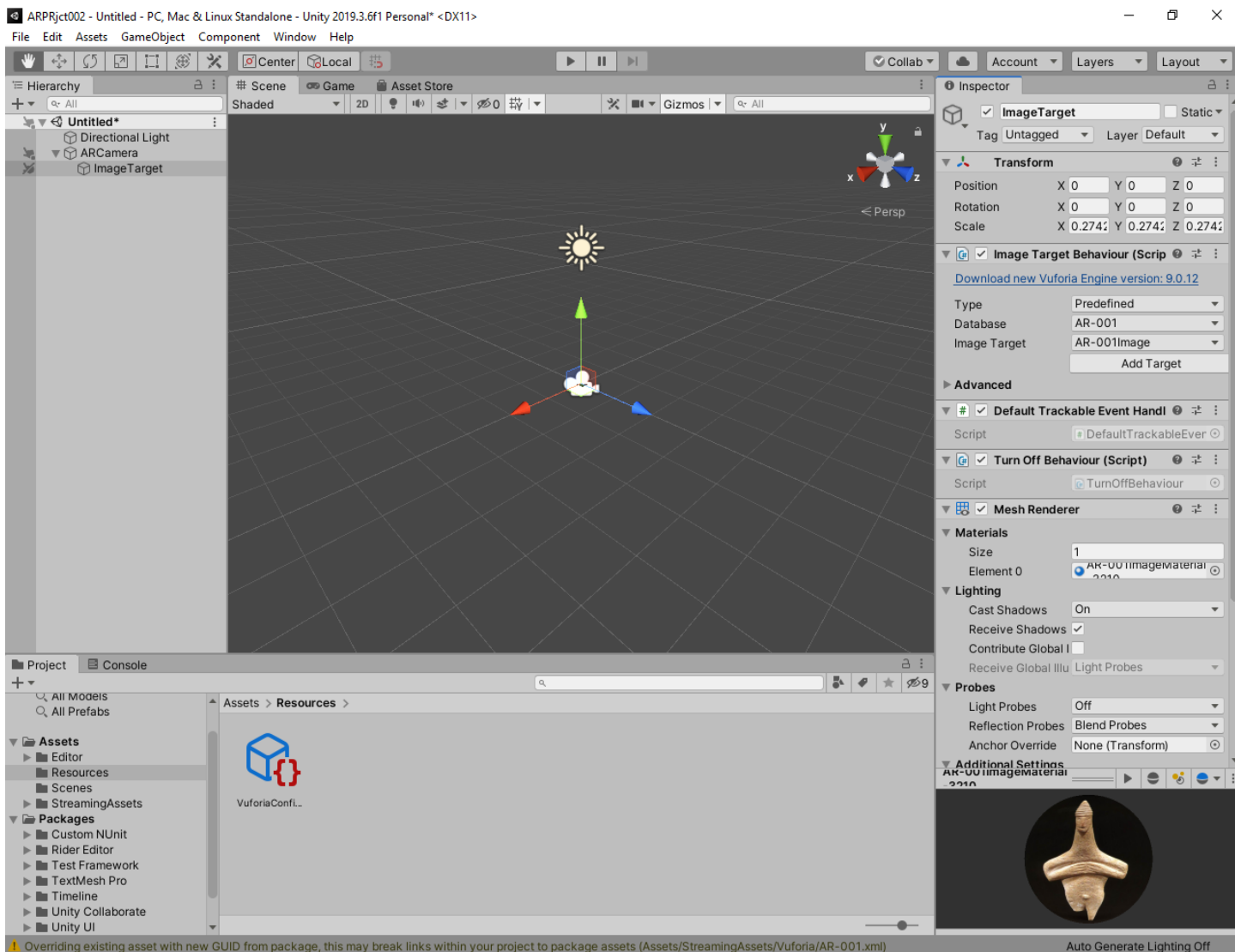


Выполняем импорт нашей БД таргетов проекта **AR-001** путем нажатия клавиши **Import**.

Проверить результат данного действия можно, выбрав в окне **Hierarchy** позицию **ImageTarget**. Далее, в проявившемся окне **Inspector** можно убедиться в наличии БД таргетов проекта **AR-001**, выбрав выпадающий список в области **Image Target Behaviour** → **Database**:



Из БД таргетов **AR-001** выбираем нужный нам **Image Target**, для данной ЛР это **AR-001Image.jpg**.



Теперь в поле сцены имеются два интересующих нас объекта **Vuforia**:

- **AR Camera** и
- **Image Target**.

Нам необходимо добиться правильного взаимного расположения камеры и таргета. Контролировать правильность выполнения этого действия можно в окне предварительного просмотра (выбрать **AR Camera** в области **Hierarchy**) в нижней правой части области **Scene View** – окно **Camera Preview**. Используйте ручки **Transform (Position, Rotation, Scale)** в области **Inspector** или аппарат **Gizmo** или **Toolbar** как для объекта **AR Camera** так и для объекта **Image Target** (выбираются в **Hierarchy**). Хорошим результатом является тот, когда изображение **Image Target** занимает от 30% до 60% окна **Camera Preview**. Добейтесь этого.

Ниже приведен вариант взаимного расположения камеры и таргета, когда **ARCamera** смотрит фронтально на таргет, удаленный на расстояние «1» от камеры. Обратите внимание на значения **Transform** в области **Inspector** для обоих объектов (**ARCamera** и **Image target**).

The screenshot displays the Unity 2019.3.6f1 Personal* interface. The main scene view shows a 3D environment with a sun, a camera, and an AR camera. The Hierarchy panel on the left shows the scene's structure, including a Directional Light, ARCamera, and ImageTarget. The Inspector panel on the right shows the properties of the selected ImageTarget, including its Transform, Image Target Behaviour (Script), Default Trackable Event Handl, Turn Off Behaviour (Script), and Mesh Renderer. The Project panel at the bottom left shows the Assets folder structure, including Resources and Packages. The Console panel at the bottom right shows a warning message: "Overriding existing asset with new GUID from package, this may break links within your project to package assets (Assets/StreamingAssets/Vuforia/AR-001.xml)".

Hierarchy

- Untitled*
- Directional Light
- ARCamera
- ImageTarget

Inspector

ImageTarget

Tag: Untagged, Layer: Default

Transform

Position	X: 0	Y: 0	Z: 1
Rotation	X: -90	Y: 0	Z: 0
Scale	X: 1.1663	Y: 1.1663	Z: 1.1663

Image Target Behaviour (Script)

Download new Vuforia Engine version: 9.0.12

Type: Predefined

Database: AR-001

Image Target: AR-001Image

Add Target

Advanced

Default Trackable Event Handl

Script: DefaultTrackableEver

Turn Off Behaviour (Script)

Script: TurnOffBehaviour

Mesh Renderer

Materials

Size: 1

Element 0: AR-001ImageMaterial

Lighting

Cast Shadows: On

Receive Shadows:

Contribute Global Illumination:

Receive Global Illumination: Light Probes

Probes

Light Probes: Off

Reflection Probes: Blend Probes

Anchor Override: None (Transform)

