

Unity - I

Ссылки

<https://unity.com/ru/download>
<https://www.visualstudio.com/products/visual-studio-community-vs>
<https://docs.unity3d.com/hub/manual/index.html>

<https://learn.unity.com/learn/>
<https://unity.com/ru/how-to/beginner-video-game-resources>

Источники ассетов:
<https://nit.com.ru/page/materialy-k-knigam-3>

<https://assetstore.unity.com/>
<https://sketchfab.com/> - модели (бесплатные - отключить выбор лицензий Standard, Editorial), поставить флаг Downloadable.
<https://www.turbosquid.com/>
<https://free3d.com/>

<https://www.mixamo.com/> - анимации (загрузить туда модель, применить анимацию, выгрузить сборку модель + анимация)
<https://www.flaticon.com/> - иконки для интерфейса
<https://freesound.org/> - звуки
<https://www.freesfx.net/sfx/safe> - звуковые эффекты

Инструменты

Unity
Visual Studio / VSCode / Rider
Blender (аналог 3D Max) - создавать модели
Gimp (аналог Photoshop) - работать с текстурами
Audacity - редактирование звуков
FL Studio - создание музыки (профессиональный инструмент)

Установка

В этом руководстве рассмотрена установка **Unity 2022** в **Windows**. Для пользователей **Linux** надо будет ознакомиться с материалами:
Установка Unity в Linux, + Rider, Github Desktop и Git
<https://habr.com/ru/articles/654525/>

Unity Hub

Сначала надо установить **Unity Hub** - это лаунчер для установки разных версий движка **Unity**.
В Unity Hub создаются проекты игр, содержатся ссылки на обучающие материалы и разные ресурсы.
Скачать Unity Hub: <https://unity.com/ru/download>
Для использования Unity понадобится создать учетную запись - **бесплатный тип Personal**.

Visual Studio

Для работы с программным кодом надо установить среду разработки для C# - **Visual Studio Community 2022** (бесплатная версия), либо запустить ее установку вместе Unity на следующем шаге.
<https://www.visualstudio.com/products/visual-studio-community-vs>

Unity необходима **Visual Studio** для компиляции проекта, даже если рабочим редактором кода будет другая программа.

Хоткеи для Visual Studio

TAB - автоматически “допечатать” текст, который предлагает среда при анализе команд пользователя

Alt + стрелка вверх/вниз - перемещает текущую строку кода вверх/вниз

Ctrl + D - продублировать текущую строку кода (по умолчанию может быть иное сочетание клавиш)

Ctrl + L - удалить текущую строку

Ctrl + K, Ctrl + D - автоформатирование всего кода в открытом файле - отступы, пробелы

Ctrl + R, Ctrl + R - переименовать выделенную переменную во всем коде

Ctrl + K, Ctrl + C - закомментировать выделенные строки кода (поставить // в начале)

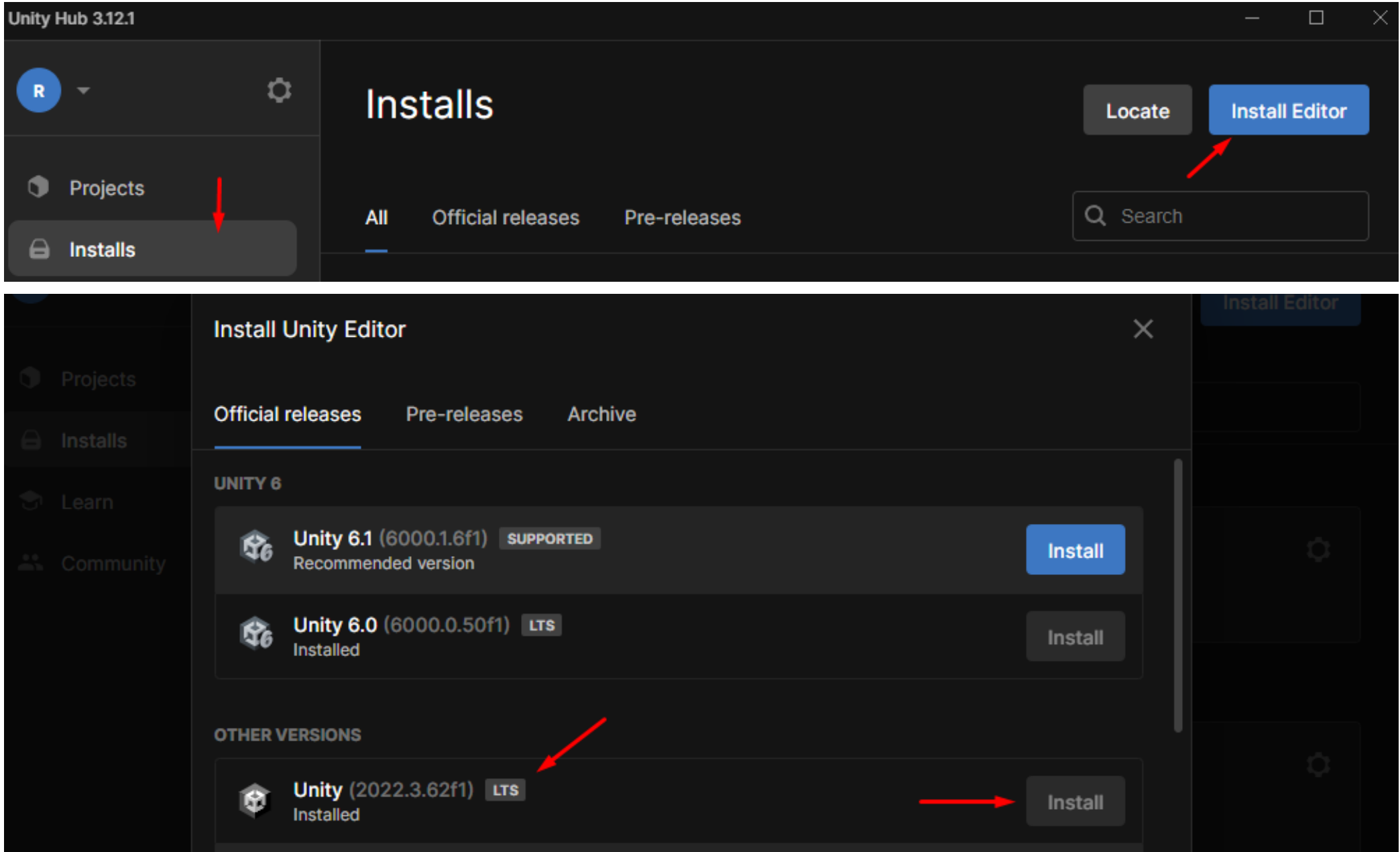
Ctrl + K, Ctrl + U - раскомментировать выделенные строки кода (удалить // в начале)

Ctrl + пробел - раскрыть окно выбора функций/методов после точки: random.

Примечание: альтернативные IDE - **Visual Studio Code** или **Jetbrains Rider** - ими можно заменить **Visual Studio** в настройках **Unity**:
Меню редактора **Edit / Preferences / External Tools / External Script Editor**.

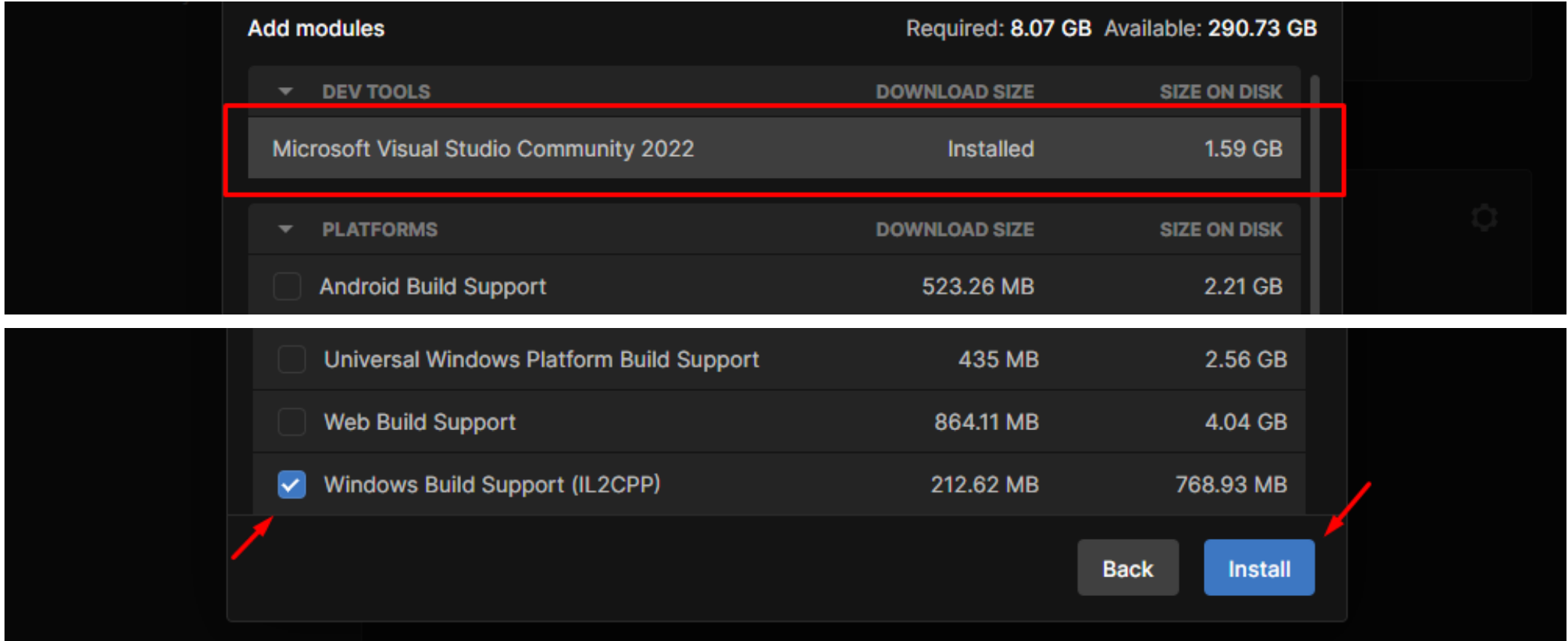
Unity 2022

При установке движка выбрать версию [Unity 2022 LTS](#) (LTS = стабильная версия и долгосрочная поддержка):



