Лабораторная работа № 3. Проект – разработка простого AR-Приложения для Android-устройства (смартфон, планшет и пр.). Создание в графическом редакторе Unity 3D сцены дополненной реальности: визуализация управляемой анимации 3D-Модели с использованием виртуальной кнопки. Часть 1. Изучение возможностей по созданию анимации объектов контента в Проектах AR.

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

Введение.

Работа по созданию приложений ДР заключается в заведении проекта и объектов проекта 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 виртуальной **3D-модели** будет осуществляться через определяемую в **Vuforia** метку (**Target**). Допустимые в **Vuforia** типы таргетов были подробно рассмотрены ранее. Требования к **Target** в данной ЛР будут определены ниже.

Следует обратить внимание, что 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 и дополнения сцены 2D-изображениями или 3D-моделями**.

Задание на ЛР – разработка элементов виртуального интерфейса для работы в ДР, т.е. установление связей между поведением (**behaviour**) виртуальной **3D-модели** (у нас - **«анимация»**) и состоянием виртуального элемента управления (**«виртуальная кнопка» → события:** кнопка отжата/кнопка нажата).

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 заранее подготовленную 3D – модель, которая будет анимирована. А управление анимацией будет осуществляться с помощью виртуальной кнопки. В первой части ЛР№ 3 студенты осваивают способы задания анимации объектов контента.

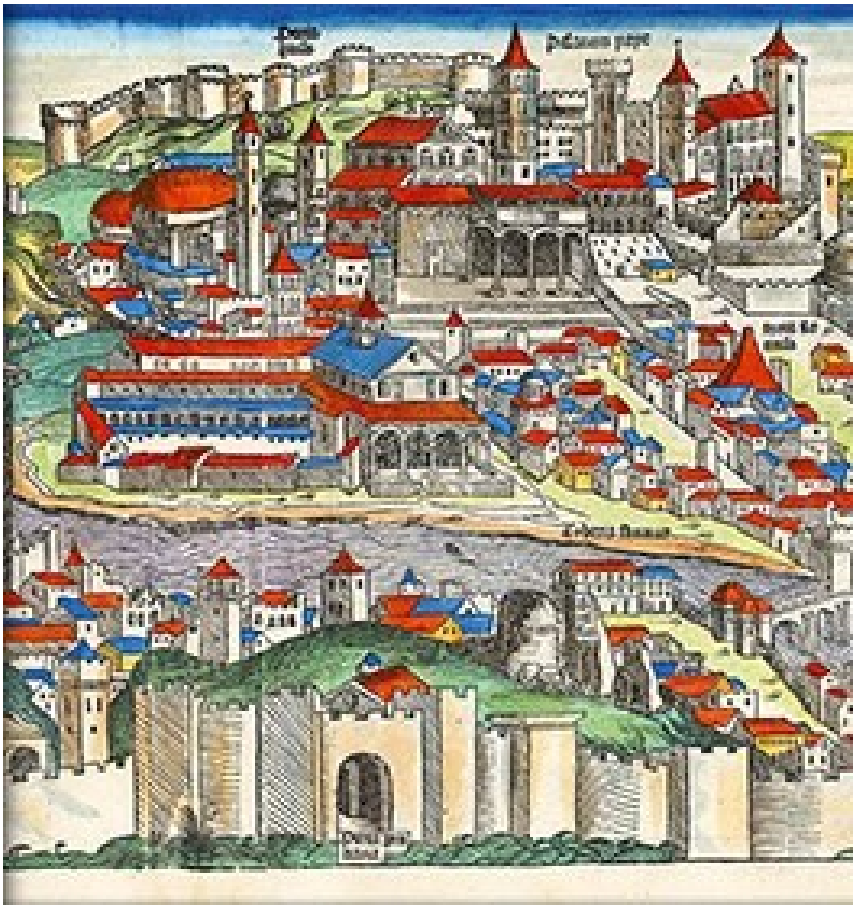
Предварительные условия для начала работы – см. ЛР №2.

Пункты 1, 2 выполнения ЛР: На сайте https://developer.vuforia.com/ осуществляем вход с заведенными ранее логином/паролем для получения **лицензионного кода** и заведения **таргета** (генерация меток). Все шаги и возможные проблемы/особенности выполнения этих шагов – см. **Описание ЛР№2**.

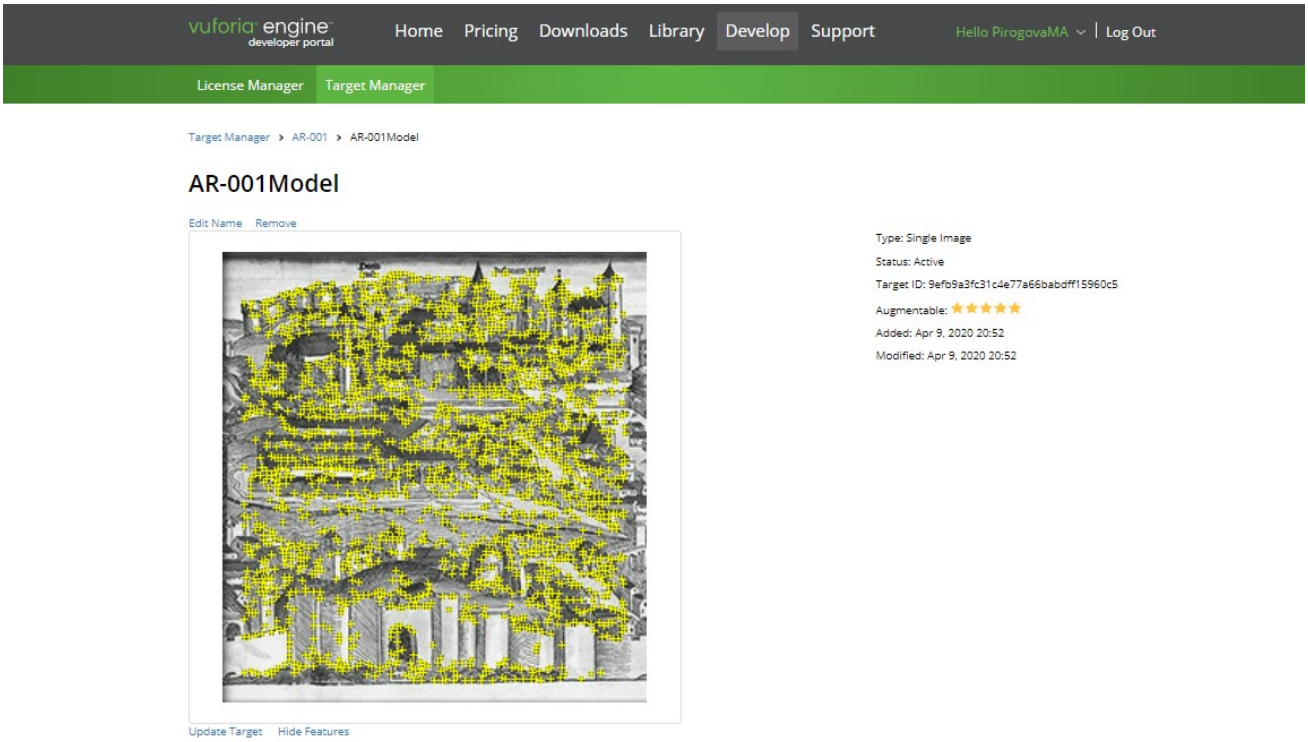
Виртуальная кнопка, которую мы собираемся использовать в данной ЛР, должна размещаться в пространстве, сканируемом **AR Camera** и быть привязанной к таргету.

При выполнении **Пункта 1** обратите внимание - для данной ЛР, где используется и виртуальный объект (**3D-Модель**) и виртуальная кнопка, существенно важно наличие «хорошего» таргета, количество «звездочек» у которого д.б. максимально, а точки распознавания на рисунке таргета («крестики») равномерно распределены по полю таргета. Это связано с тем, что таргет, помещенный перед камерой сканирующего устройства, должен быть достаточен не только для воспроизведения **3D-модели**, но и будет служить основой для размещения виртуального элемента управления – **кнопки**.

Предлагается в качестве таргета использовать уже известное вам изображение, которое было добавлено в Базу данных таргетов для ЛР №2, часть3.