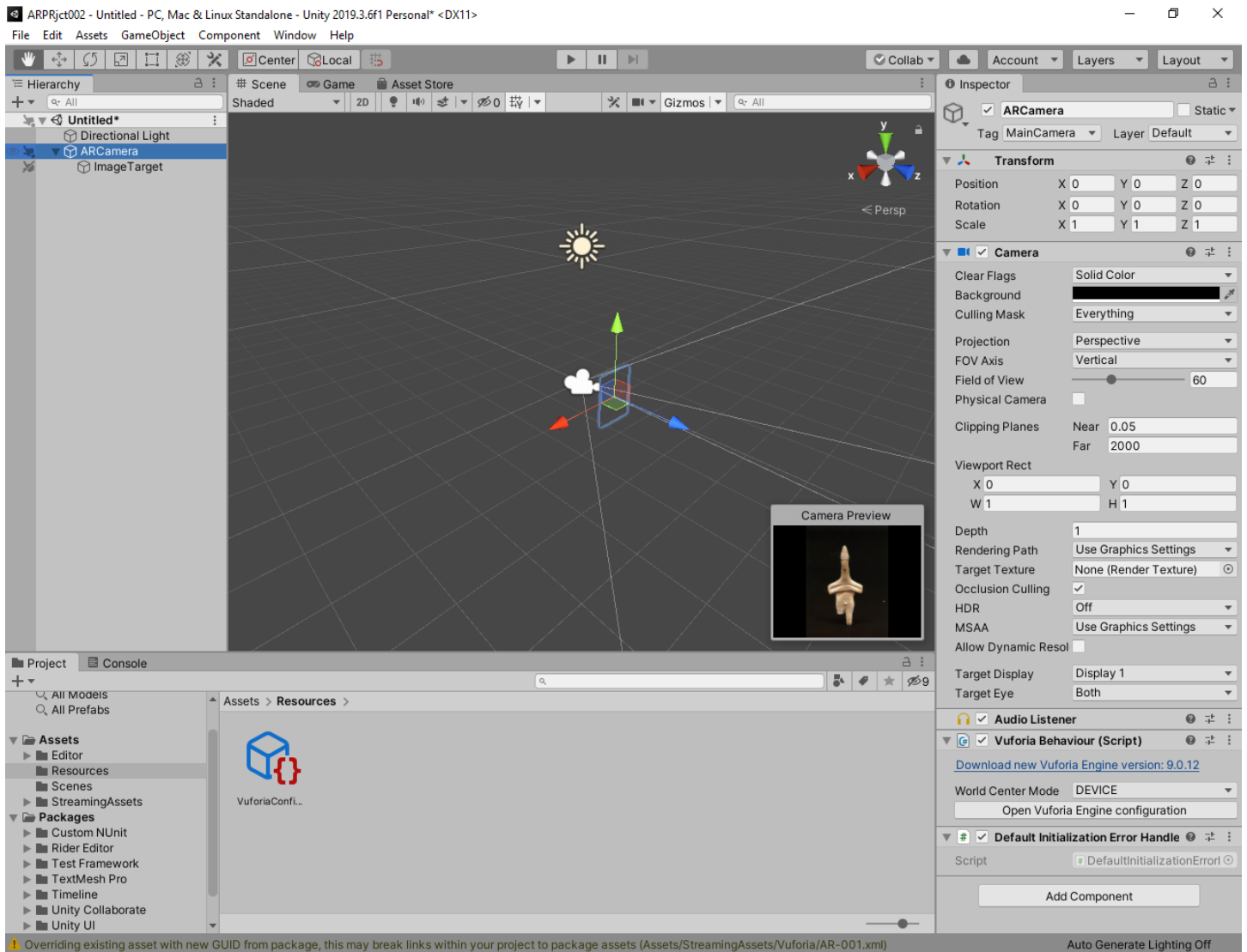
Additional Settings

AR-001ImageMaterial

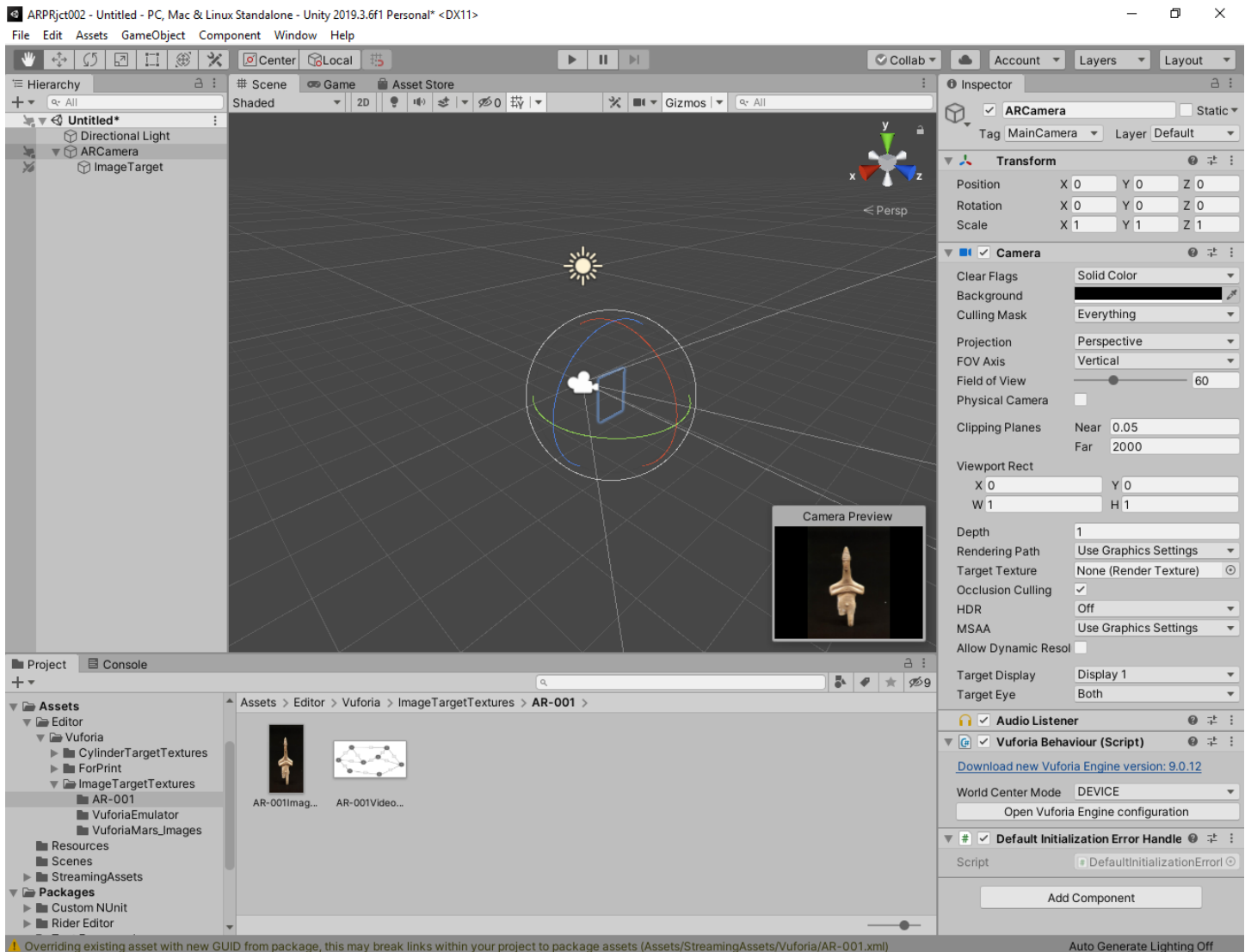
Auto Generate Lighting Off

Overriding existing asset with new GUID from package, this may break links within your project to package assets (Assets/StreamingAssets/Vuforia/AR-001.xml)

Auto Generate Lighting Off



По мере выполнения всех этих действий следите за изменением наполнения области **Project** редактора **Unity 3D** и содержанием появляющихся там папок, в том числе и тех, которые имеют отношение к нашему проекту:



Теперь надо разместить контент, связанный с меткой, в сцене ДР. В нашем случае метка (таргет), объект **Image Target**, уже размещен в сцене.

6. Теперь необходимо загрузить тот плоский объект (2D-изображение), который должен появиться на экране устройства поверх изображения транслируемой реальности на месте, определяемом таргетом.

2D-Изображение (контент) в данной ЛР – это заранее подготовленный файл - **AR-001ContentImg.jpg**. Находится в файловой структуре на локальной машине.

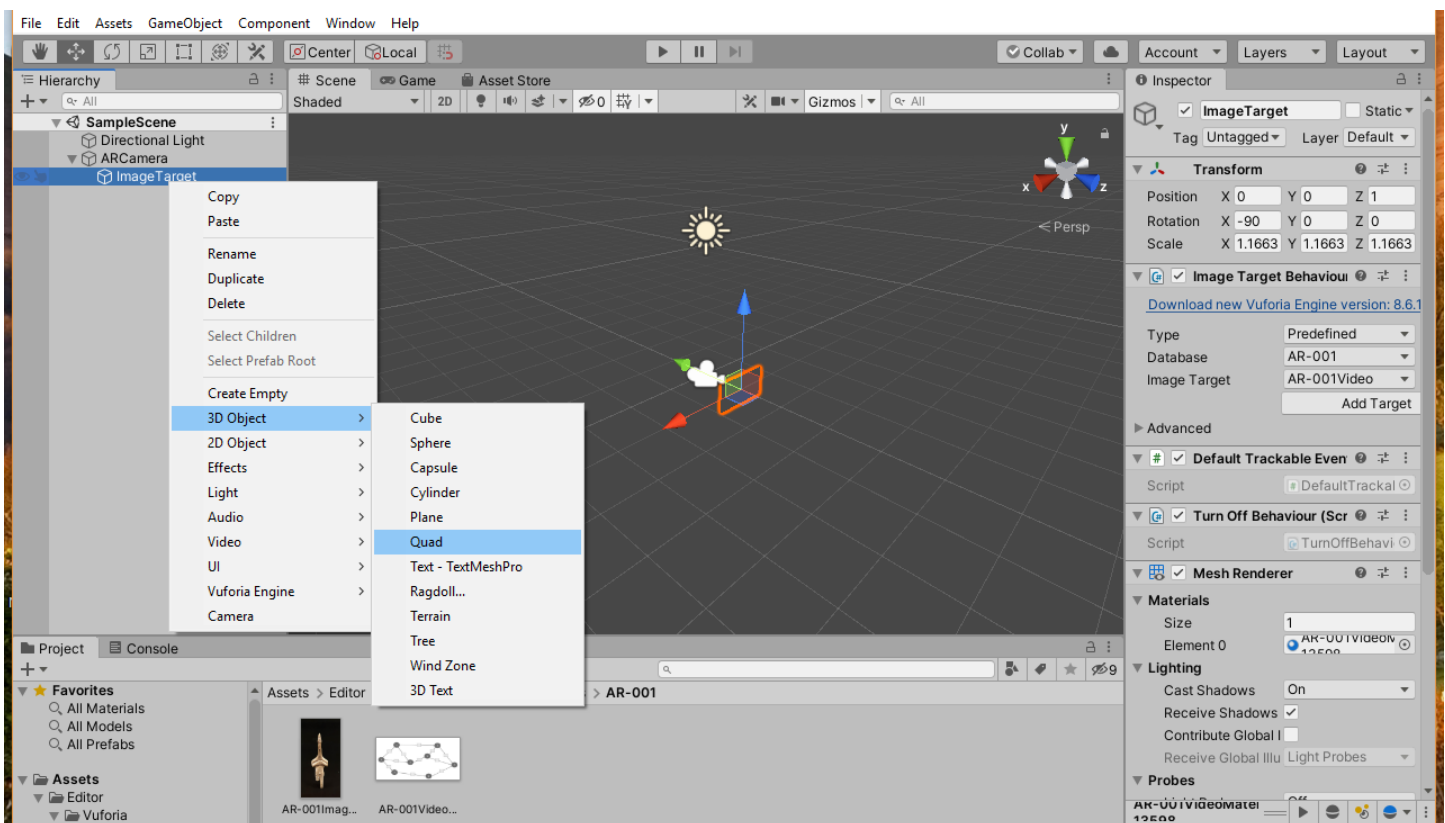
Для размещения контента в **Unity 3D** воспользуемся функционалом **Unity 3D**. Для этого в системе существует большое количество собственных объектов. В частности, эти объекты можно увидеть в основной панели меню **Unity 3D** в группе **Game Object**. Один из них, чаще всего используемый для размещения 2D-контента в трехмерном пространстве, это – **Quad**.

Т.о. в **Unity 3D** связываем **Image Target** с размещаемым контентом через объект (шаблон, контейнер) **Quad**. Для такого связывания в иерархии (**Hierarchy**) **Quad** должен располагаться под **Image Target**.