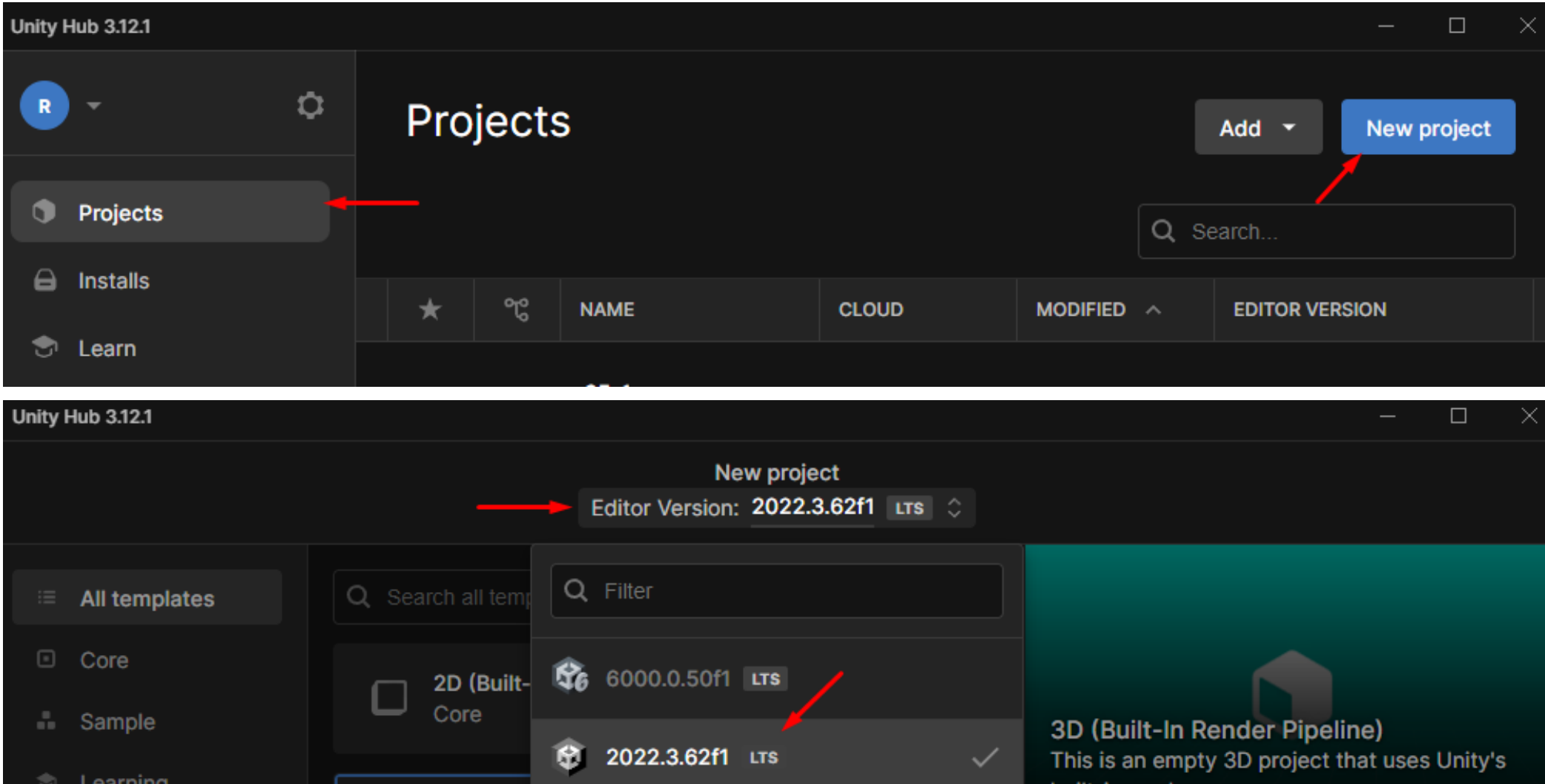
Из необходимых модулей:

- * [Microsoft Visual Studio Community 2022](#) (Community бесплатная версия) - если уже установлена, то установщик ее найдет.
- * Целевая платформа для сборки игры, в рамках этого руководства - [Windows Build Support](#).

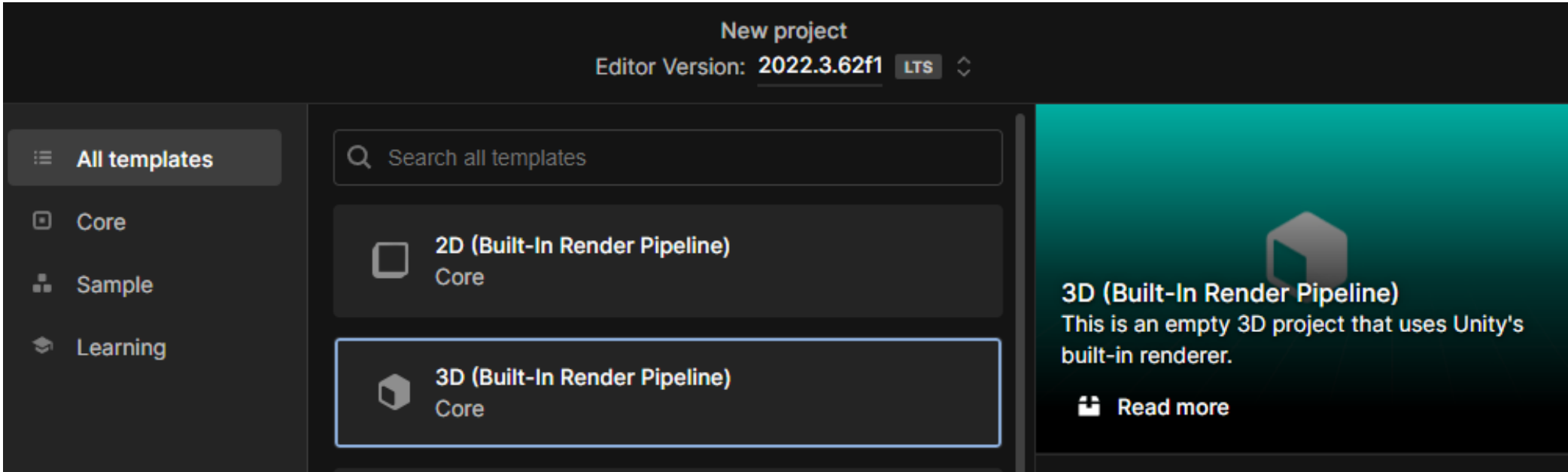


Создание проекта

Вкладка **Project** - [[New project](#)]. Если установлено несколько версий Unity, то поменять на версию 2022:



Выбрать **шаблон** в зависимости от типа создаваемой игры. Если не надо делать классическую двухмерную игру, то шаблон всегда 3D. Проект будет создан при наличии шаблона. Если шаблон еще не скачан, то сначала он будет загружен. Настройки проекта - имя, размещение на диске, имя организации пока не представляют интереса. Создать - [[Create project](#)].



В проекте хранится все содержимое игры - файлы из которых она состоит - **Assets**. Ассеты содержат в себе все: модели персонажей, окружения, текстуры, иконки, анимации, звуки, шаблоны игровых существностей, уровни (карты), скрипты.

Типы игр

Выбор шаблона-заготовки проекта определяется тем, какая игра будет создаваться. Это лишь устанавливает настройки в нужное положение, и при желании все можно поменять. В **3D** проекты изображения загружаются как **текстуры**, в **2D** изображения загружаются как **спрайты**.

3D Шутер (Counter-Strike) = 3D + камера с **перспективой** (глубиной).

2.5D Градострой (SimCity) = 3D + камера и вид сцены в **орфографическом** режиме.