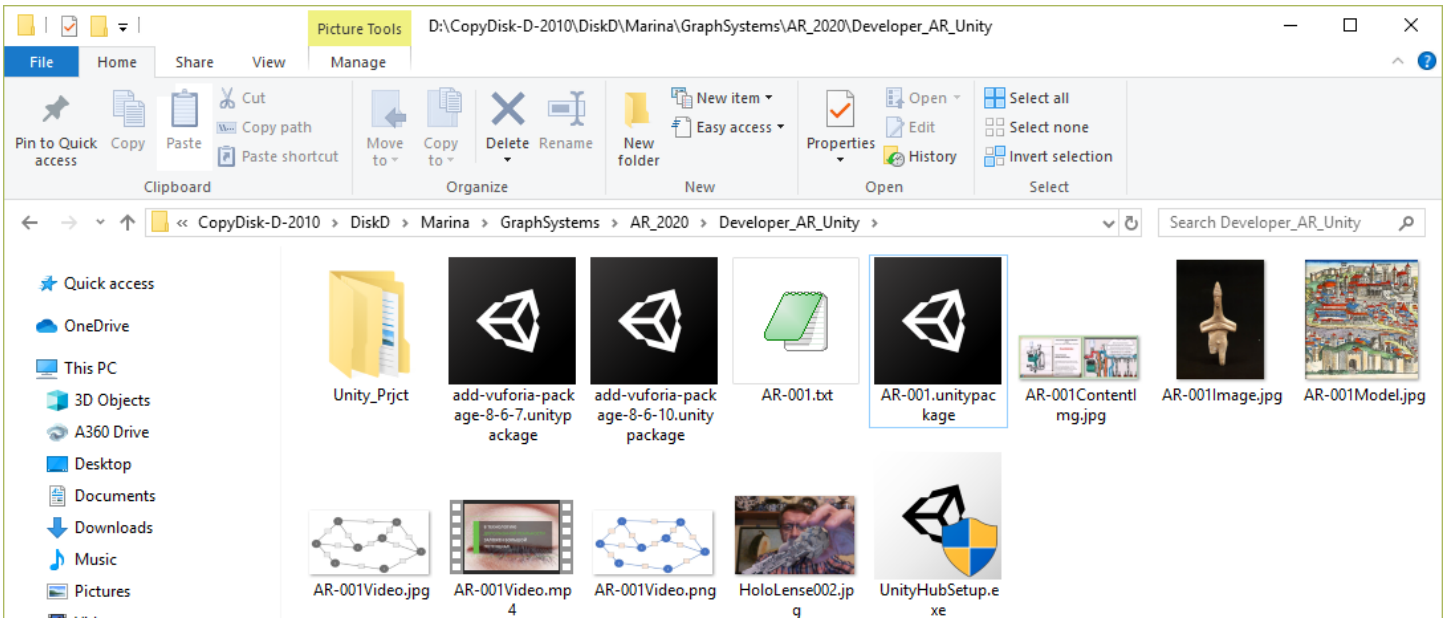
Качество этого изображения соответствует перечисленным выше критериям:



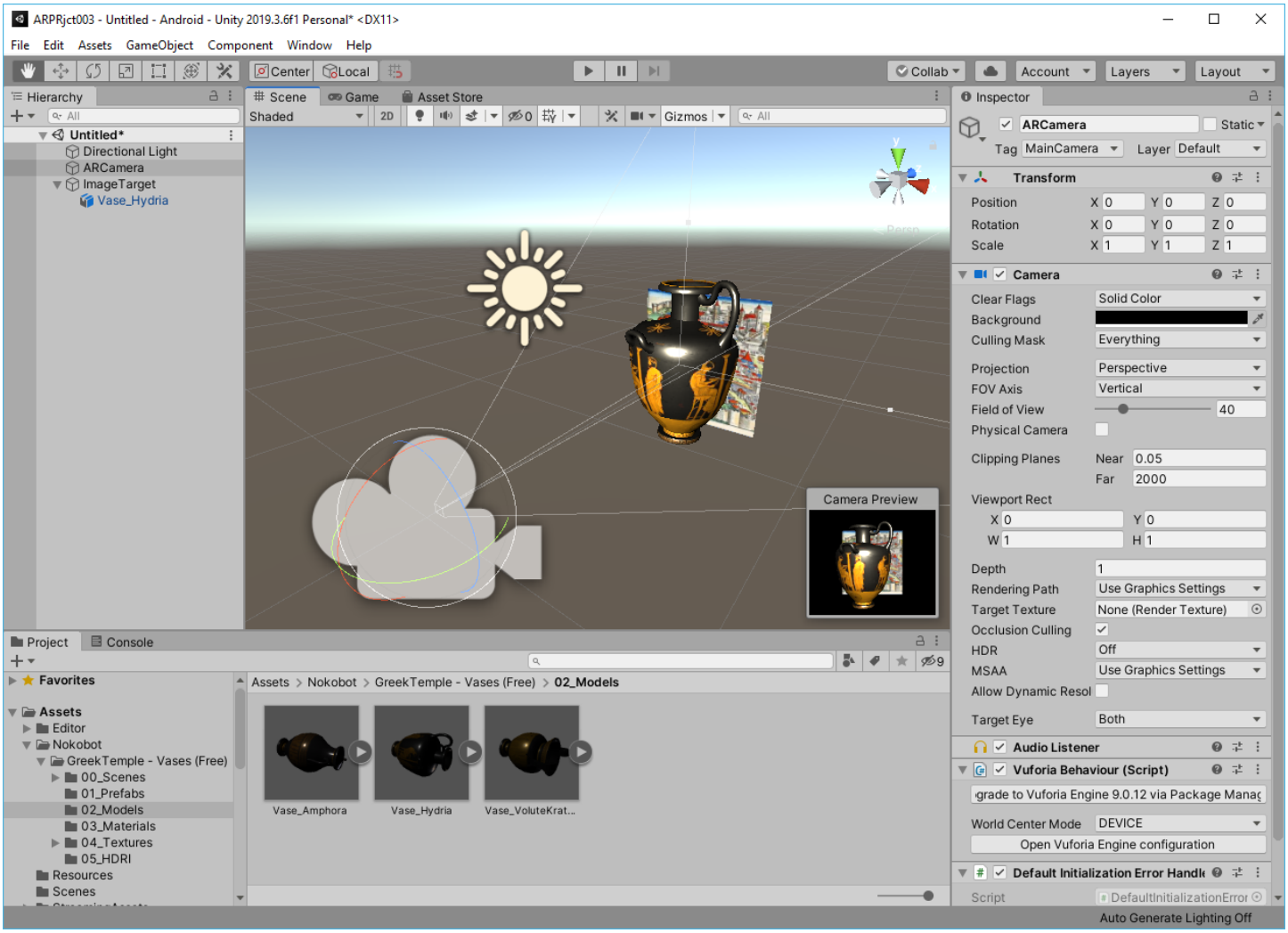
Вы можете выбрать любое другое изображение для таргета с учетом данных выше рекомендаций.

В данном описании в результате выполнения **Пунктов 1 и 2**:

- Получен лицензионный код (можно использовать полученный ранее); при этом предлагается продолжать работать в том же проекте **Vuforia – AR-001**
- В БД таргетов будем использовать уже имеющийся в БД таргет **AR-001Model.jpg.**



3. **Переходим в проект в Unity 3D.** Продолжаем работать в проекте, заведенном в ЛР№2, часть 3 (с геометрической моделью, в данном случае – **3D-модель греческой амфоры**). **В нашем примере Проект имеет название ARPRjct003.** Вы можете завести новый проект, с новой моделью – опыта у вас уже достаточно. Все необходимые шаги были проделаны вами в предыдущей ЛР. Если вы решили использовать новую модель, повторите эти шаги. В нашем случае на данном этапе выполнения ЛР состав сцены будет таким, как представлено ниже.