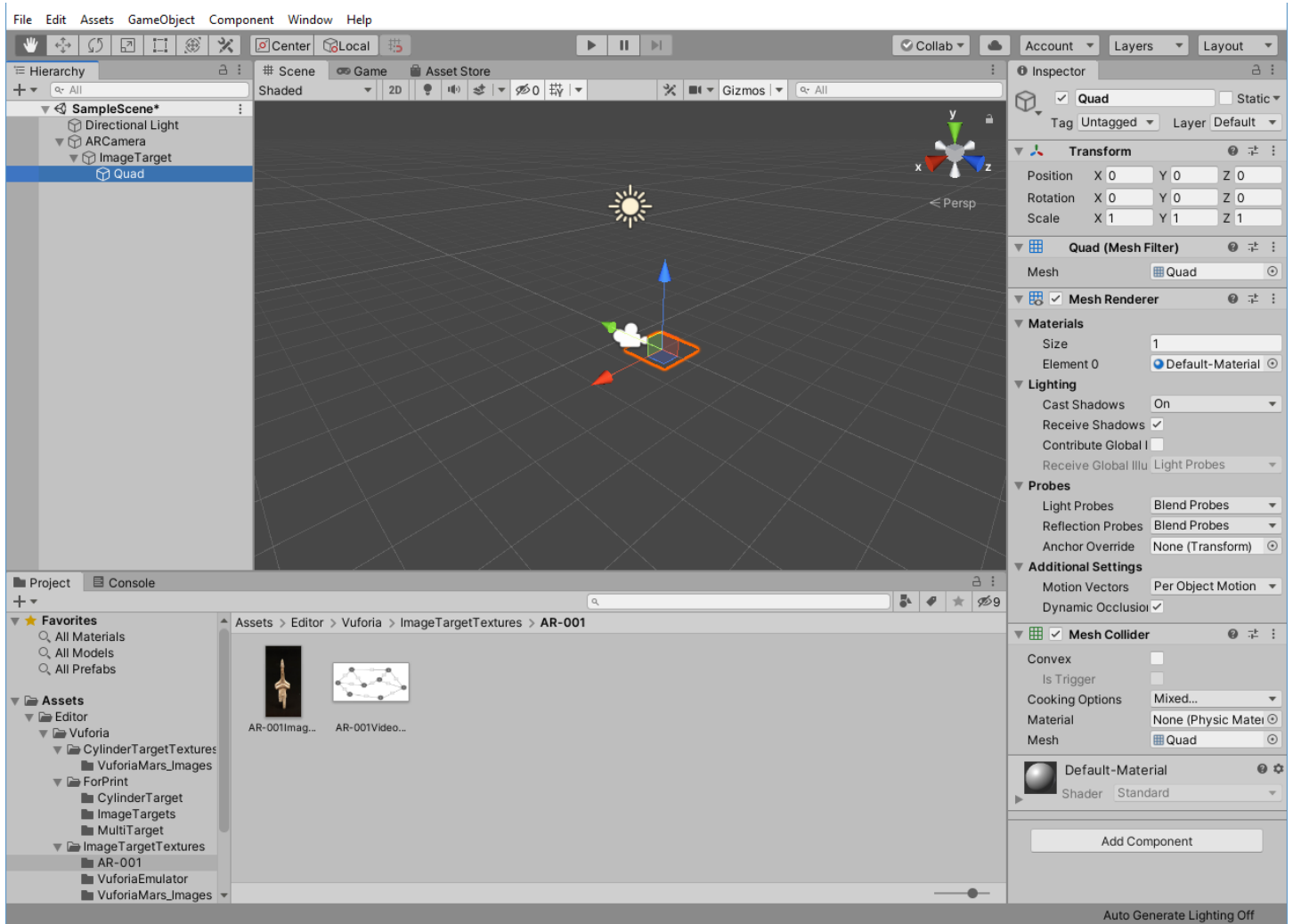
Последовательность операций для получения результата:

- В **Hierarchy** выбираем **Image Target**;
- По правой клавише мыши в выпадающем меню находим строчку **3D Object**;
- В связанном с ней списке альтернатив выбираем **Quad**.

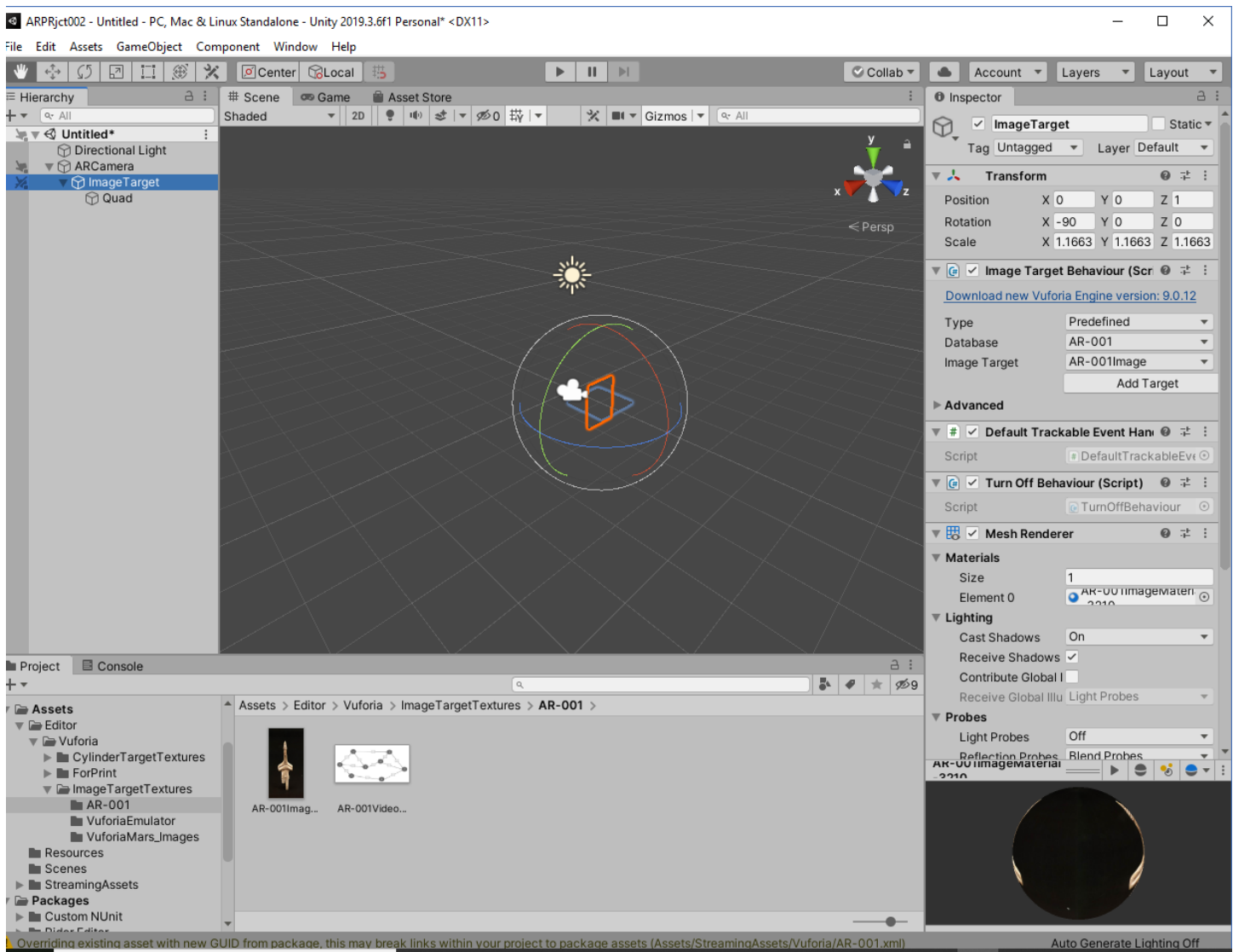
Unity 3D: 3D Object → Quad, контейнер 2D-объекта:



В результате в сцене появляется новый объект **Quad**, а в **Inspector'e**– информация о нем:

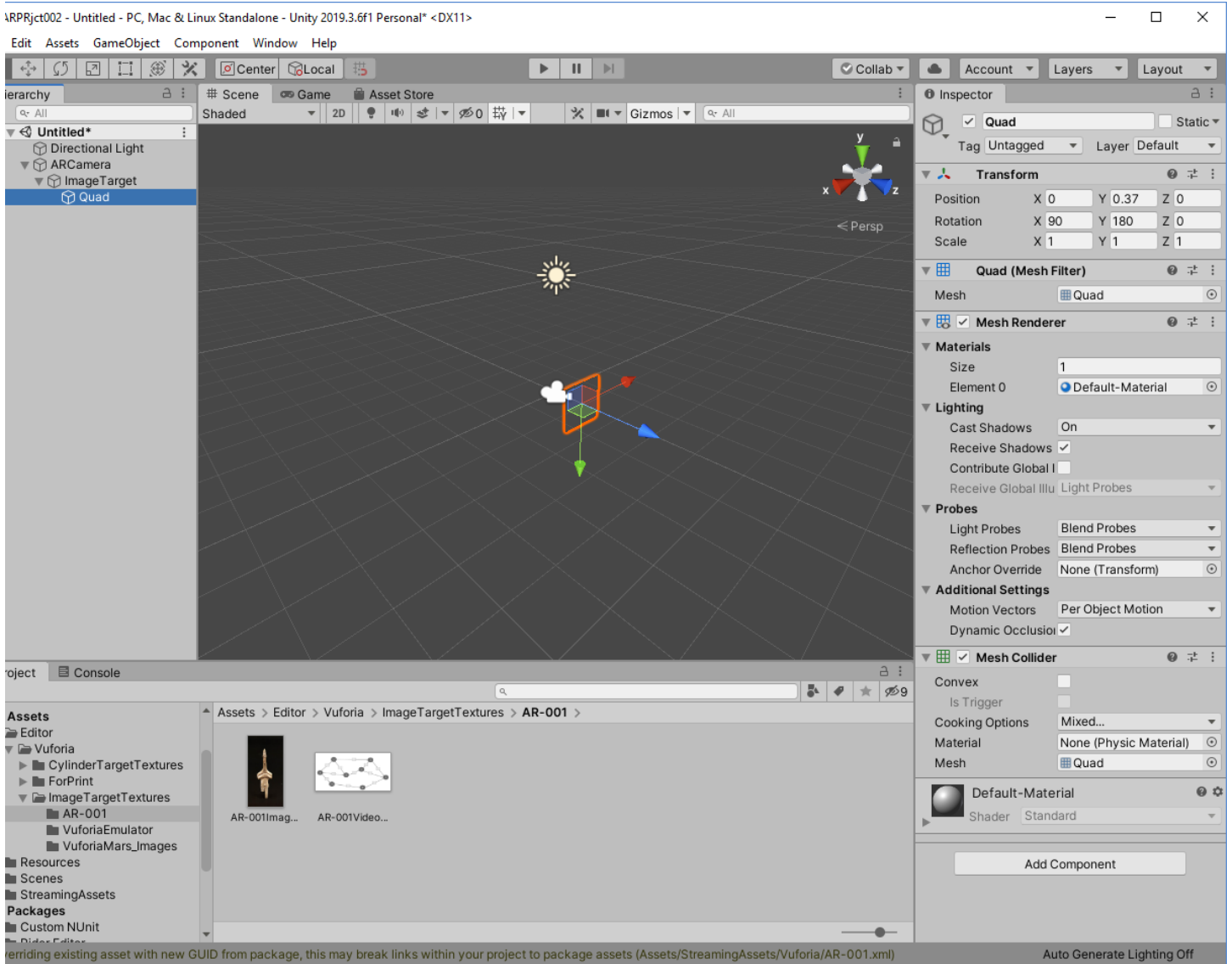


Исходное взаимное расположение таргета (выбран в **Hierarchy**) и контейнера для видеоклипа видно из нижеприведенного вида сцены:

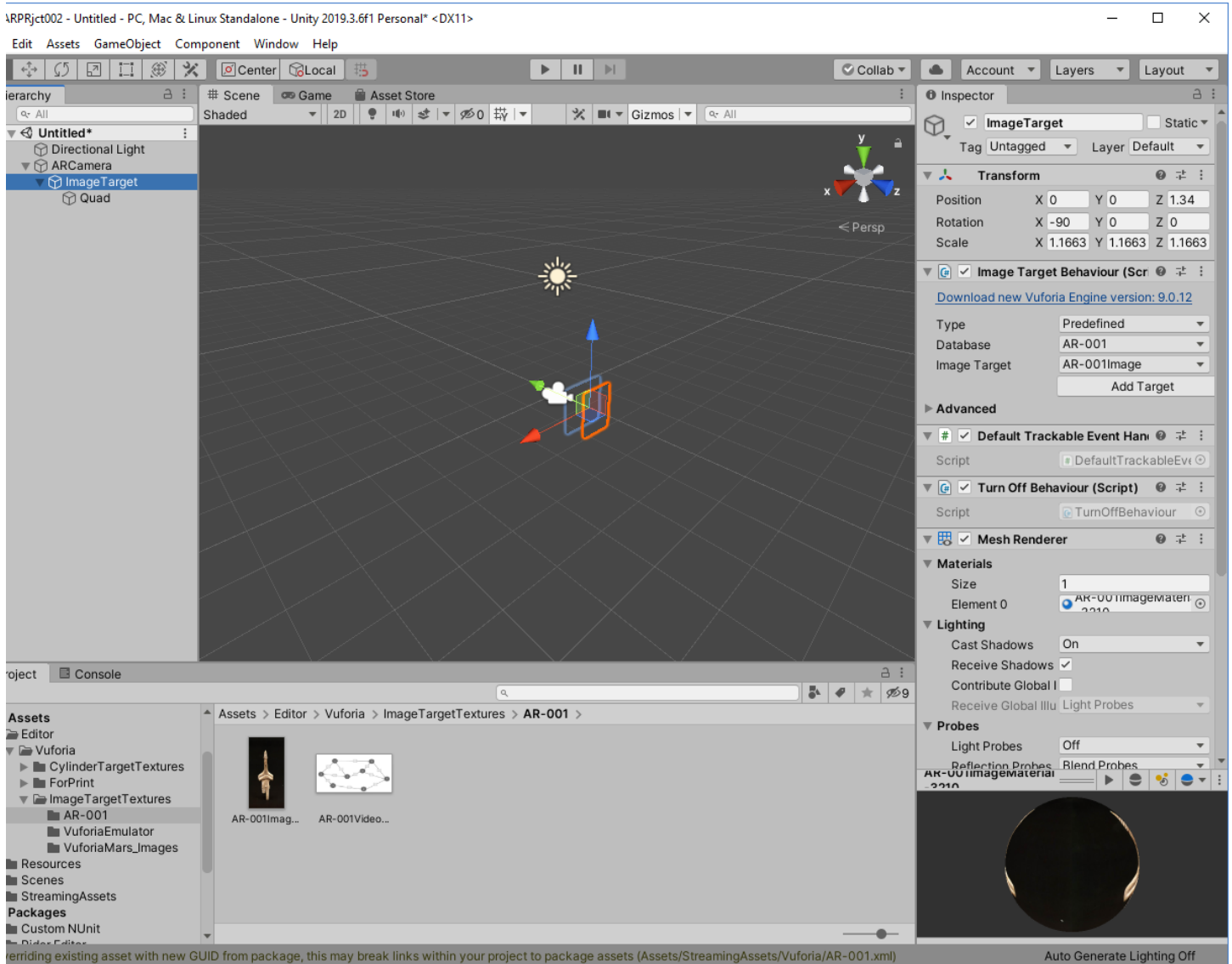


Не меняя расположения камеры и таргета, найдем наиболее удобное положение для контейнера 2D-Изображения – **Quad'a**. Например, экран с изображением (**Quad** с изображением) будет расположен на таком же расстоянии от камеры, что и таргет, но с небольшим разворотом по вертикали (ось «Y»). Вы можете выбрать расположение самостоятельно (**Toolbar** или **Transform** в **Inspector'e**).