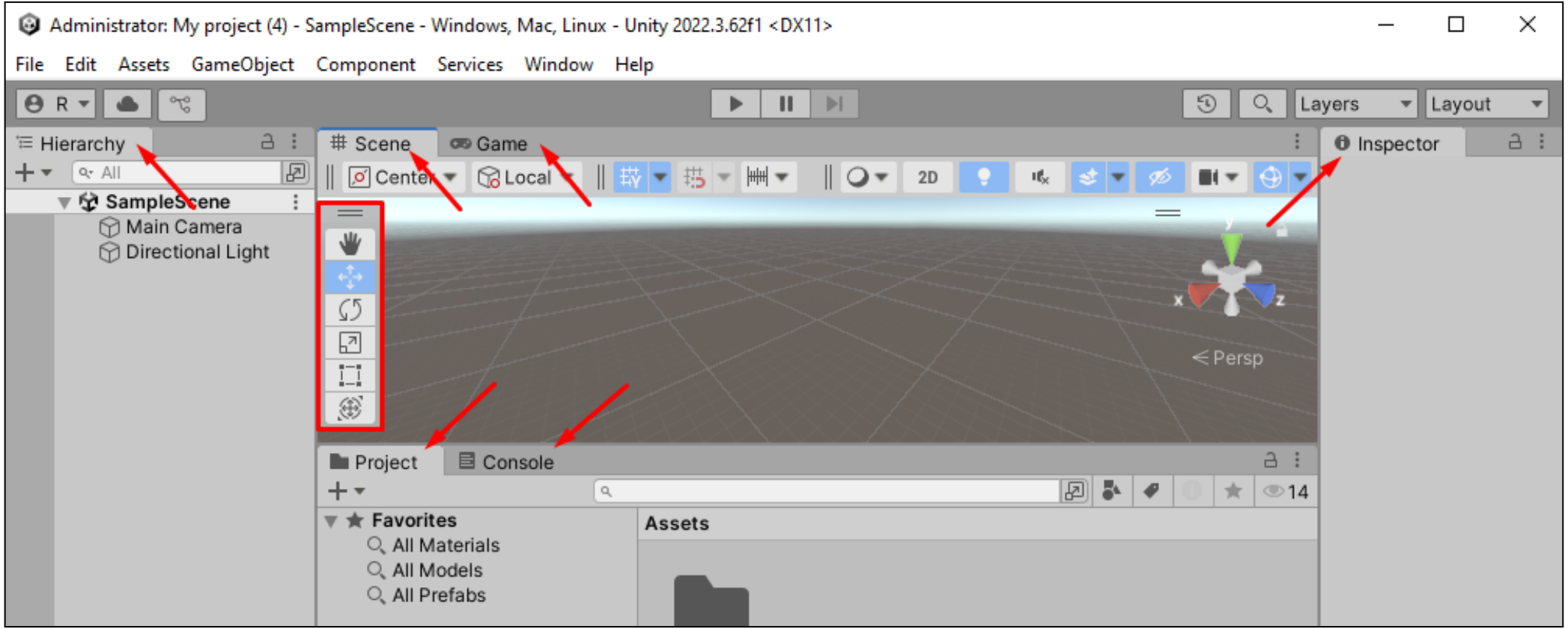
2D Платформер (Марио) = 2D + камера и вид сцены в **орфографическом** режиме.

2D игра с 3D графикой (Mortal Combat) = 3D + сцена в **орфографическом** режиме, “скользящая” камера с **перспективой**. Персонаж игрока перемещается по 2 осям.

Тема редактора

С недавних пор темная тема стала бесплатной, и можно переключаться днем на светлую, ночью на темную. Меню редактора **Edit / Preferences / General / Editor Theme**

Рабочие окна



[Hierarchy]

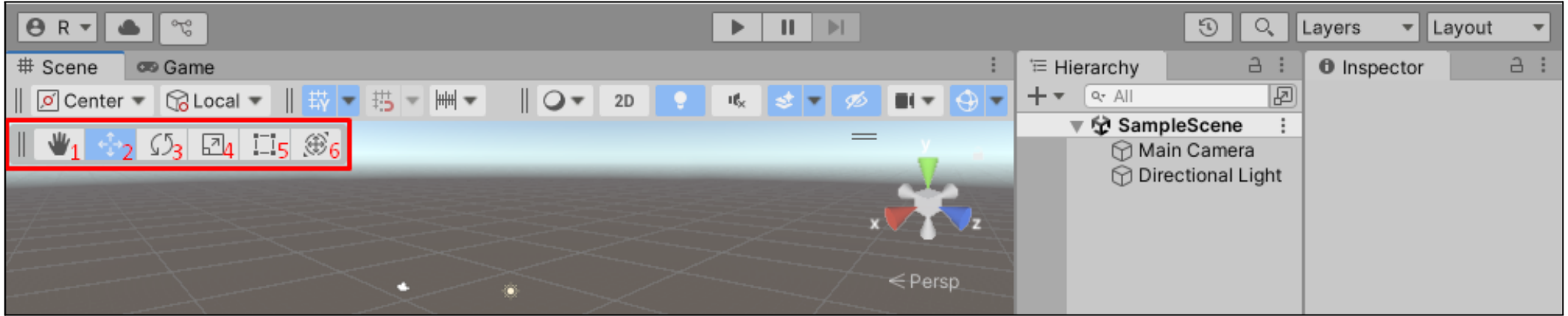
Отображает список всех объектов на **активной сцене** и взаимодействует с активной сценой. Для большей практичности это окно лучше перетащить из левой части экрана в правую, поближе к окну [Inspector].

[Scene]

Отображение содержимого **активной сцены** - здесь проводится основная работа над проектом - добавить модели и спрайты, камеры, свет и другие игровые элементы, составляющие игру.

CTRL + S :: Сохранить текущее состояние сцены.

[Верхняя панель задач] в окне [Scene] (панель можно перетаскивать и менять ориентацию расположения - через ее контекстное меню) На этой панели есть все инструменты / функции для изменения объектов сцены.



1. [Q] → Перемещение в сцене
2. [W] → Перемещение объекта
3. [E] → Поворот
4. [R] → Размер
5. [T] → 2D изменение
6. [Y] → Трансформация

[Game]

В окне [Game] можно посмотреть на результат работы с объектами на активной сцене - как ее увидит игрок.

[Inspector]

В окне инспектора добавляются и настраиваются любые свойства выбранного игрового объекта.

[Project]

Окно проекта содержит все ассеты (ресурсы / файлы), используемые игрой. Для удобства надо организовывать ассеты по папкам внутри папки [Assets]. Все что создается в процессе работы (и сцены в том числе) рекомендуется складывать внутри отдельной папки, не смешивая с подгружаемыми ресурсами. [Packages] - настройки модулей, которые не стоит трогать без опыта работы в Unity.

[Console]

Окно используется для отладки проекта или вывода системных сообщений - ошибки и т.п.

Окна можно менять по размеру, перемещать, создавать дубликаты одного окна. Сохранение конфигурации окон в раскрывающемся списке [Layout] / [Save Layout...] (верхний правый угол окна редактора). Там же выбрать готовые конфигурации или свою собственную. У некоторых окон в правом верхнем углу есть кнопка в виде замка, он блокирует обновление содержимого окна (иногда это нужно).

Game Object

Все что используется в игре и размещается в игровом пространстве является объектами - **Game Object**. Объект может быть пустым. Пустой объект не видно на сцене, и с ним почти никак нельзя взаимодействовать. Со всеми объектами можно взаимодействовать через программный код, в котором эти объекты представлены экземплярами классов. При добавлении объекту **компонентов** в окне **[Inspector]** он приобретает различные свойства - видимость, цвет, поведение и т.д.

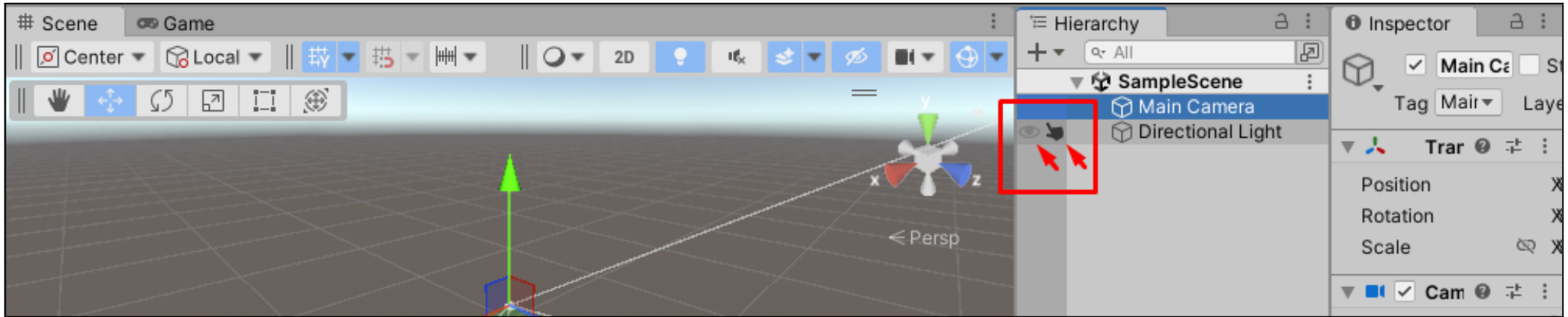
[Hierarchy]

Окно иерархии объектов отображает список всех объектов на **активной сцене** и взаимодействует с **активной сценой**.