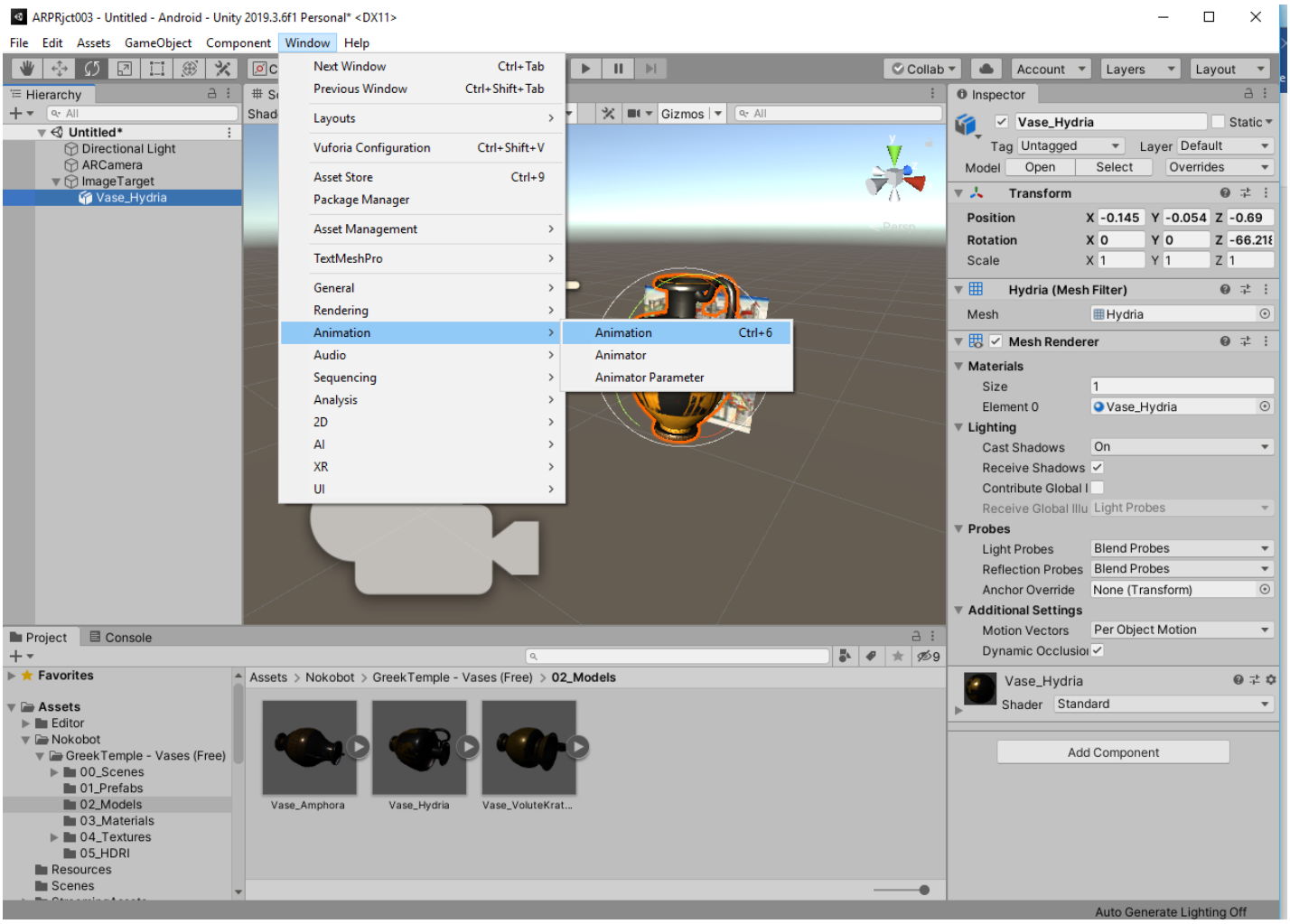


Таким образом, на этом шаге у нас уже создан объект контента, связанный с таргетом – **Vase_Hydria→** см. **Hierarchy**.

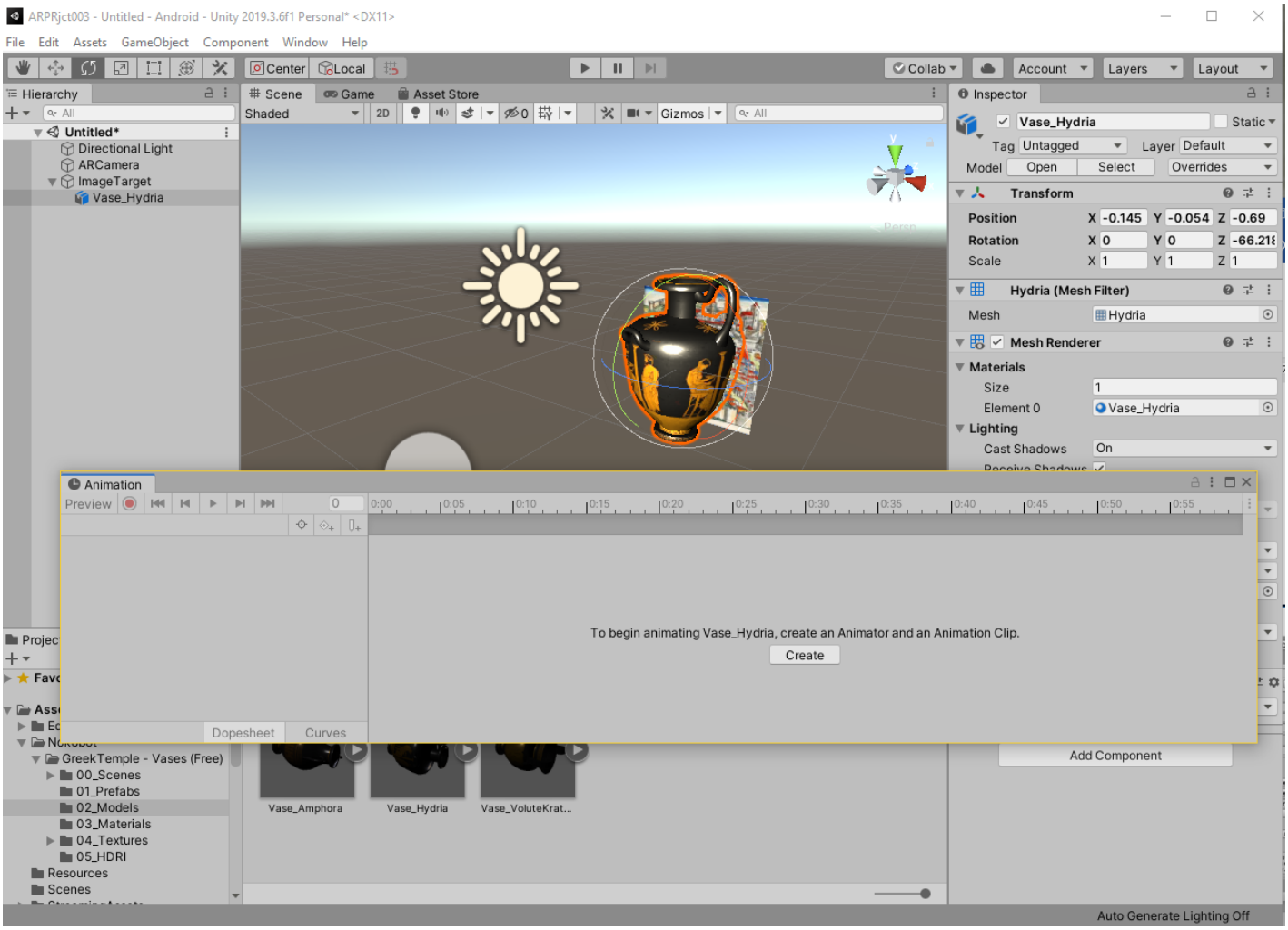
4. **Для управления виртуальным объектом Vase_Hydria** создадим поведение (**behaviour**) **3D-модели**, которым будем управлять с помощью элемента управления – **виртуальная кнопка**. Например, при нажатии на виртуальную кнопку виртуальный объект должен: вращаться, перемещаться, масштабироваться и т.д. Для этого воспользуемся базовой функциональностью **Unity 3D** и выберем достаточно сложное, мультифункциональное поведение (**behaviour**) - **Animation.**

   https://docs.unity3d.com/ru/current/Manual/AnimationSection.html

Для этого при выбранном в **Hierarchy** объекте **Vase_Hydria** выбираем в падающем меню **Window→Animation→Animation**:

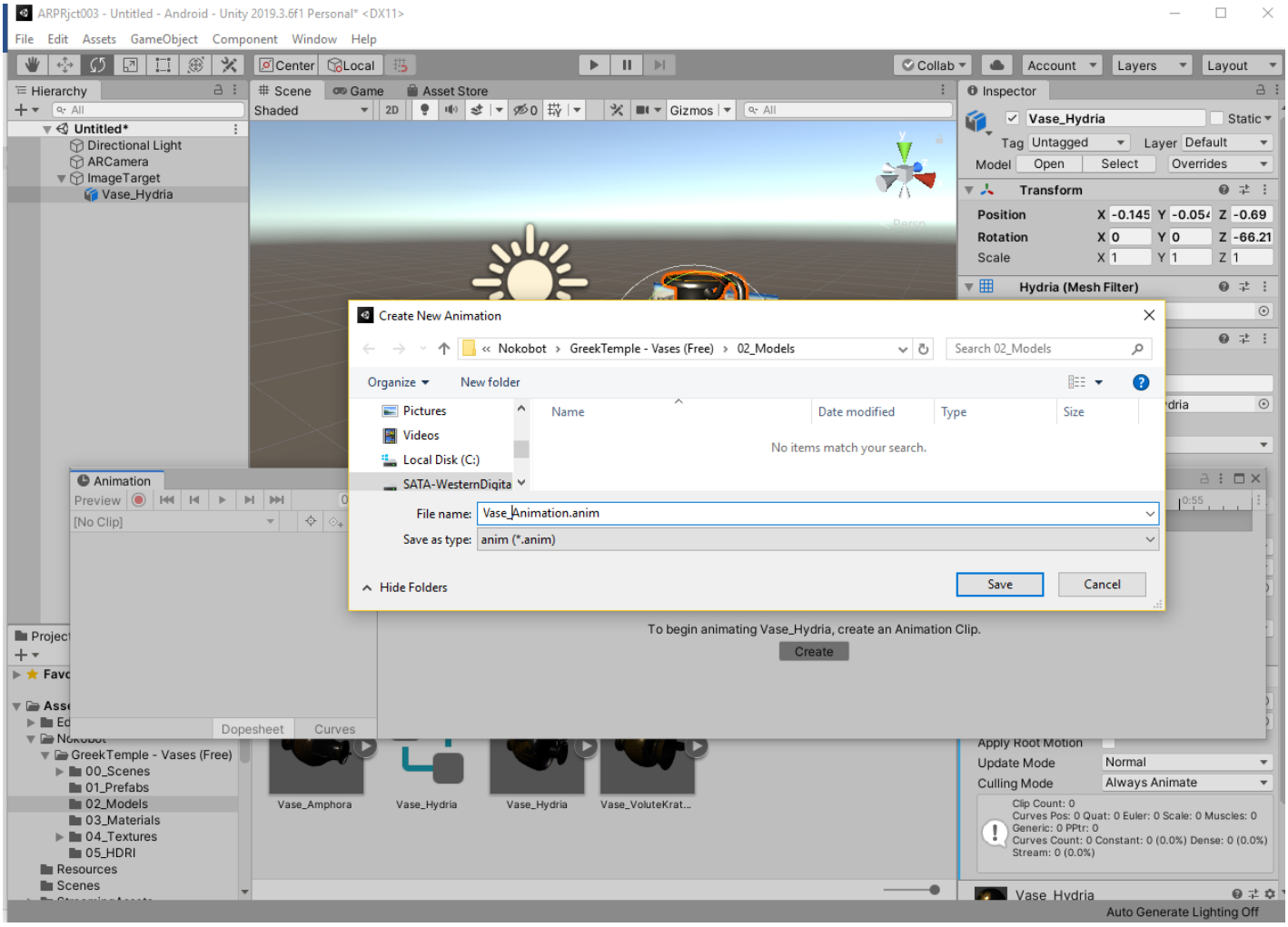


В результате появляется окно создания анимации:

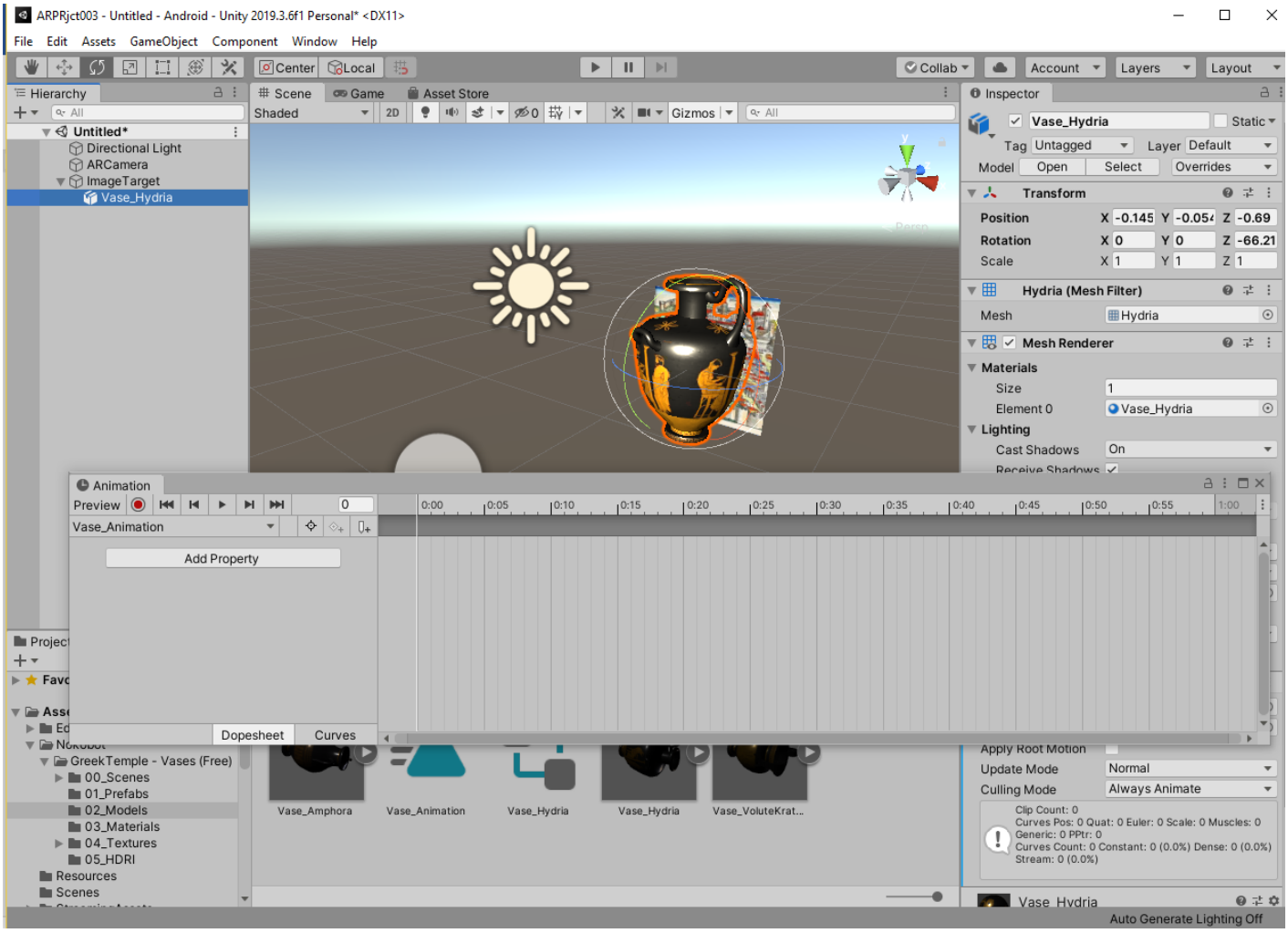


По клавише **Create** переходим к созданию анимации:

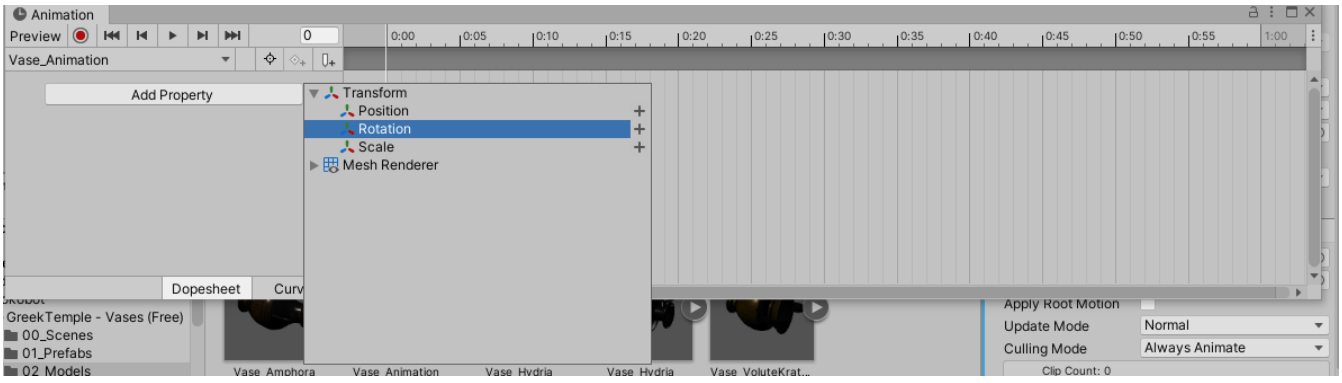
«Анимация» (сочетание известных вам стандартных операций над объектом в трехмерном пространстве – **поворот/сдвиг/масштабирование**) в данной системе представляет из себя набор данных, который сохраняется в локальной файловой системе в файле с расширением **.anim**. Присвоим ему пользовательское имя **Vase_Animation.anim.** Зафиксируйте это имя для дальнейшего использования в скриптах и программировании управляемой анимации.