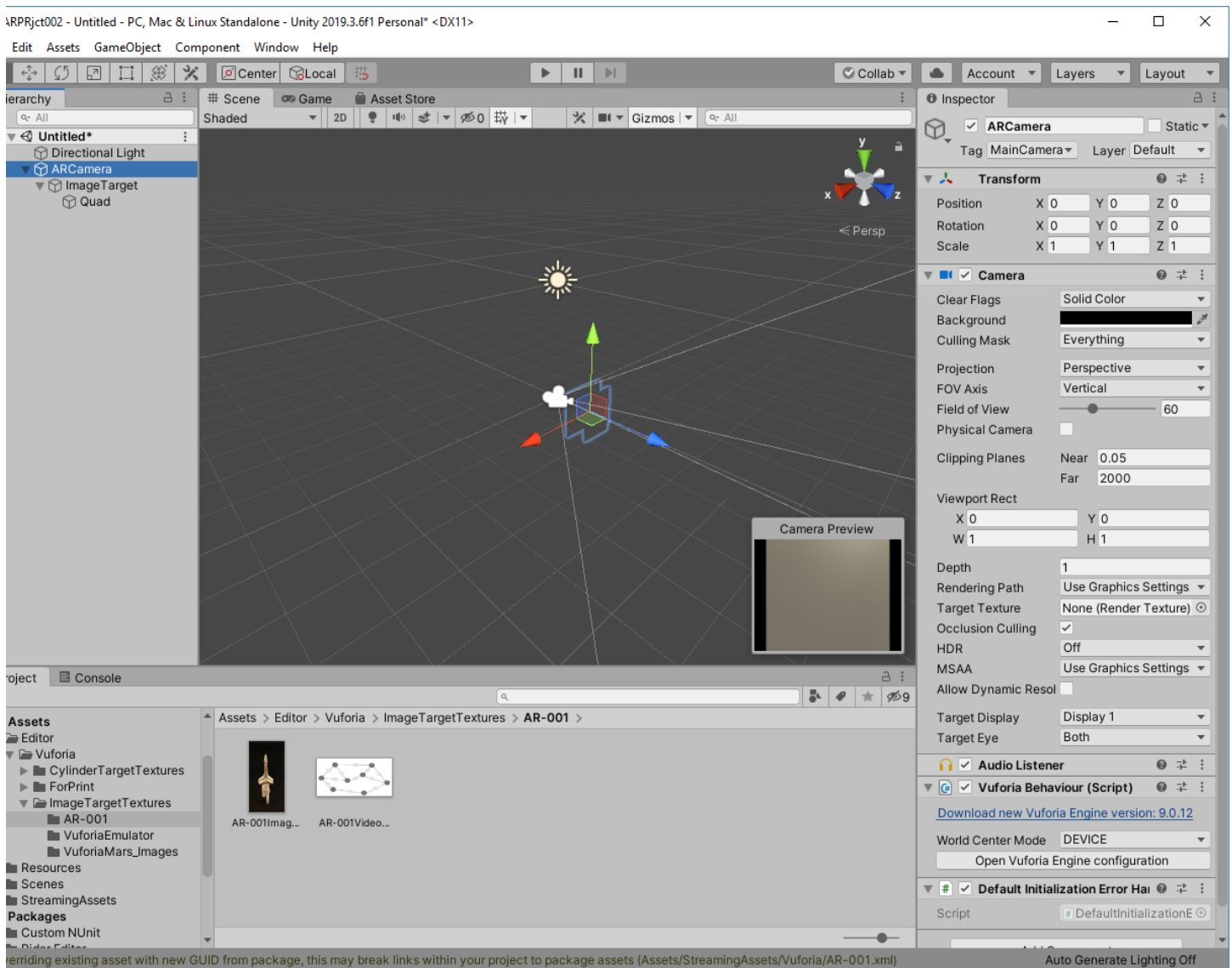
- **Quad:**



- **Image Target и Quad:**

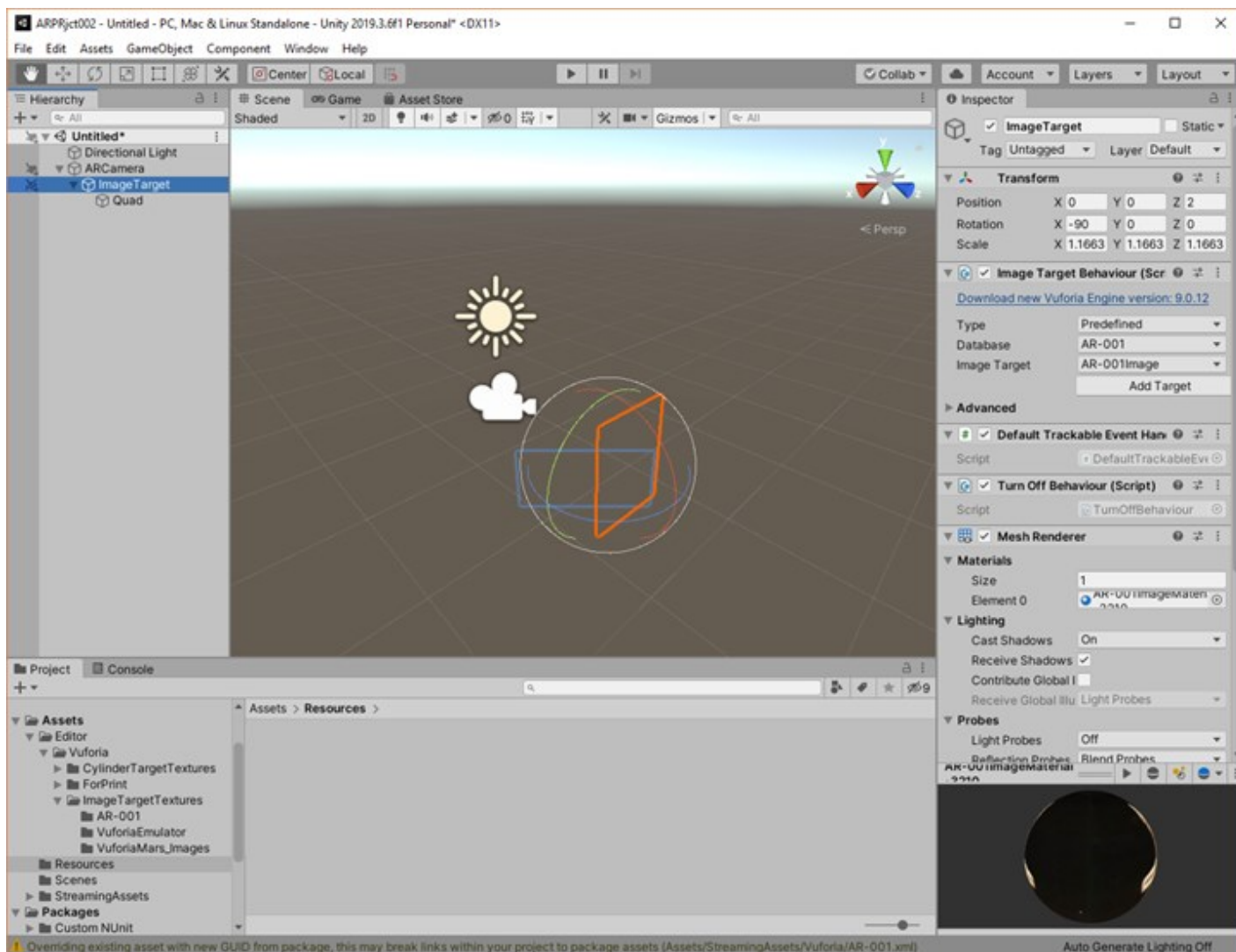


- То, что видит **ARCamera**:



То, что вы видите означает, что **Quad** расположен перпендикулярно оси камеры, в поле зрения камеры, но ближе к ней, чем таргет.

Добейтесь описанного выше расположения квадрата и таргета: **Quad** для изображения расположен на таком же расстоянии от камеры, что и таргет, но с небольшим разворотом по вертикали (ось «Y») и сдвинут в сторону, но в поле зрения камеры (см. ниже):



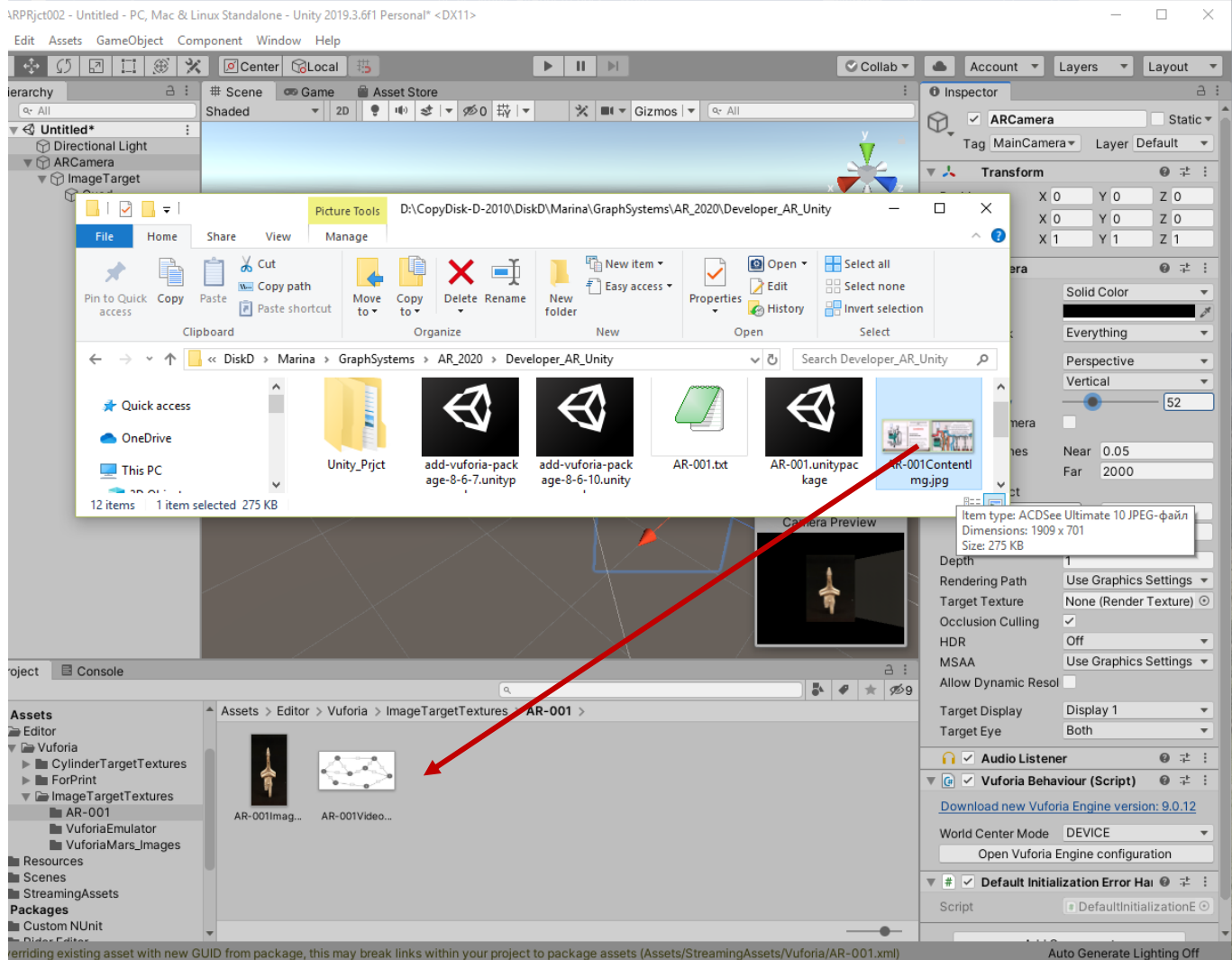
Таким образом, в сцене размещены **ARCamera**, таргет и контейнер для 2D-контента: Изображения.