Выбрать объект, нажать кнопку **[F]** - переместить взгляд на выбранный объект.

Позволяет включать и отключать **видимость** объектов (для отображения надо навести курсор мышки).

Позволяет заблокировать **выбор** объекта в окне сцены (для отображения надо навести курсор мышки).



Имеет окно поиска по трем фильтрам:

- имя объекта
- имя компонента (указать название компонента целиком)
- имя объекта + имя компонента

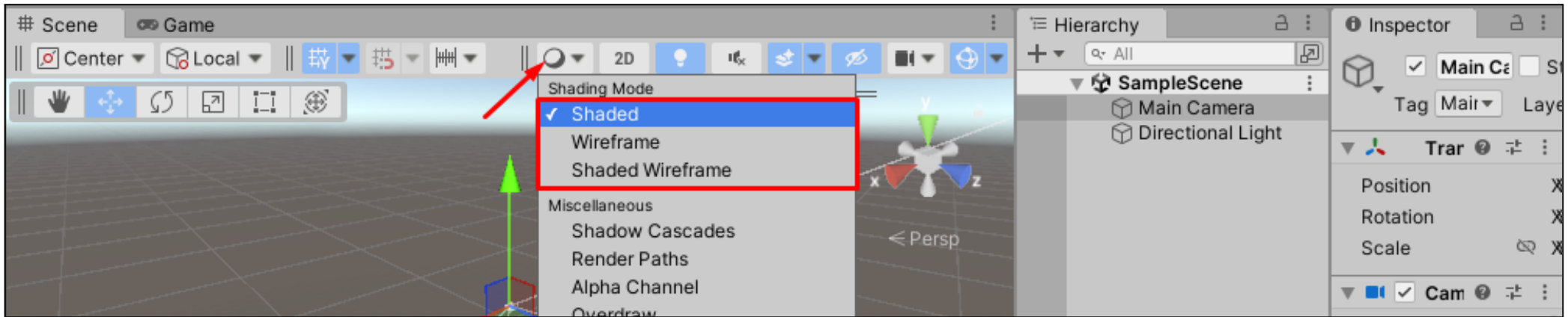
Позволяет управлять вложенностью объектов (взаимосвязью) - перетащить один объект в другой в окне иерархии объектов.

В зависимости от положения главного объекта, вложенные / связанные с ним объекты будут сохранять относительные координаты - положение от центра основного объекта. Если эту связь удалить, то вложенные объекты изменят свои глобальные координаты на обновленные, соответствующие их положению на сцене относительно центра сцены.

[Scene]

Сцены это файлы, которые хранят свой список объектов, их координат и размерностей. Сцена = игровой уровень.

Параметры камеры для отображения содержимого сцены:



Режимы отображения сцен - группа **[Draw Mode]**:

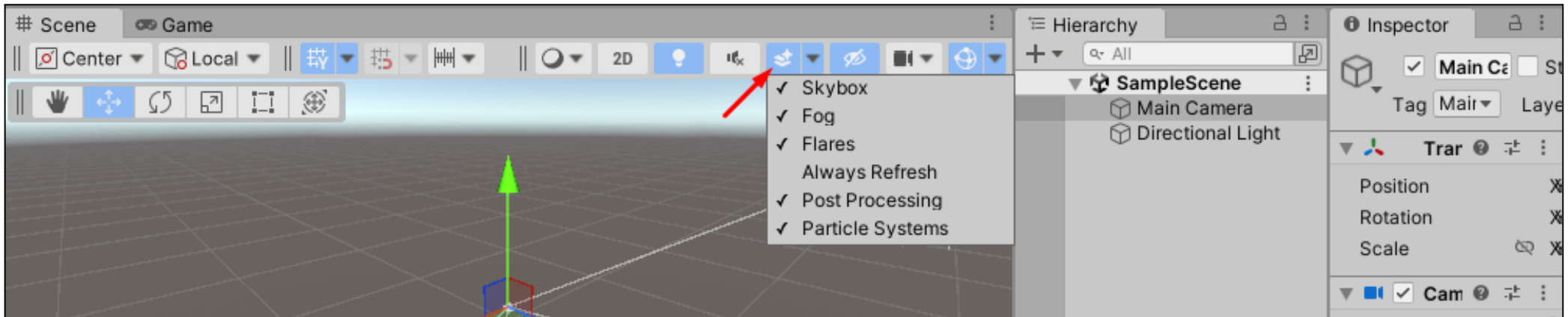
- **[Shaded]** - вариант по умолчанию, отображать как в игре
- **[Wireframe]** - без отображения текстур, только контуры объектов (вершины, ребра и грани)
- **[Shaded Wireframe]** - совмещенный режим

[2D] - кнопка переключения между 3D в 2D режимами отображения. В 2D режиме у камеры пропадает перспектива, и пропадает возможность редактирования объектов по оси Z.

[Тип освещения] - при включении используются только источники света расположенные на сцене. Когда она выключена на сцене имитируется свет, исходящий от точки обзора.

[Воспроизведение звука] - включить / выключить.

[Меню отображения визуальных эффектов] на сцене.

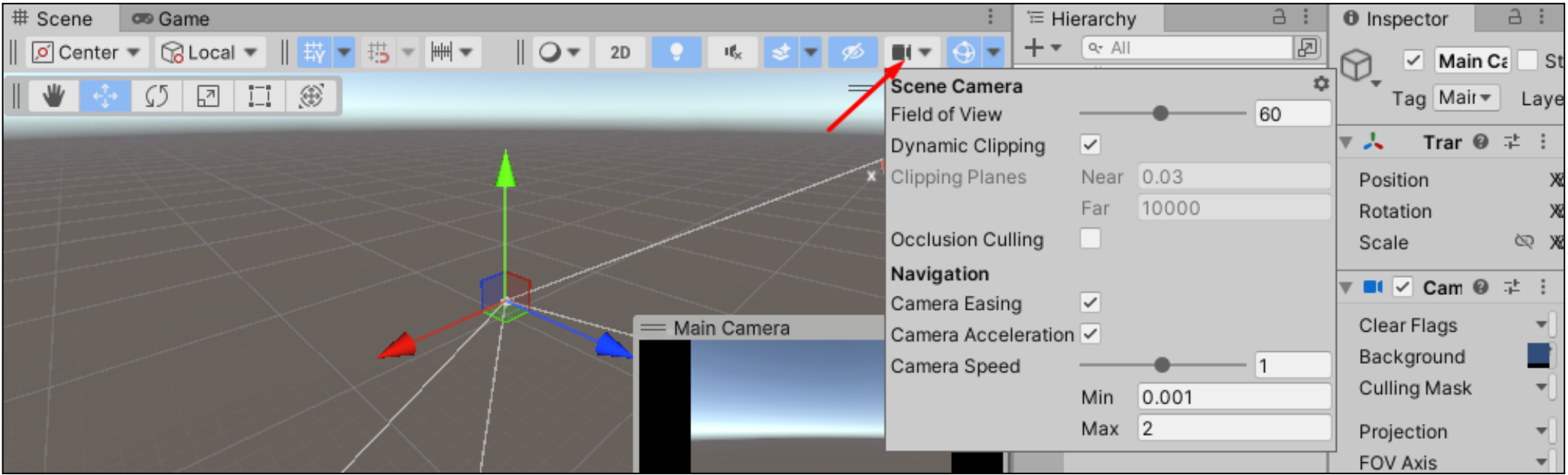


Здесь можно выбрать какие именно эффекты надо или не надо отображать на сцене.

- **Skybox** - небо
- **Fog** - туман
- **Flares** - блики
- **Always Refresh** - обновление анимации материала
- **Post Processing** - постобработка
- **Particle Systems** - частицы

[Скрытие невидимых объектов] - иногда нужно на время спрятать какой-либо объект, чтобы он не отображался на сцене.

[Настройки камеры] - взгляд на сцену.



- **Field of View** - поле зрения - угол обзора и дальность прорисовки
- **Dynamic Clipping** - динамическая регулировка отрисовки объектов
- **Clipping Planes** (доступно если выключить **Dynamic Clipping**) - отвечает за дальнюю и ближнюю плоскость камеры
Near - как близко к игроку должен быть объект, чтобы он не отображался в кадре
Far - как далеко к игроку должен быть объект, чтобы он не отображался в кадре
- **Occlusion Culling** - инструмент оптимизации, который режет сцены на блоки и отрисовывает только те, что видны игроку
- **Camera Easing** - плавность движения камеры (взгляда)
- **Camera Acceleration** - ускорение движения камеры (взгляда)
- **Camera Speed** - скорость движения камеры (взгляда)

[Опции Gizmos] - отображение границ объектов - вершины, ребра и грани моделей и другие вспомогательные элементы.