После сохранения мы попадаем в окно создания **Animation**:

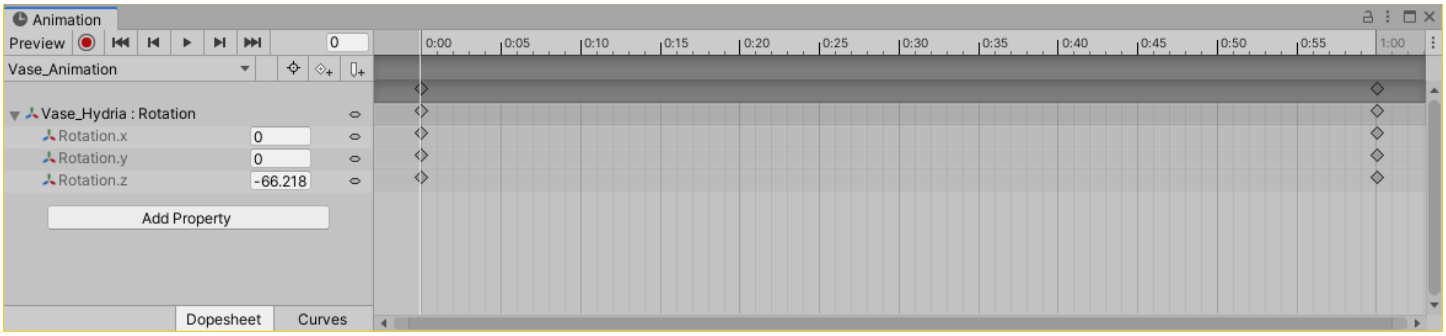


В данной ЛР для простоты создадим **Animation** в виде простого вращения объекта контента – **3D**-модели **Vase_Hydria вокруг оси Z** (в нашем случае это и есть вертикальная ось вращения) **на 360 градусов за определенное время (1сек)**. Для этого в окне **Animation** выбираем **Add Properties→Transform→Rotation:**



Как было рекомендовано выше, из всех возможных типов изменений для выполнения анимации выберем известный вам из других курсов стандартный тип изменения - **преобразование** – **Transform**, который распадается на **поворот/сдвиг/масштабирование.**

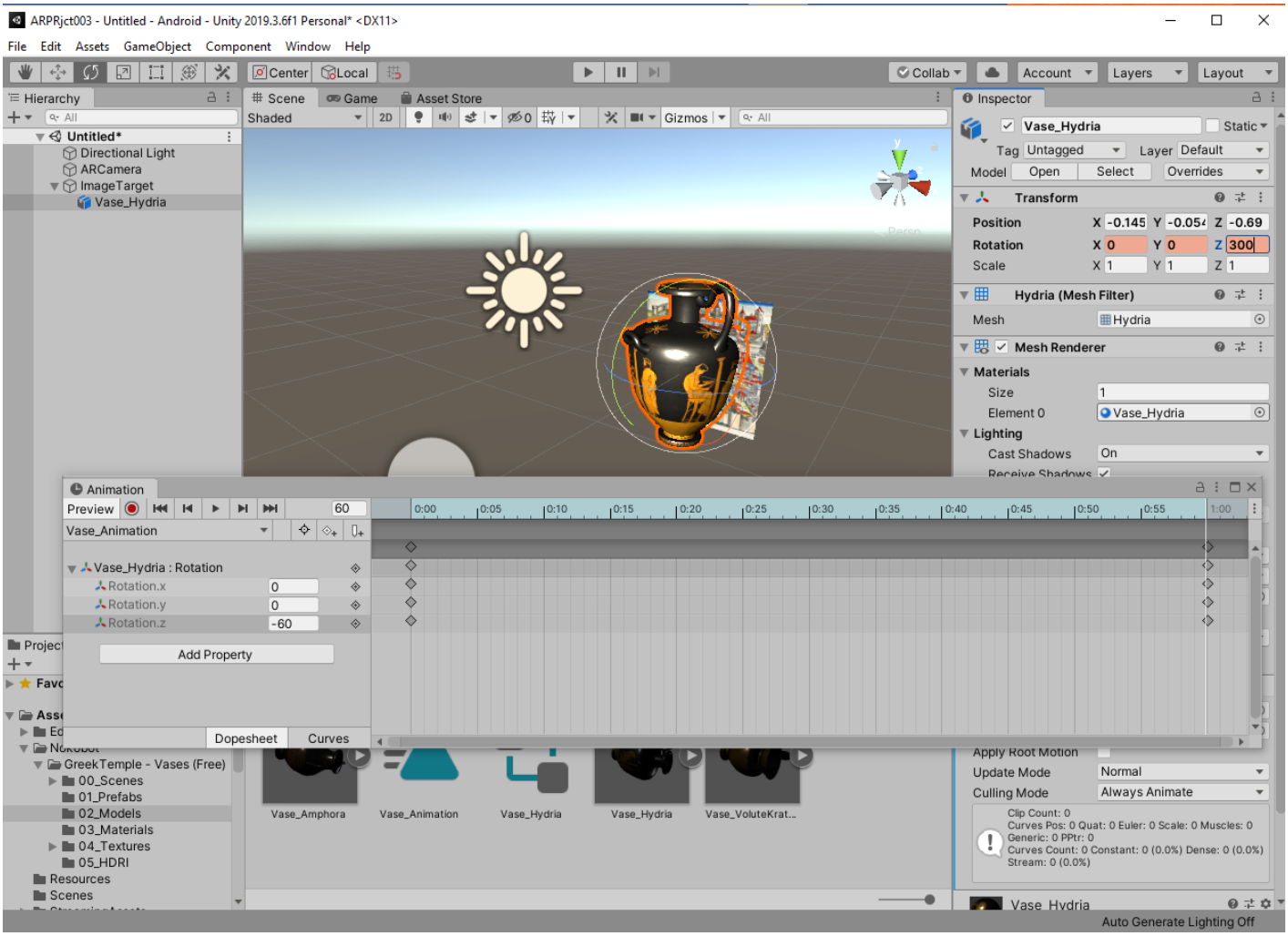
Добавляем **Rotation (+):**



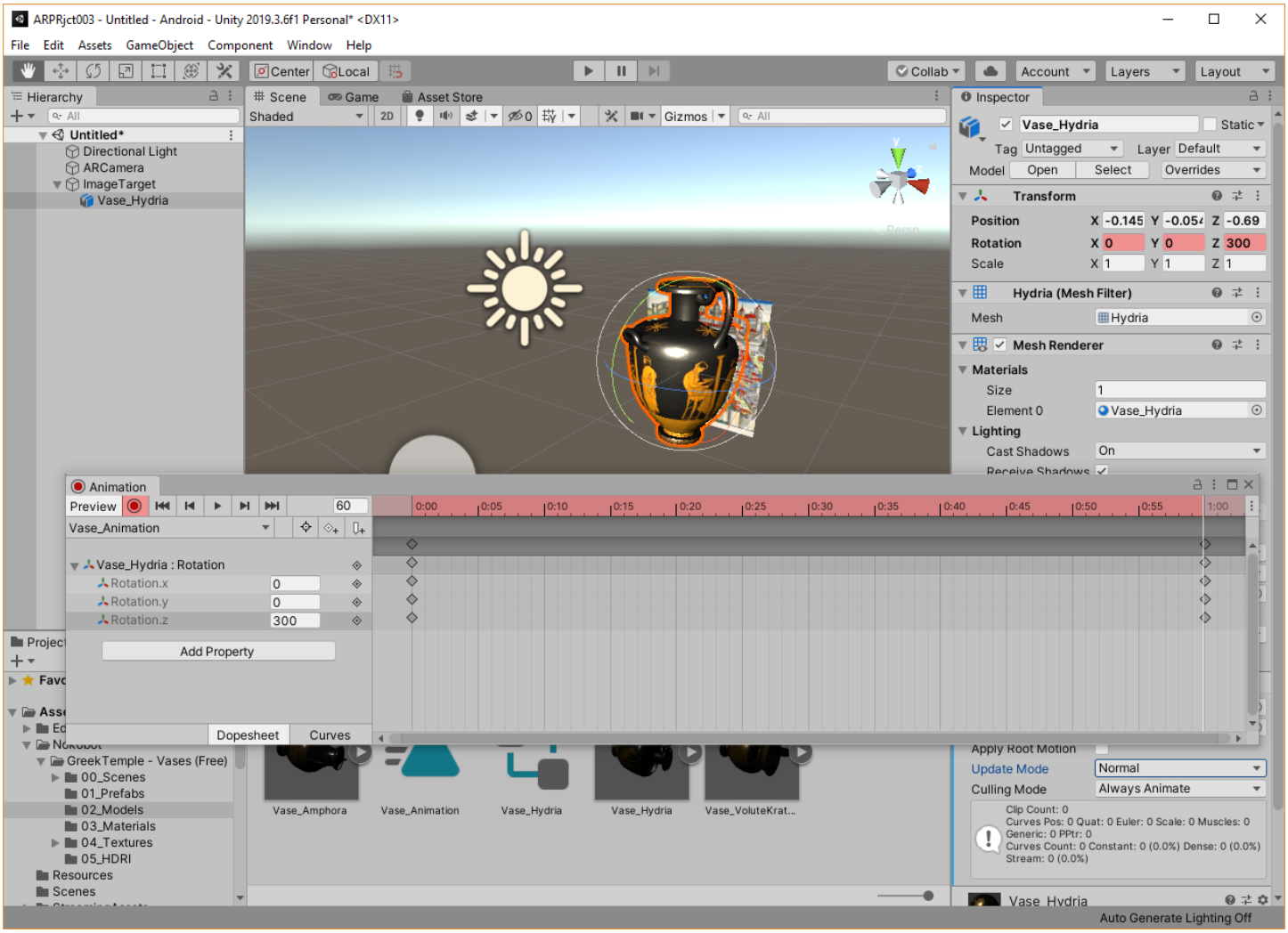
Выбираем нужный тип поворота (**Rotation**) – **вокруг оси Z**. Для этого раскрываем