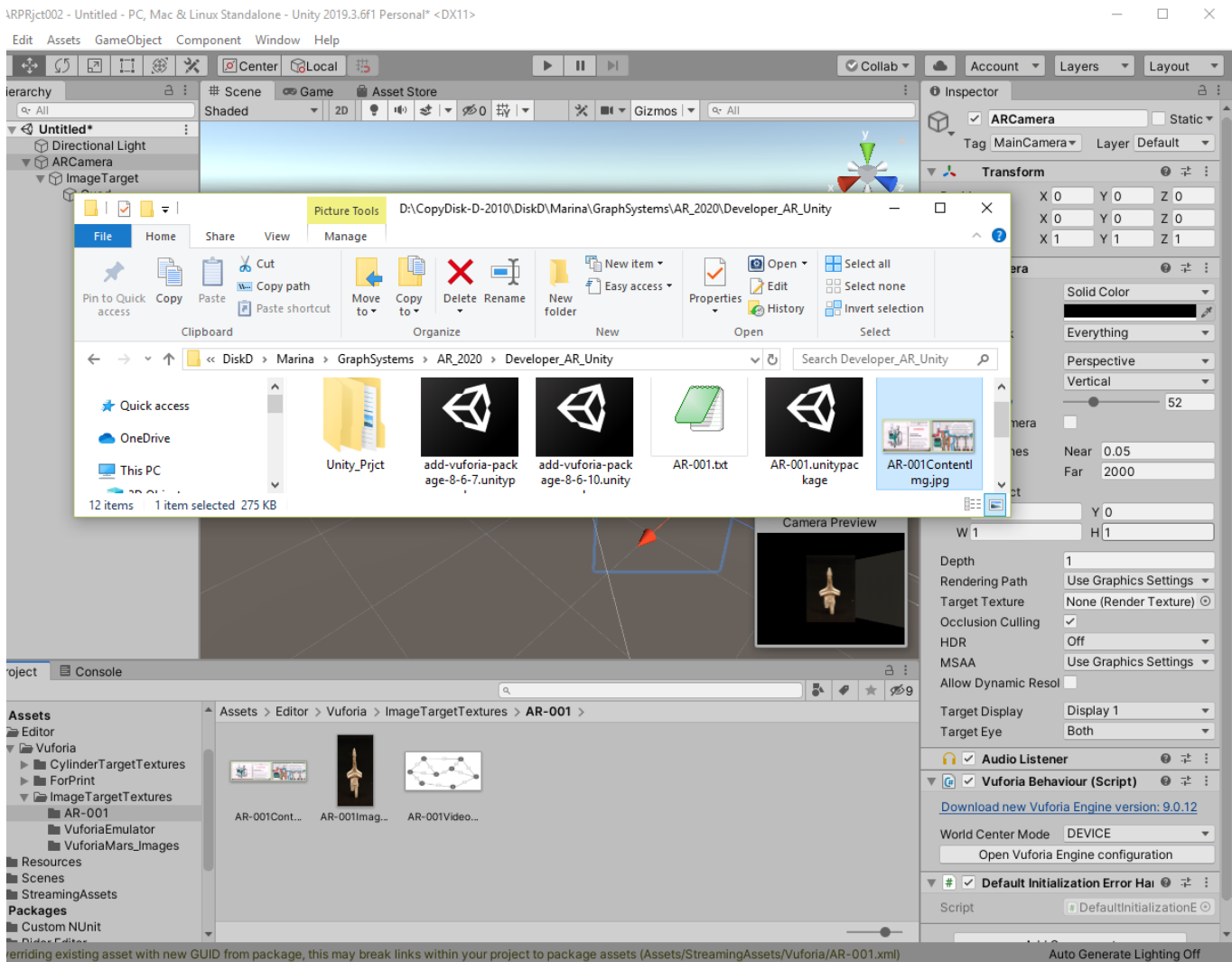
Переходим к подготовке контента для помещения его в контейнер Quad.

Контент в данном случае это визуализируемый объект, помещаемый в контейнер **Quad**. В данной ЛР в качестве визуализируемого объекта будем использовать **плоское изображение** в одном из допустимых для **Unity 3D** форматов - **.jpg**.



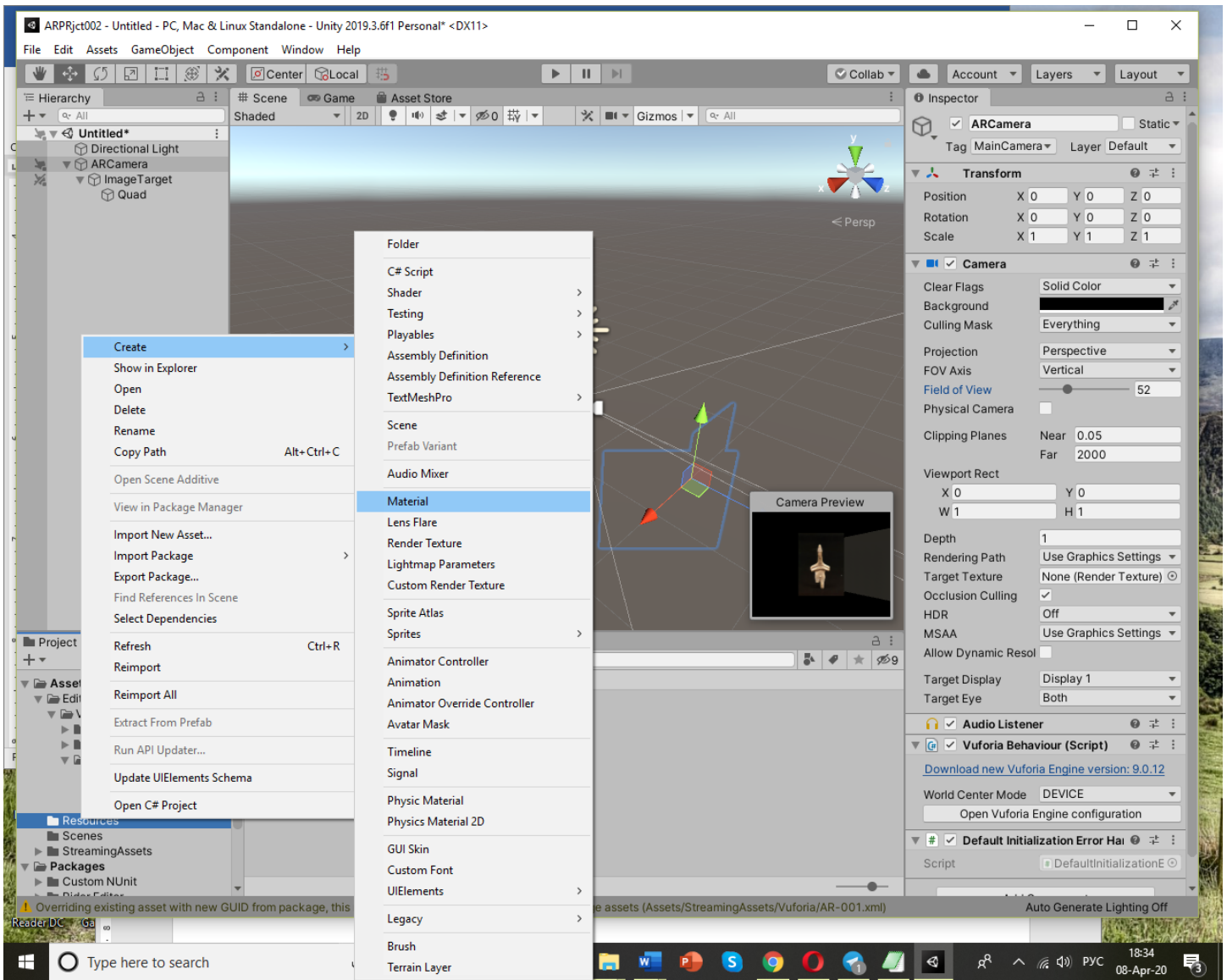
Загрузим наше **2D Изображение** (у нас – это **AR-001ContentImg.jpg**) в поле **Assets-Resources** из локальной файловой структуры с помощью операции **Drag-n-Drop**.



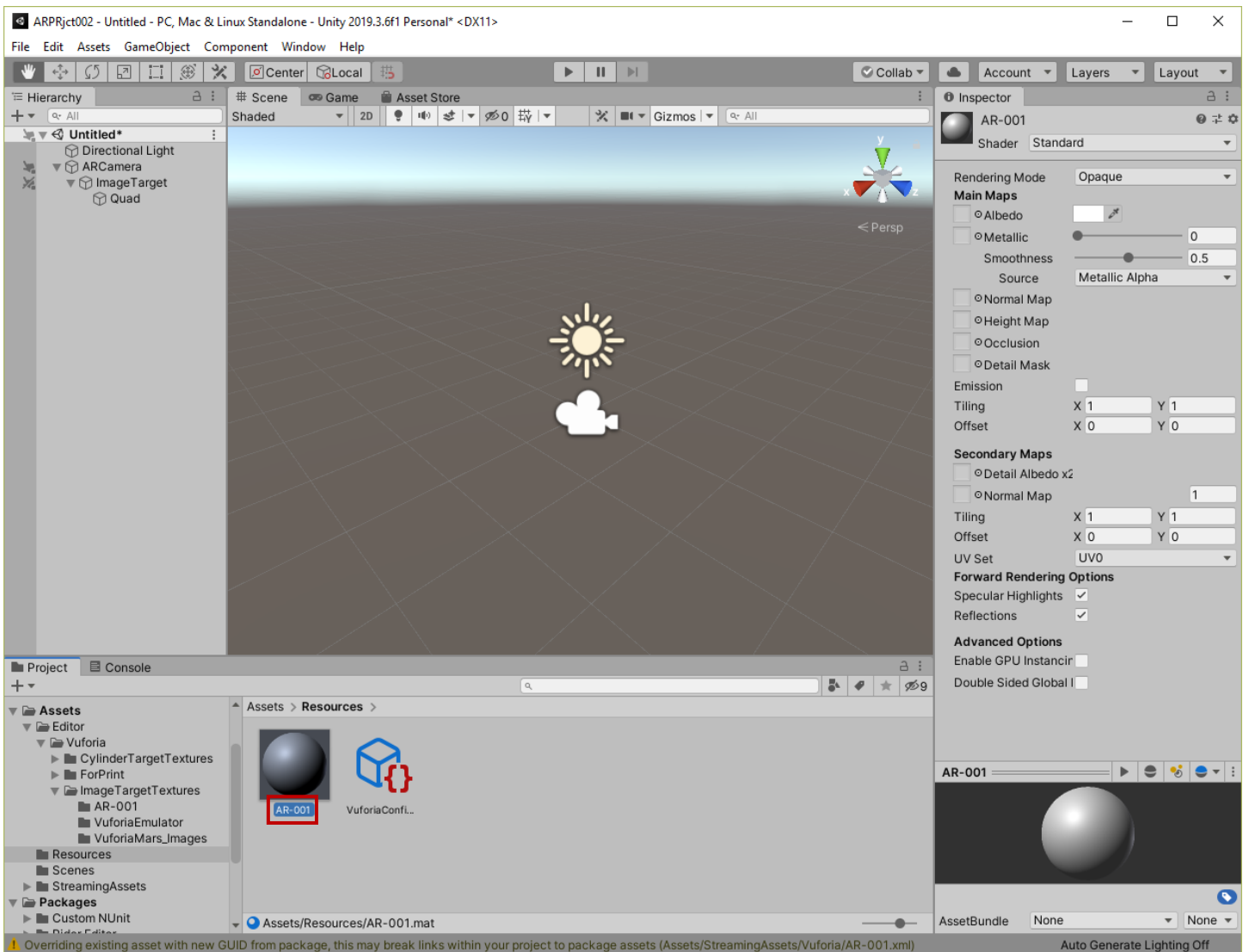


В отличие от первой части ЛР № 2 (замещающий виртуальный объект – видеоклип), при заполнении контейнера **Quad** конкретным значением из файла **.jpg**, необходимо использовать особый тип **asset'a** (актива) **Unity 3D**, который называется **Material** («материал»). Именно он будет являться контентом в данном случае.