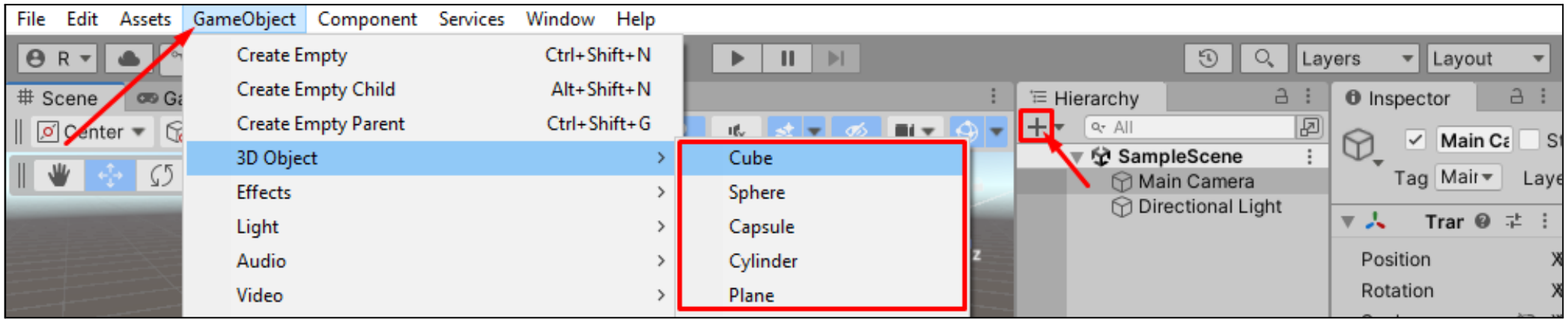
✂ Навигация на сцене

В режиме **перемещения объектов** [W] (режим по умолчанию) кнопки мыши могут менять вид взаимодействия в зависимости от модификатора:

	ЛКМ		выделение объектов
	ПКМ		управление взглядом на сцену (режим полета), кнопки WASD+QE - смещение, приближение / отдаление
	Колесо	объект	отцентрировать взгляд на объект
	Колесо	прокрутка	отдалить / приблизить взгляд медленно
[F]	или	ЛКМ+ЛКМ	отцентрировать взгляд на объект, если уже отцентрирован - приблизить / отдалить
	ЛКМ	движение	рамка выделения объектов
	ПКМ	движение	повернуть взгляд на сцене
	Колесо	движение	сместить взгляд на сцене
ALT	ЛКМ	движение	вращать взгляд вокруг центра текущей позиции / объекта - движение по орбите
ALT	ПКМ	движение	отдалить / приблизить взгляд
ALT	Колесо	движение	сместить взгляд на сцене
CTRL	ЛКМ		поочередно на объектах - выделение нескольких объектов / на выделенных - снять выделение
CTRL	ПКМ		-
CTRL	Колесо		-
CTRL	ЛКМ	движение	рамка для снятия выделения объектов
CTRL	ПКМ	движение	-
CTRL	Колесо	движение	-
SHIFT	ЛКМ		поочередно на объектах - выделение нескольких объектов / на выделенных - снять выделение
SHIFT	ПКМ		режим полета камеры с большей скоростью и с ускорением
SHIFT	Колесо		отцентрировать взгляд на объект
SHIFT	ЛКМ	движение	рамка выделения объектов
SHIFT	ПКМ	движение	повернуть взгляд на сцене
SHIFT	Колесо	движение	сместить взгляд на сцене (быстро)

✂ **Работа с объектами**

Создать новый объект на сцене можно через меню редактора, либо аналогичную команду контекстного меню в окне **[Hierarchy]** или **[+]**.
Каждому новому объекту надо дать имя (или оставить текущее шаблонное).



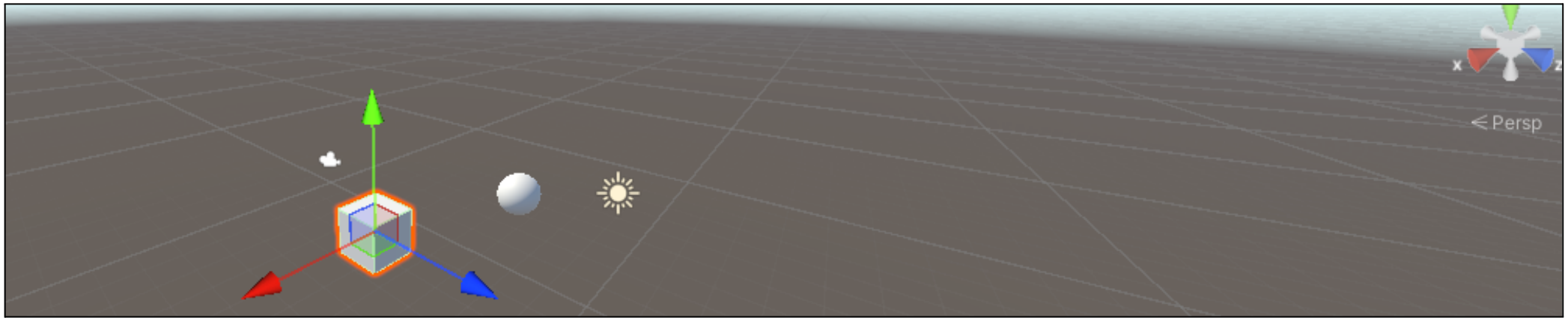
Такие примитивы обычно используются временно при тестировании, пока нет подходящей модели объекта.

Plane - по умолчанию размещается горизонтально, как поверхность - имитирует пол и земля.
Quad - по умолчанию размещается вертикально, как стена, имитируя отдаленные изображения.

Выбрать инструмент для работы с объектами на сцене:

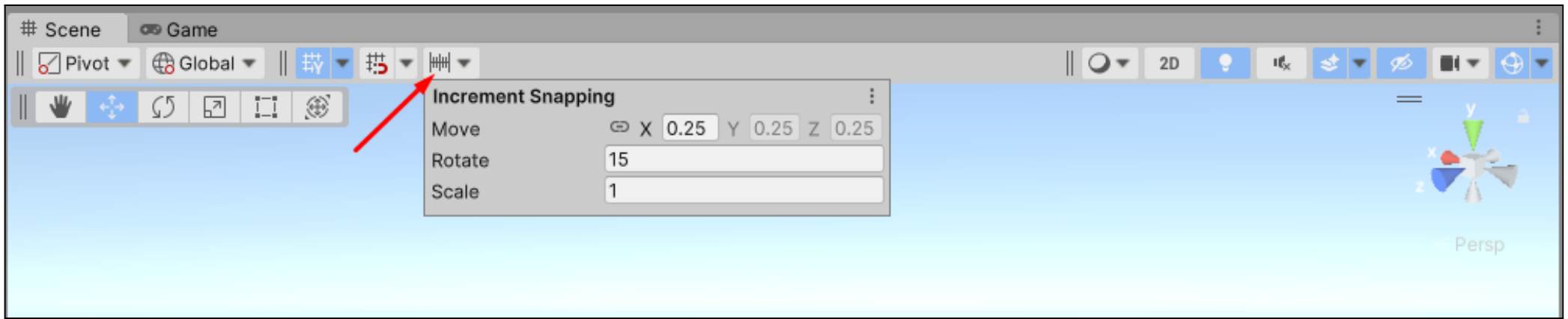
- [Q] → Обзор в сцене
- [W] → Перемещение объекта
- [E] → Поворот, цветные дуги поворачивают по одной оси, серый круг свободно вращает объект относительно точки обзора
- [R] → Размер - изменение размеров по одной оси
- [T] → 2D изменение - изменение размеров объекта в плоскости
- [Y] → Трансформация - изменение местоположения, размеров и поворота сразу

При открытии проекта для сцены автоматически выбран инструмент **перемещения объектов [W]**.
Поэтому для перемещения объекта на сцене достаточно выделить его и потянуть за одну из стрелок нужного направления.
Боковые цветные поверхности перемещают объект сразу в двух осях координат.



В режиме перемещения с зажатым **Ctrl** объект будет перемещаться на фиксированное расстояние - магнититься к сетке, аналогично с поворотом объекта и изменением его размеров.
В режиме перемещения **Ctrl + Shift** помогают выполнить привязку объекта к поверхности (к полу, к земле). Это позволяет избежать висящих в воздухе объектов.
Режим перемещения объектов с привязкой к вершинам активируется при зажатой кнопке **[V]**.