Vase_Hydria : Rotation и выбираем соответствующее вращение:

В **таймлайне окна Animation** (верхняя горизонтальная шкала окна) перемещаем белый вертикальный ползунок в позицию **«1 сек»** (крайняя правая) и в **Inspector'е** редактора **Unity** для **Vase_Hydria** в позиции **Rotation Z** устанавливаем угол поворота **(300 градусов).** Обратите внимание – абсолютное значение установленного угла – **300** градусов. Это связано с тем, что изначально наша ваза была повернута на - 60 градусов (ручкой к зрителю). Для таких начальных установок, чтобы получить полный оборот вокруг вертикальной оси, необходимо установить угол поворота вокруг **Z** в **Inspector'е** для **Vase_Hydria – 300 градусов** →



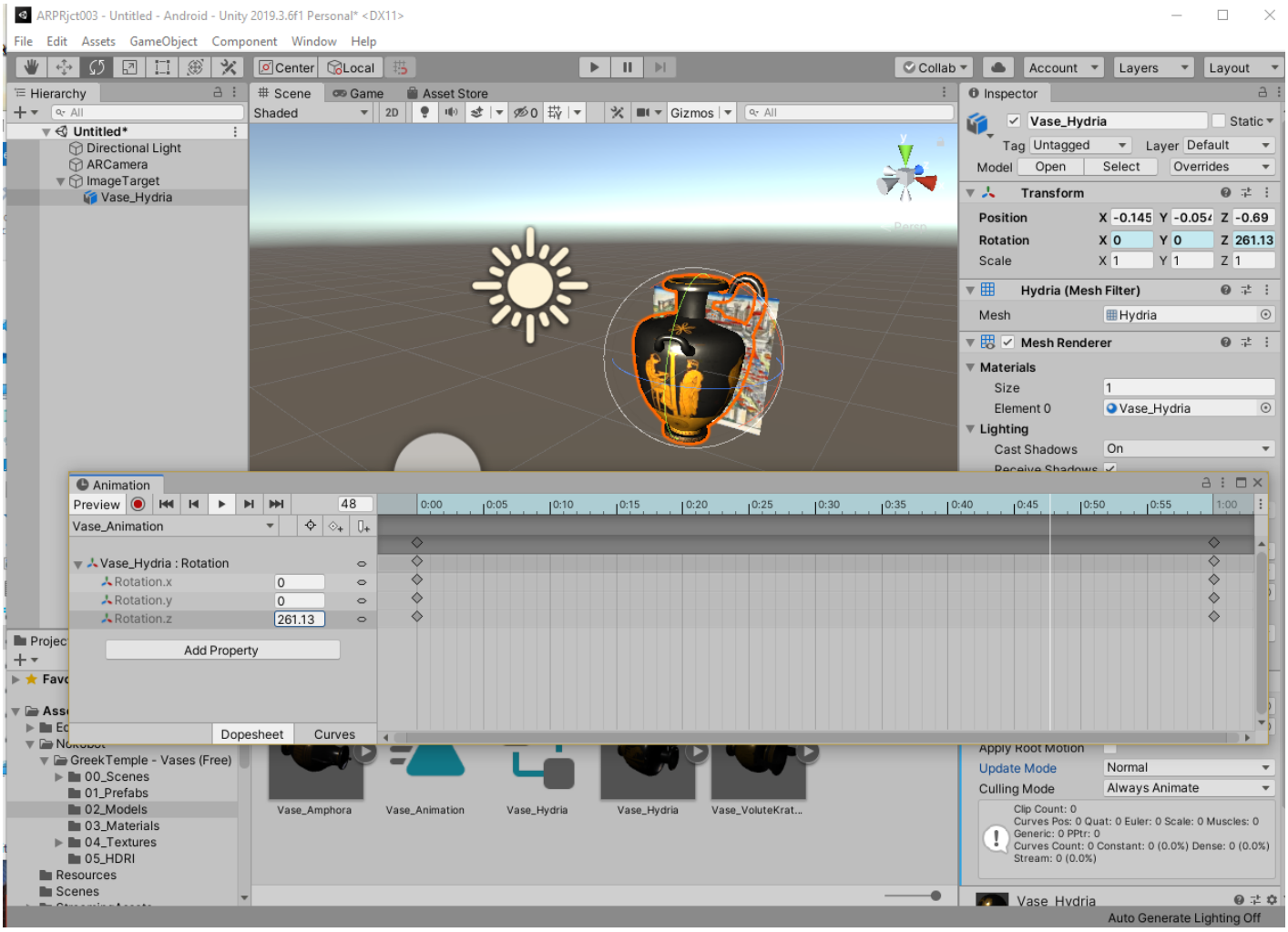
Выбираем режим записи анимации – нажимаем соответствующую кнопку (**Red Button**):



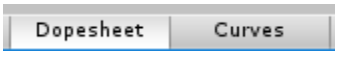
Убедитесь, что в позиции **Rotation Z в Inspector'е** установлено нужное значение угла поворота (**300**). **Если нет – установите его вручную:**

Оценим текущее состояние полученной анимации после записи. Для этого отжимаем кнопку Записи (**Red Button) и** включаем режим проигрывания – клавишу **Play:**

В результате в окне сцены **Editor'a** мы будем наблюдать записанную анимацию **Vase_Hydria** – вращение вокруг вертикальной оси (**Z**).

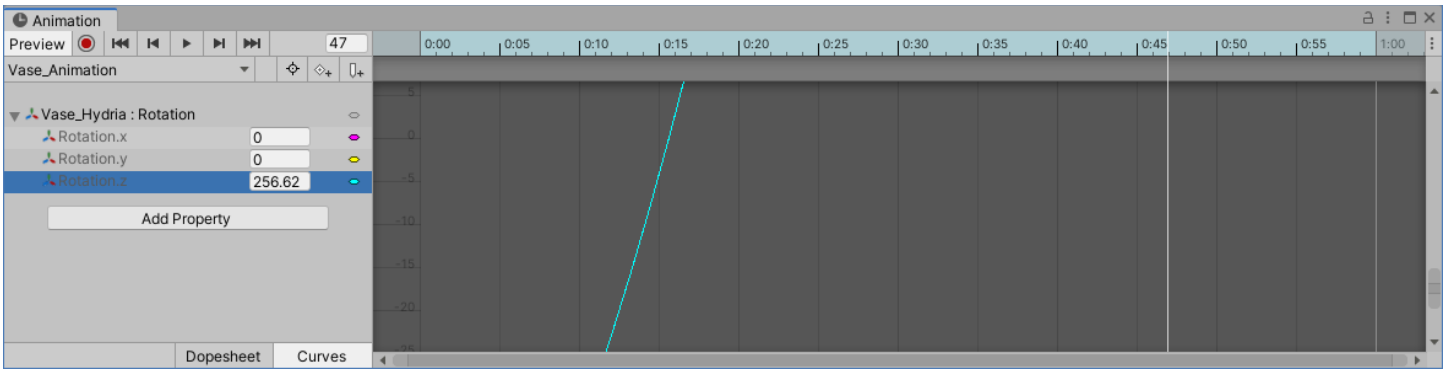


Теперь можно обработать еще один параметр анимации – скорость трансформации – у нас скорость вращения вокруг оси **Z**. Для этого в окне **Animation** переключимся в режим **Curves**