В поле **Project** редактора **Unity 3D** выбираем папку **Resources**, переходим в поле **Assets-Resources**, в пустом пространстве кликаем правой клавишей мыши и «создаем» (элемент меню **Create**) новый **Material**:

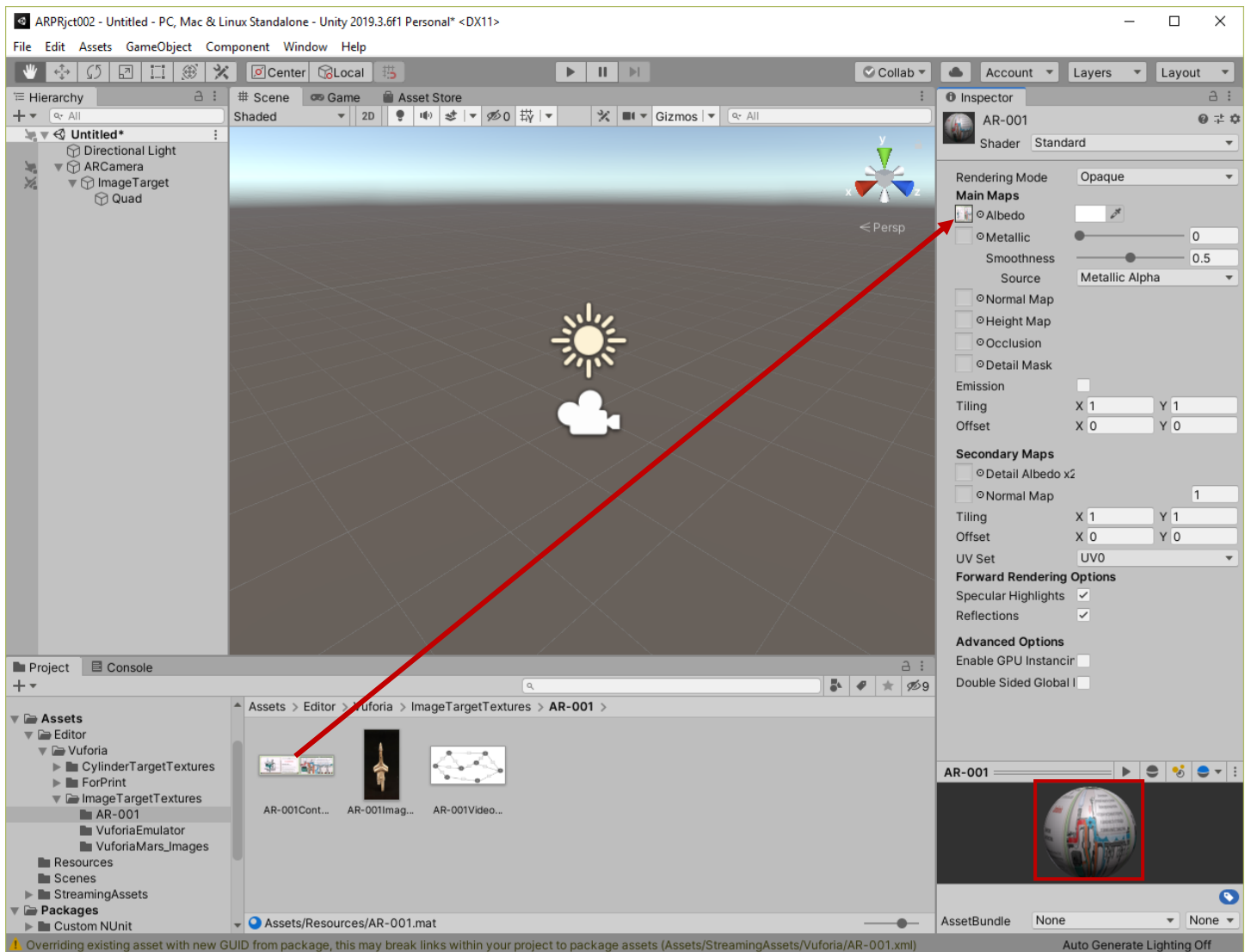


В результате в **Resources** будет добавлен новый **Material**, которому в поле **Assets** – **Resources**, под его иконкой, присвоим уникальное имя, например – **AR-001** (или **AR-002**) вместо **New Material**:



При этом в поле **Inspector** отображается вся информация, связанная с данным «материалом». Воспользуемся полем **Inspector**, для того, чтобы **ассоциировать** «материал» с 2D-изображением, хранящимся в файле **.jpg**, т.е. получить требуемый в данном случае контент.

Для этого начинаем работать с основными параметрами материала, которые отображены в **Inspector**'е в различных полях. В нашем случае, когда «материал» - это всего лишь изображение, нам важен только один его параметр (см. документацию на **Unity 3D**) - **отражательная способность материала**. Этому параметру соответствует поле **Albedo**. С помощью операции **Drag-n-Drop** помещаем в это поле ассет с файлом **AR-001ContentImg.jpg**.

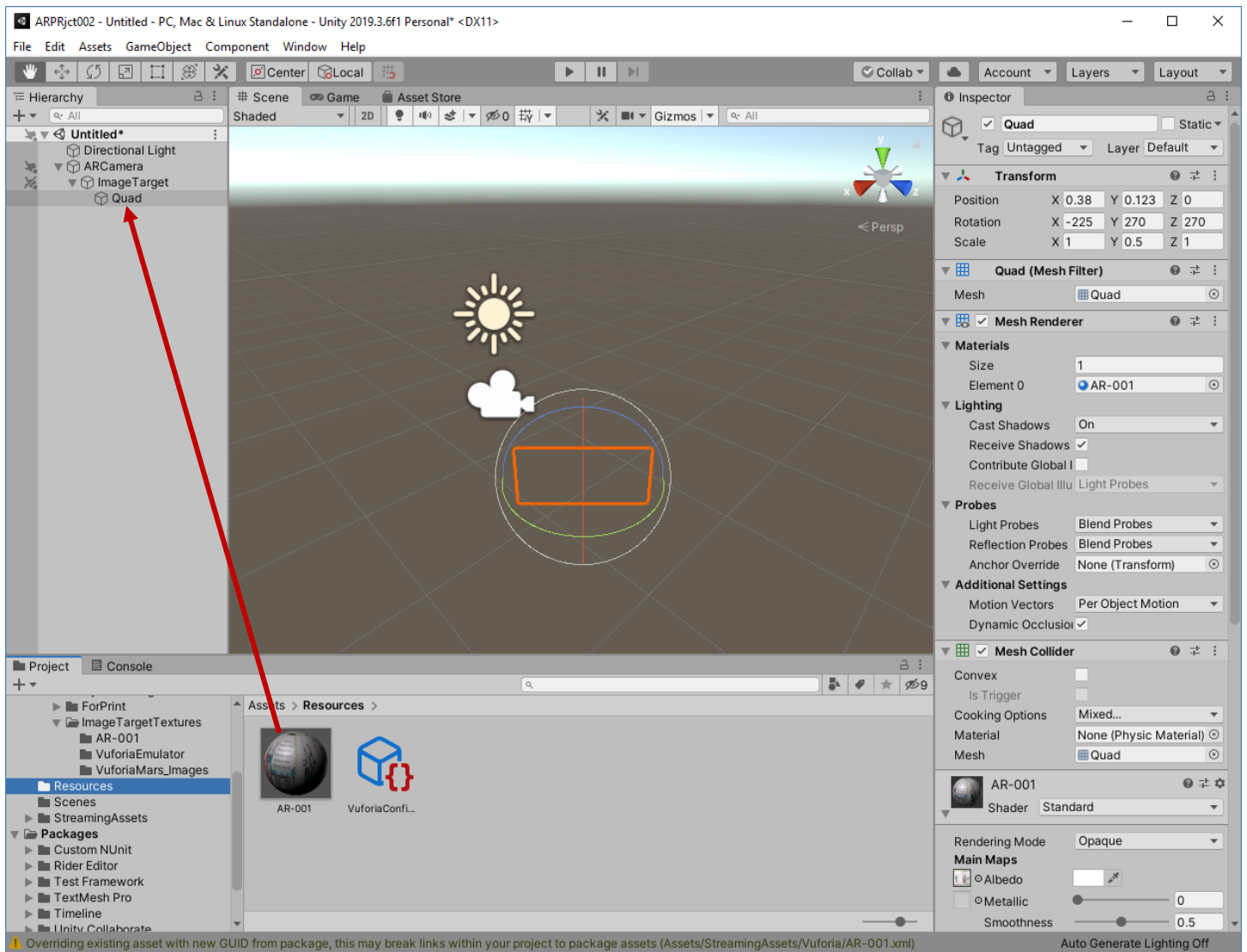


В результате в нижней части **Inspector'a** появляется интерактивный индикатор (Шарик), который свидетельствует об успешности ассоциирования ассета «материал» с ассетом «изображение».