Задать параметры для интервалов при перемещении, повороте и масштабировании - кнопка на панели **[Snap Increment]**:

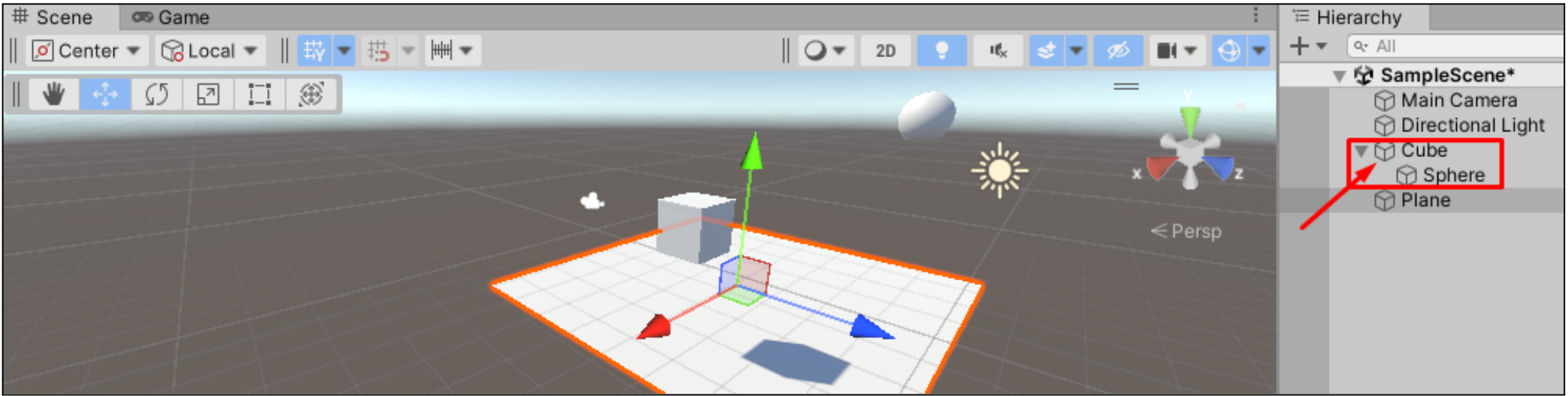


Соседняя опция позволяет перейти в режим примагничивания без необходимости использовать кнопку **Ctrl**:



В режиме поворота **Ctrl + Shift** помогают повернуть объект в нужном направлении. Взгляд выделенного объекта направляется на точку поверхности (коллайдер) другого объекта.

Установлена зависимость объекта сфера от объекта куб через окно **[Hierarchy]**, сфера перемещена в куб:

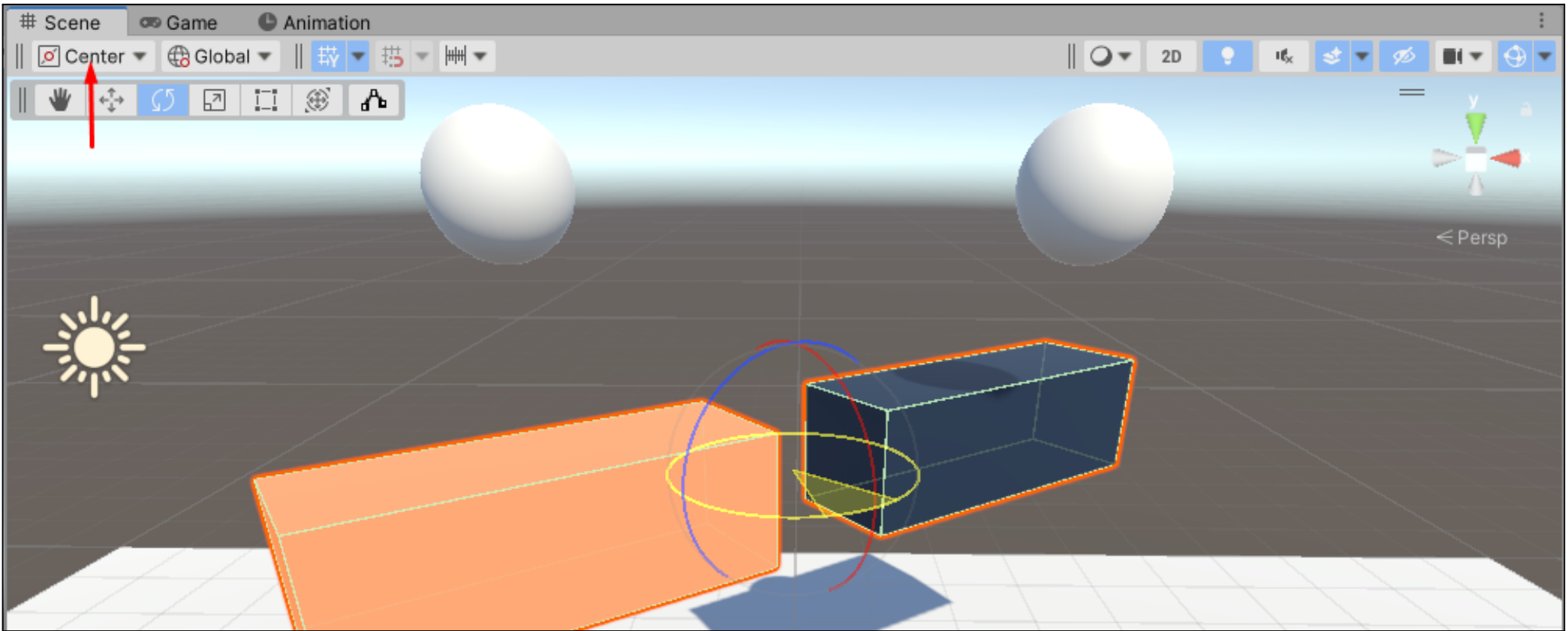


Копировать объект - **Ctrl + D**.

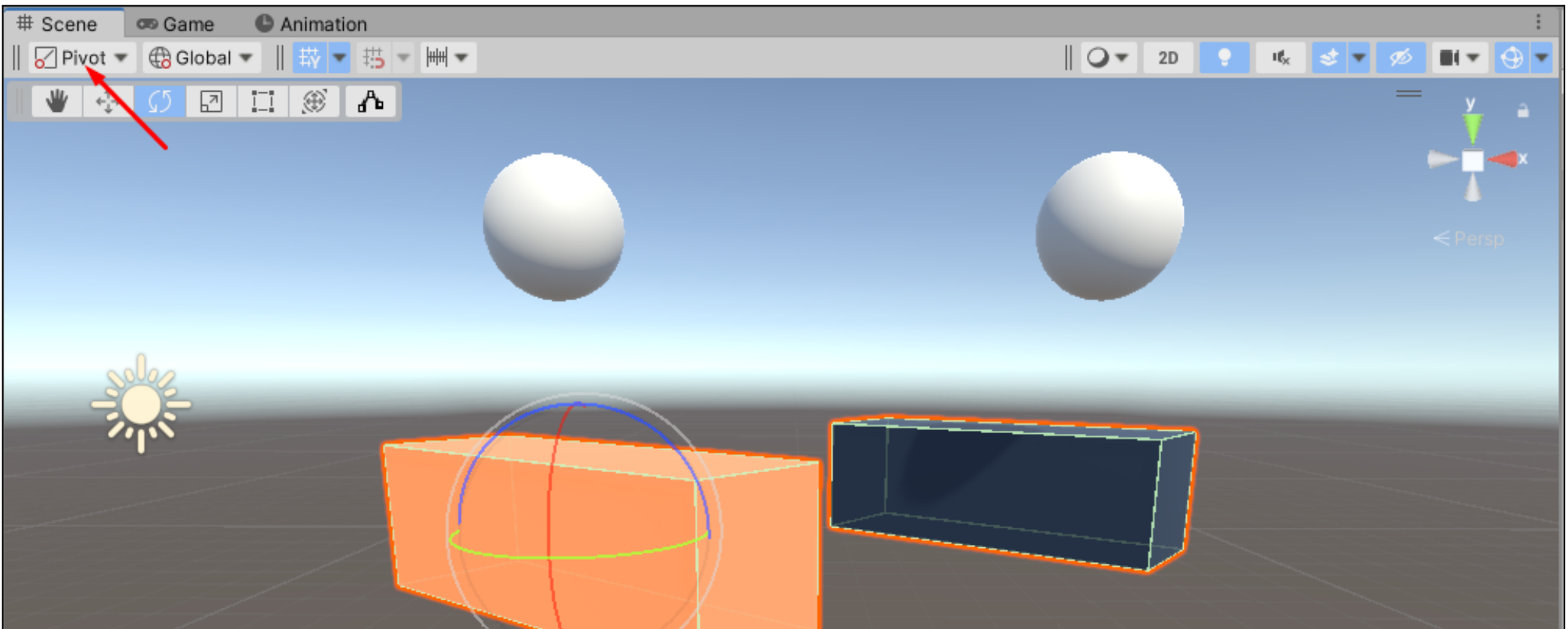
Удалить объект - **Delete**.

Если объект состоит из нескольких вложенных объектов, то при удалении основного будут удалены все связанные вложенные в него объекты. При удалении вложенного объекта - удалится только он сам. Аналогично с перемещением, поворотом, масштабированием.