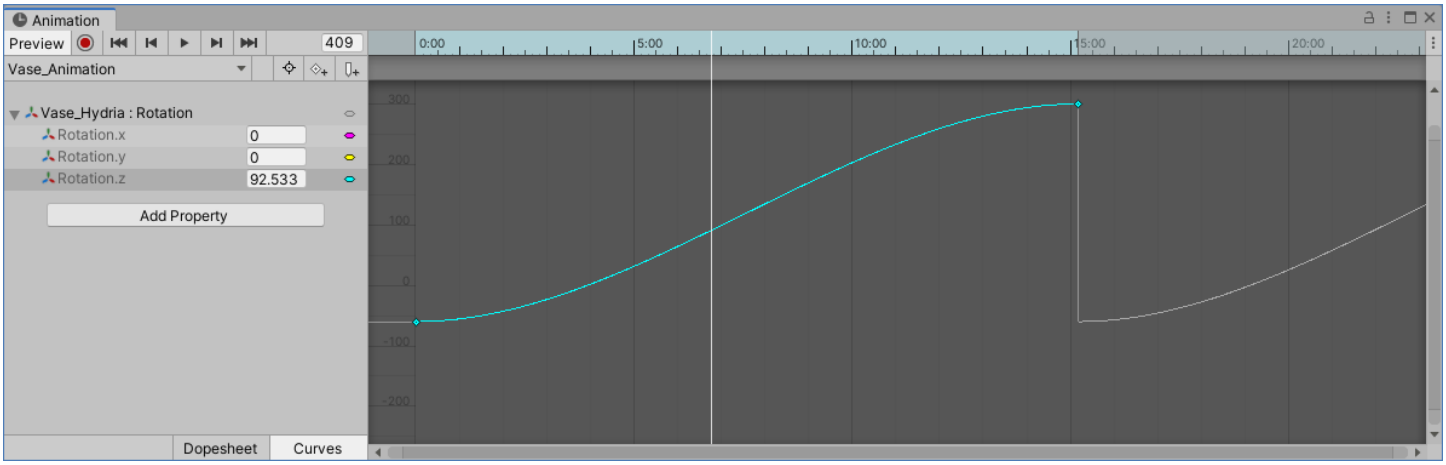
В результате получаем график изменения параметра **скорость вращения.** График свидетельствует о том, что скорость вращения неравномерна.

В первый момент вы можете увидеть в окне отображения нечто подобное:



Вертикальная шкала у левой границы серого поля экрана – это градусы. При этом вы можете увидеть только часть кривой, описывающей скорость вращения. Для того, чтобы работать со всей кривой, вам нужно поменять масштаб, для этого у вас есть ползунок у правой вертикальной

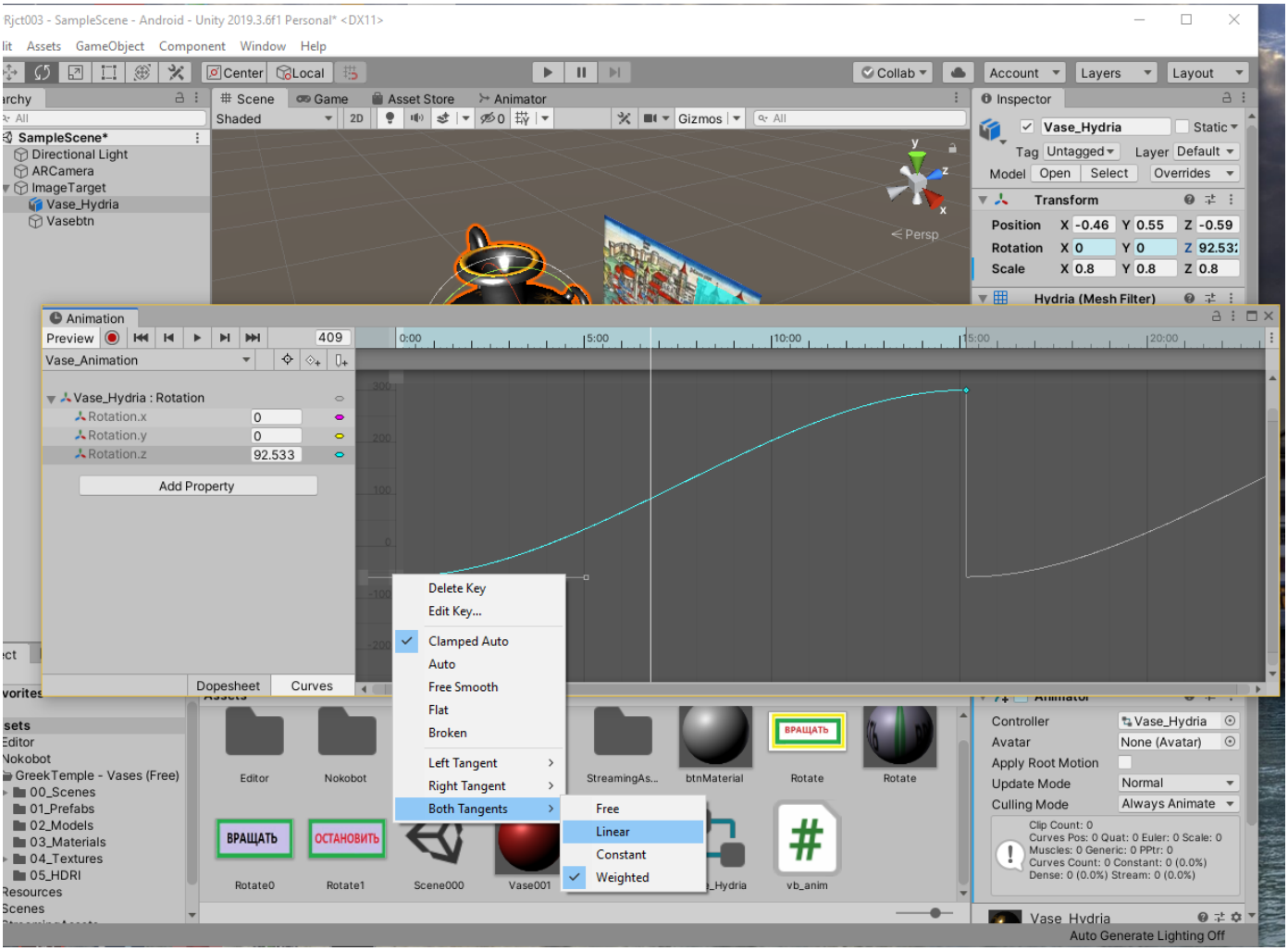
границы окна, горизонтальный ползунок, и среднее колесико мыши, изменение границ окна, с помощью которых вы можете менять масштаб внутри окна, по вертикали и по временной шкале. Добейтесь того, чтобы график скорости изменения был виден целиком. На кривой могут быть т.н. «ключи», как минимум два – в начале и конце кривой. Управлять кривизной и характером кривой можно с помощью этих ключей – добавляя/уменьшая их число, меняя их положение (с помощью клавишей мыши) на кривой и т.д. При этом можно наблюдать изменение анимации объекта в режиме предпросмотра (клавиша **Play**).