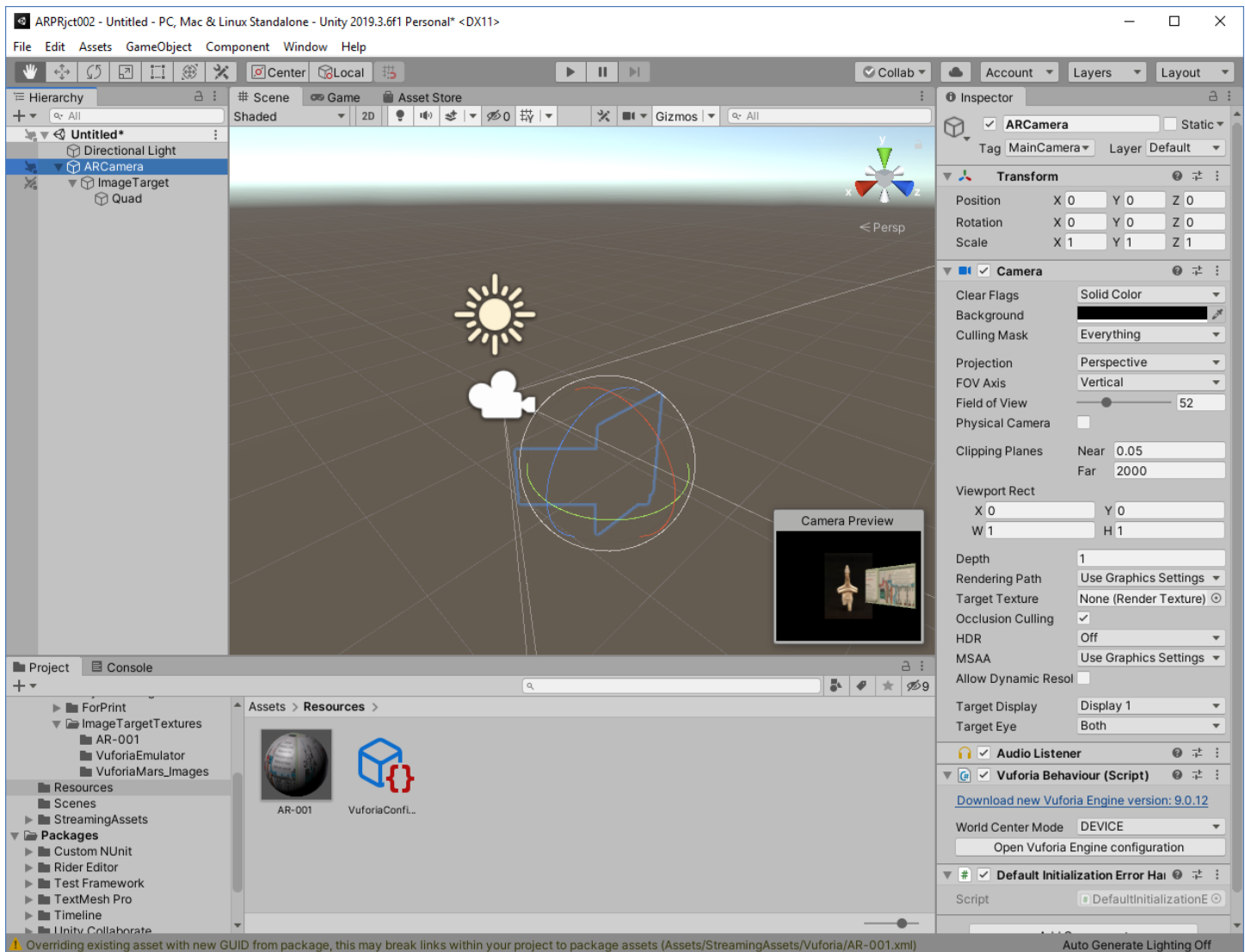
Переходим к загрузке контента (файл .jpg) в контейнер (Quad).

На данном шаге мы имеем подготовленный контент, представляющий из себя **Material**, содержащий требуемое **2D-Изображение**, и контейнер – объект **Quad**. Для загрузки контента в контейнер переместим с помощью операции **Drag-n-Drop** наш **Material** по имени **AR-001**, находящийся в поле **Project→Assets→Resurces**, в **Quad**, расположенный в поле **Hierarchy**.

ВАЖНО!! Для лучшего восприятия изображения позаботьтесь о том, чтобы соотношение сторон у **Quad** было таким же, как соотношение сторон у загружаемого изображения. В нашем случае:



В результате контейнер оказывается загружен нужным контентом, что отображается в поле **Preview ARCamera** в **Scene** редактора **Unity 3D**.



Разрабатываемое приложение должно будет использовать камеру конкретного **Android**-устройства. **Unity 3D** в нашем проекте только подготавливает универсальный драйвер **AR Camer**'ы для типового **Android**-устройства. Все особенности конкретных, передовых **Android**-устройств (стерео, 4K, и т.д.) требуют дополнительного программирования, что находится за пределами данной ЛР.

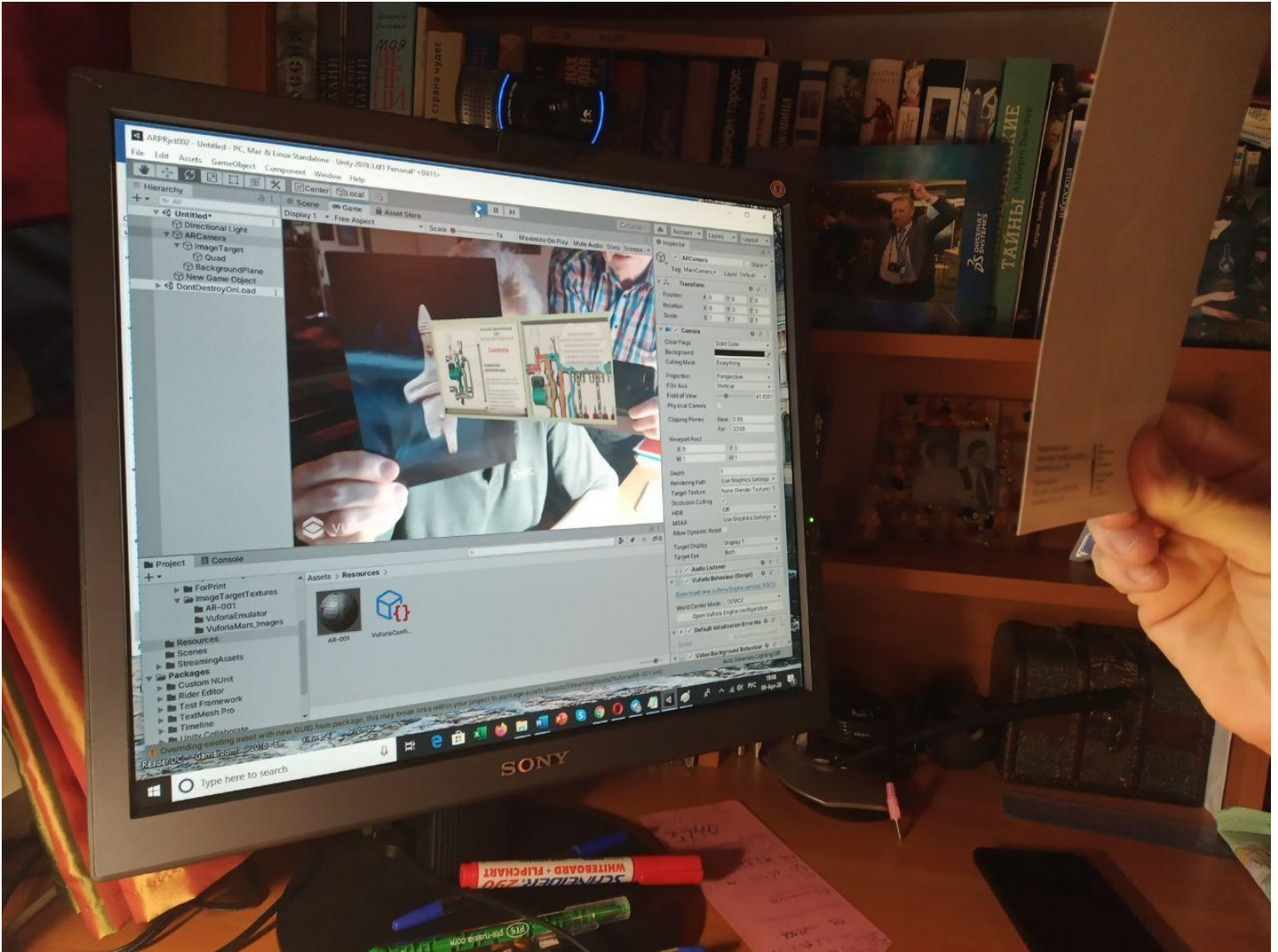
Таким образом, после проведенных манипуляций сцена нашего проекта содержит камеру ДР (**AR Camera**) для типового **Android**-устройства, метку (**Image Target**), контейнер (**Quad**) для 2D-контента, связанный с меткой.

Проверить работоспособность разрабатываемого Приложения ДР непосредственно в **Unity 3D** можно на локальной машине, если она снабжена видеокамерой.

Для этого в окне **Unity 3D** необходимо войти в режим **Play** – найти и нажать на клавишу проигрывания («**Play**») в верхней части окна, как показано на рисунке ниже:

Тест: в область просмотра камеры локальной машины помещается твердая копия выбранного для данной сцены таргета:

Тест: по выбранному таргету автоматически прикрепляется объект дополненной реальности. В данном случае – это Картинка:



Разработка сцены закончена. Можно сохранить полученный результат – сцену - не выходя из **Unity 3D**. Это может пригодиться в дальнейшем при доработке сцены, при прерывании сеанса работы в **Unity 3D** и т.д. Для сохранения сцены выполнить: **File→Save** или **File→Save as**. Сохранение производится в локальной ФС на вашей локальной машине.

7. Создание файла **.apk** для загрузки Приложения ДР на **Android-MY**.

Все установки, настройки и манипуляции с окнами и функциями **Unity 3D** для создания файла **.apk** идентичны тем, что вы выполняли в ЛР№2, часть 1. Не забудьте снабдить свое загружаемое на МУ Приложение иконкой!

Сохраняем собранный файл **.apk** в локальной файловой системе разработчика, загружаем и устанавливаем его на **Android** - МУ любым известным вам способом.

Видео разработанного в данной ЛР Приложения ДР доступно по ссылке:

<https://youtu.be/ХРАкxZАЕТ98>

Разработанное в рамках ЛР № 2, часть 2 AR-Приложение необходимо продемонстрировать преподавателю. Для этого загрузите свой .арк на любой файлообменник или доступное облако и пришлите мне ссылку!

Не забудьте про таргет!