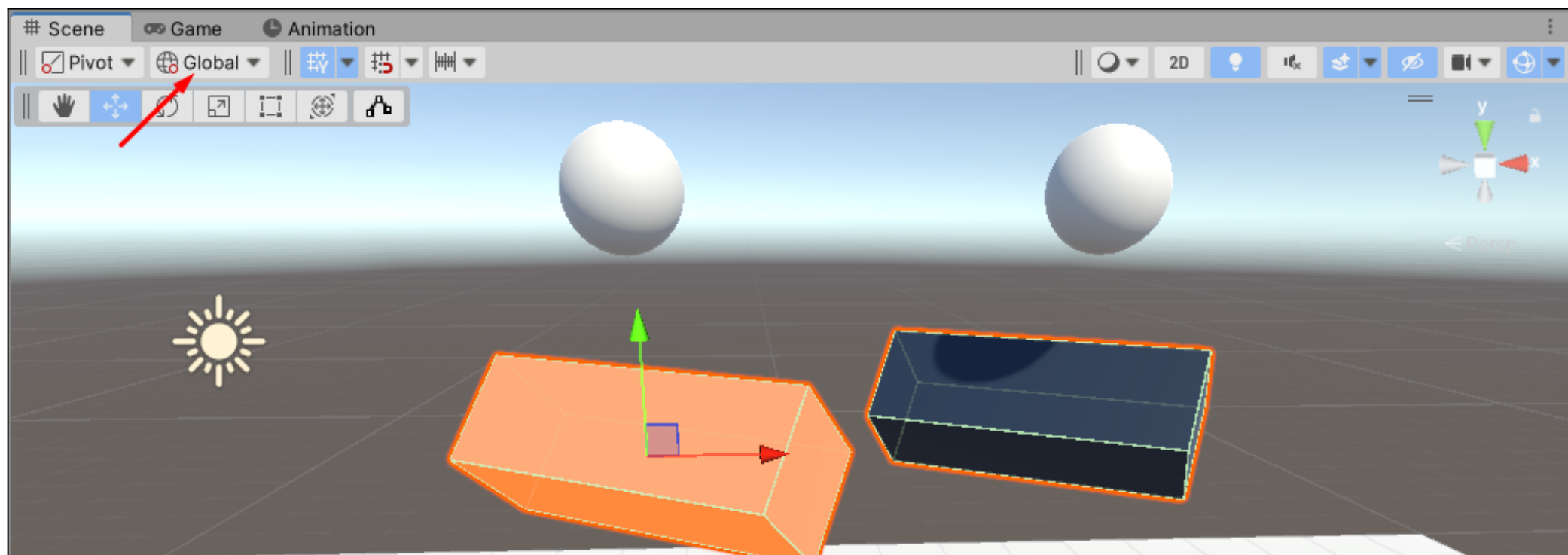
При активной опции **[Center]** группа выделенных объектов поворачивается как один большой, вокруг общего центра.



При активной опции **[Pivot]** каждый объект в группе вращается вокруг собственного центра:

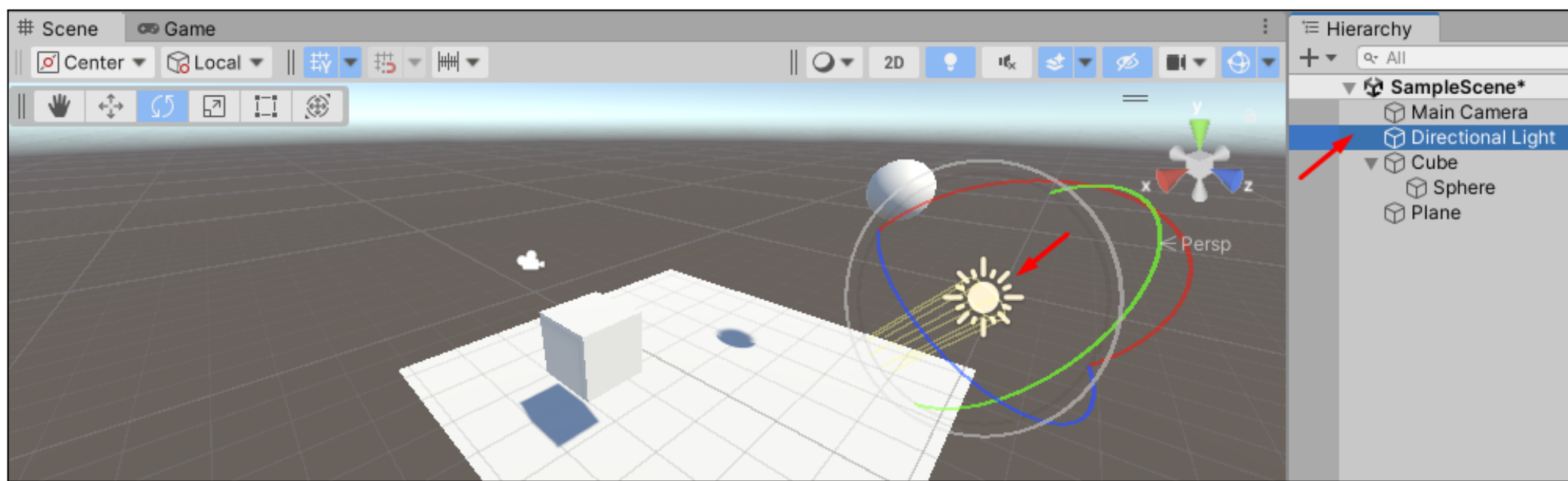


С активной опцией **[Local]** при повороте объектов вместе с ним поворачиваются индикаторы вращения, смещения, масштабирования. При переключении на опцию **[Global]** маркеры переходят на постоянную ориентацию в пространстве по осям **X, Y, Z**:



Directional Light - базовый источник прямого освещения на сцене, создаваемый автоматически для каждой новой сцены.

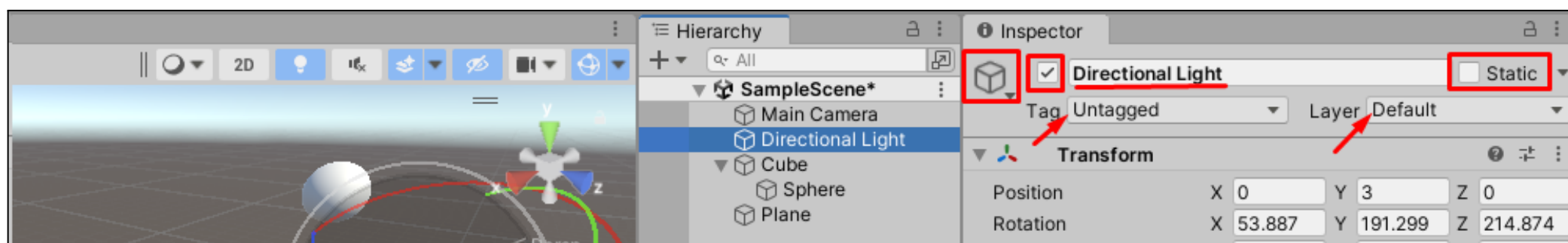
По умолчанию он имитирует солнечный свет, и если его покрутить, то можно получить тени от объектов для разного времени суток.



[Inspector]

✂ Свойства объекта

Свойства объекта отображены в окне [Inspector].



[Имя объекта]

Имя можно менять без ограничений, может повторяться у нескольких объектов.

[Активность]

В активном состоянии объект функционирует как положено, в выключенном не реагирует на окружающий мир и становится невидимым.

[Static]

Для оптимизации игры. Статичные объекты обрабатываются только при запуске сцены.

[Маркеры]

Иконка помогает быстрее находить нужный объект на сцене, особо полезно для невидимых объектов - точек спавна и т.п.

[Tag]

Теги помогают группировать объекты по смысловой нагрузке. Например, **Player** - все, что касается игрока.

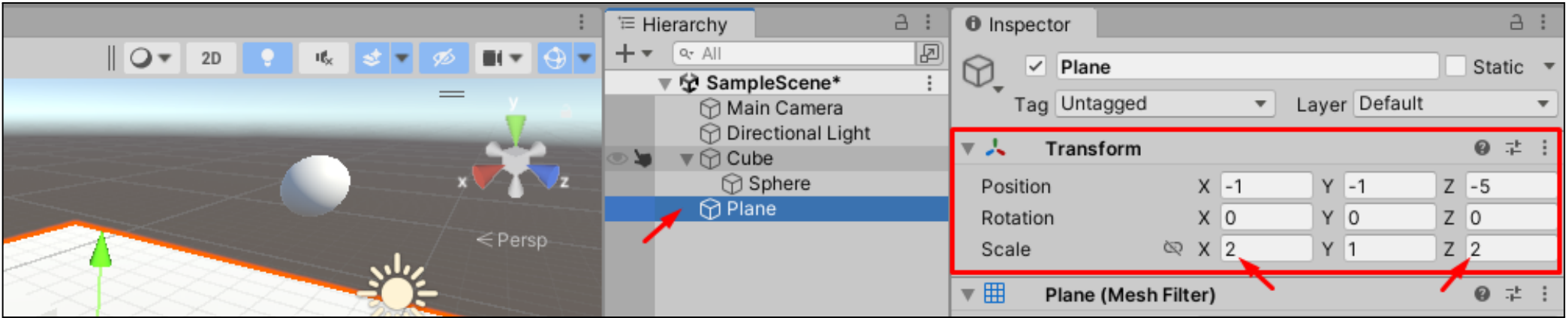
[Layer]

К какому слою относится выбранный объект. Слои нужны для настройки того, какие объект могут взаимодействовать друг с другом с помощью физики. Можно использовать готовые слои или создавать собственные. Какие слои отображать указывается в настройках камеры.