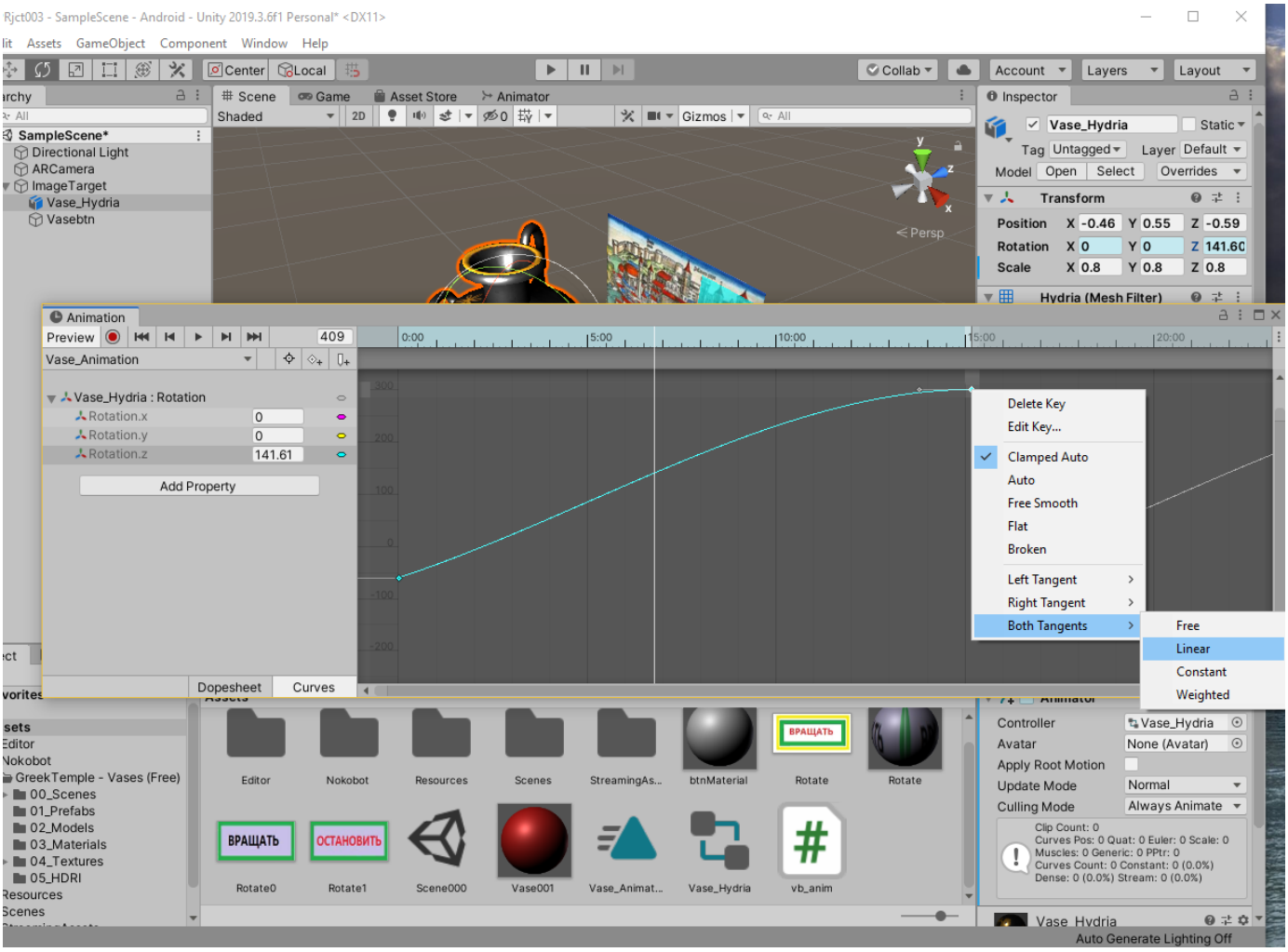
Теперь можно добиться равномерности скорости вращения. Равномерное вращение соответствует линейному характеру поведения кривой. Обратите внимание, что любая нелинейность означает ускорение или замедление вращения, изменение знака первой производной кривой – изменение направления вращения объекта контента. У разработчика есть масса инструментов, с помощью которых он может добиться различного характера анимации. Остается только освоить их все и поэкспериментировать с удовольствием.

Наша задача в данном примере – добиться равномерного вращения вазы. Для этого в начале кривой поместим указатель и по правой клавише мыши вызовем меню коррекции графика:

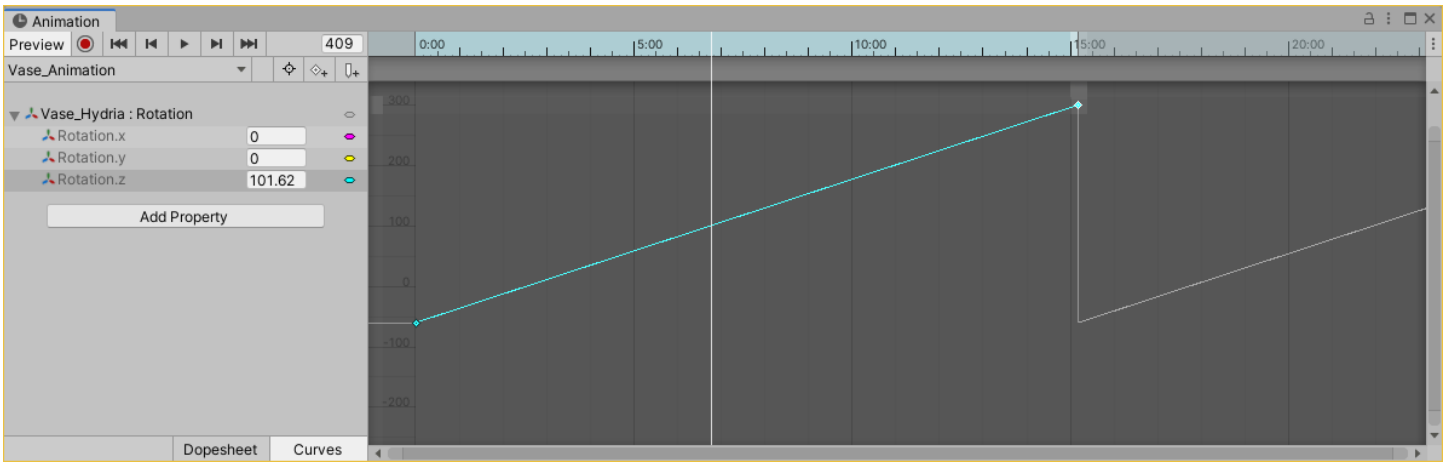
Both Tangents→Linear



Выполним те же самые действия на конечной точке **TimeLine**. В результате мы добились равномерности скорости вращения:


Rjct003 - SampleScene - Android - Unity 2019.3.6f1 Personal* <DX11>
Assets GameObject Component Window Help
Center Local
Collab Account Layers Layout
Scene Game Asset Store Animator
Shaded 2D Gizmos
SampleScene*
Directional Light
ARCamera
ImageTarget
Vase_Hydria
Vasebtn
Persp
Inspector
Vase_Hydria
Static
Tag Untagged Layer Default
Model Open Select Overrides
Transform
Position X -0.46 Y 0.55 Z -0.59
Rotation X 0 Y 0 Z 141.60
Scale X 0.8 Y 0.8 Z 0.8
Hydria (Mesh Filter)
Animation
Preview 409
Vase_Animation
Vase_Hydria : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z 141.61
Add Property
Dopesheet Curves
Delete Key
Edit Key...
Clamped Auto
Auto
Free Smooth
Flat
Broken
Left Tangent
Right Tangent
Both Tangents
Free
Linear
Constant
Weighted
Editor
Nokobot
Resources
Scenes
StreamingAs...
btnMaterial
Rotate
ВРАЩАТЬ
ОСТАНОВИТЬ
Rotate0
Rotate1
Scene000
Vase001
Vase_Animat...
Vase_Hydria
vb_anim
Controller Vase_Hydria
Avatar None (Avatar)
Apply Root Motion
Update Mode Normal
Culling Mode Always Animate
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0
Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%)
Dense: 0 (0.0%) Stream: 0 (0.0%)
Auto Generate Lighting Off

Результат:


Animation
Preview 409
Vase_Animation
Vase_Hydria : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z 101.62
Add Property
Dopesheet Curves

Работа по созданию простейшей анимации виртуального объекта **3D Модели завершена**. Элементы меню окна **Animation**, с которым вы познакомились на этом шаге, **позволят вам создавать трансформации любой сложности на любом временном интервале.**

Закроем окно **Animation** и убедимся, что в **Assets окна Project** появился новый **Asset** – **Vase_Animation**. Запомним это имя для дальнейшей работы.

На данном этапе вы можете протестировать свою сцену с анимированным объектом контента – **3D**-моделью - известным вам способом в редакторе **Unity 3D**, воспользовавшись камерой локального компьютера и выбранным таргетом.

Собранный файл **.apk** позволит продемонстрировать в режиме ДР ваш анимированный объект.

Задание для самостоятельной индивидуальной работы. ЛР №3, Часть1:

Разработать 3 Приложения ДР для:

- Вращения **3D-модели** вокруг выбранной оси (в зависимости от модели);
- Перемещения **3D-модели** по диагонали, например – из нижнего левого угла в верхний правый;
- **«Наезда»/«Удаления»** вращающейся модели на зрителя.

Разработанные в рамках ЛР № 3, часть 1 AR-Приложения необходимо продемонстрировать преподавателю. Креатив приветствуется!

Одну (а может быть и все три) из разработанных в рамках этого задания анимаций вы сможете использовать во второй части ЛР №3 для связывания с виртуальной кнопкой.