❌ **Компоненты объекта**

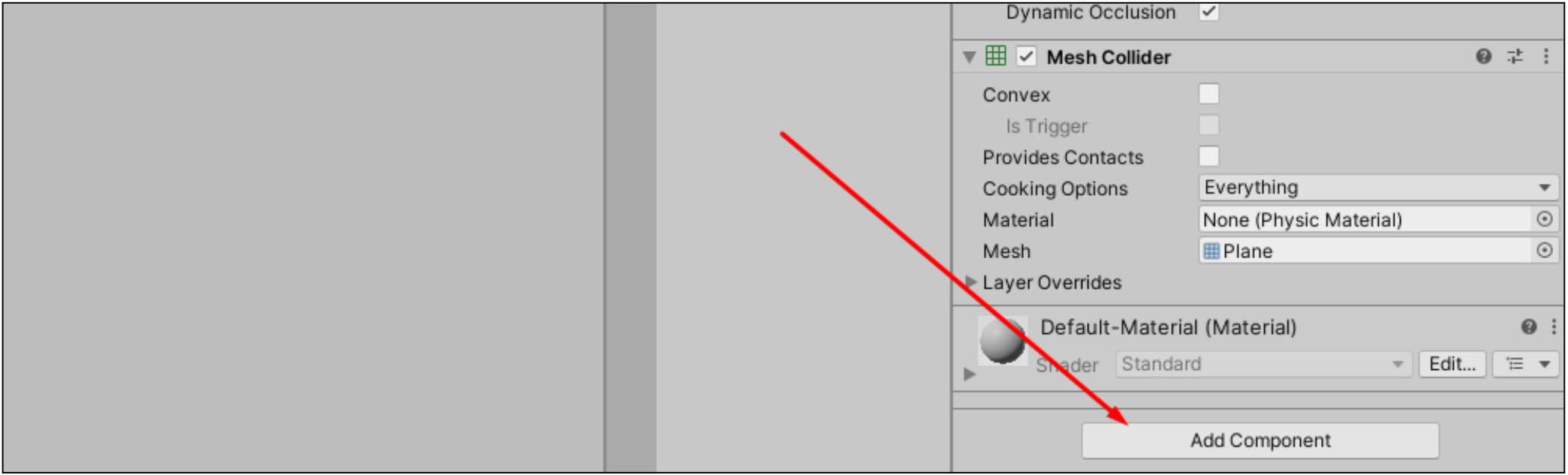
Game Object приобретает все необходимые **свойства** когда ему добавляют **компонент** с определенным поведением.
Компоненты - части игрового объекта, отвечающие за его поведение в процессе игры - перемещение или реакцию на триггеры.

Компоненты объекта отображены в окне **[Inspector]**. В нем отображаются все компоненты объекта, параметры, а также свойства объекта.
Компонент [Transform] добавляется на объект в момент создания объекта, его нельзя удалить с объекта.
Он отвечает за **положение, поворот, размер** / пропорции относительно начальных параметров.



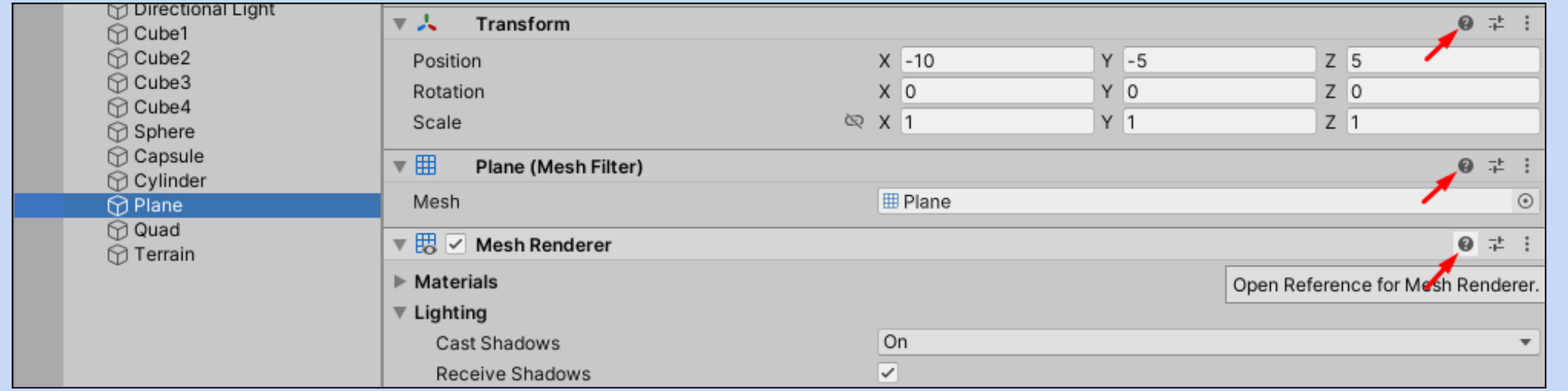
У параметра **Scale** по умолчанию отключена опция (зачеркнут овал) пропорционального увеличения размеров объекта по всем осям.

Новые **компоненты** добавляются объекту по кнопке **[Add Component]** (внизу окна инспектора).



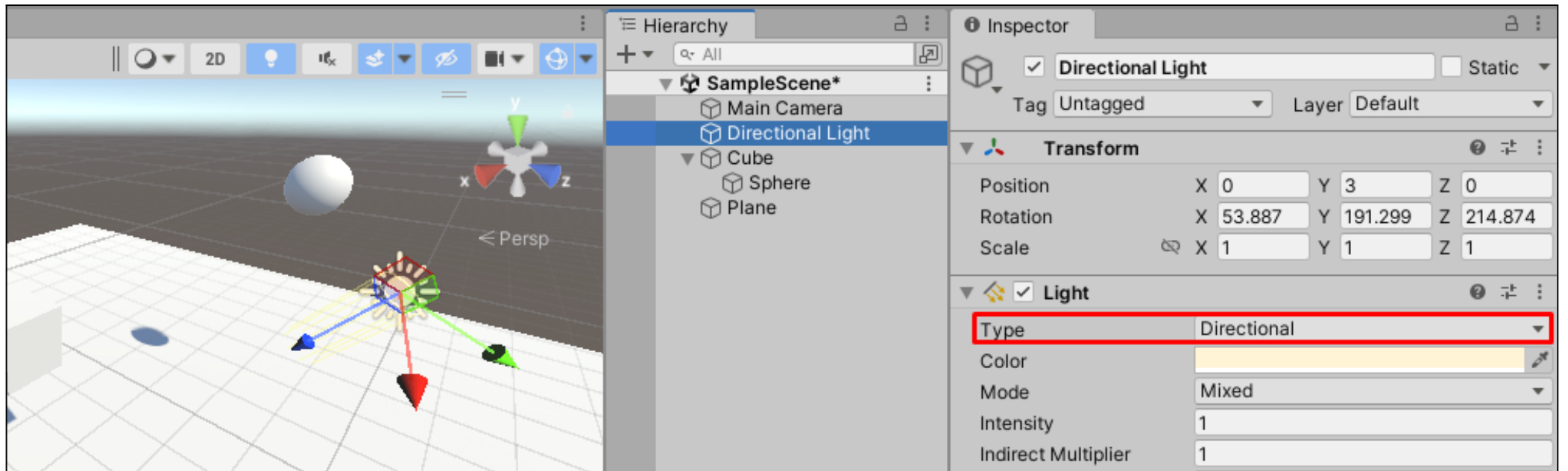
- Компонент **[Mesh Filter]** отвечает за форму объекта.
- Компонент **[Mesh Renderer]** отвечает за визуализацию объекта - без этого компонента объект не сможет быть видимым.
- Компонент **[Material]** вмещает в себя текстуру и шейдер.
- Компонент **[Collider]** границы объекта (хитбокс) - отвечает взаимодействие при столкновении с другими объектами.
- Компонент **[Rigidbody]** добавляет физику для взаимодействия с окружающим миром (падение из-за гравитации, отталкивание).
- Компонент **[Script]** привязывает к объекту сценарий программного кода.
- Компонент **[Audio Source]** добавляет слот для звукового файла - аудио клипа, объект становится источником звука.
- Компонент **[Light]** делает объект источником света.
- Компонент **[Character Controller]** элемент управления, добавляет некоторую физику объекту.
- Компонент **[Animator]** элемент управления анимацией модели.

В руководстве описаны только некоторые элементы Unity и их параметры, с пояснениями по практическому использованию.
Получить справку по любому элементу можно кликнув на знак вопроса, в браузере будет открыта страница с полным описанием:



※ Источники света

Directional Light - источник прямого освещения. Такой источник освещения по умолчанию имитирует солнце - **Directional**:



В свойствах через поле **Type** можно изменить режим работы:

- **Spot** - луч света - прожектор или фонарик, чем дальше объект от света, тем менее он освещен
- **Directional** - направленный свет - солнце
- **Point** - точечный свет - лампа / факел / костер, освещает объекты в определенном радиусе от себя
- **Area (baked only)** - круглая или прямоугольная зона, работает только в режиме заранее "запеченного" света, эта зона испускает свет только в одном направлении - в виде луча из центра фигуры (окно), затенение более мягкое и реалистичное, чем точечное

Color - цвет, по умолчанию устанавливается близкий к солнечному.

Mode - как будет обрабатываться свет источника:

- **Baked** - заранее просчитан и "запечен" - экономия ресурсов во время игры
- **Realtime** - обработка в процессе игры, больше нагрузка на процессор во время игры, игра занимает меньше места на диске.
- **Mixed** - совмещает оба варианта

Intensity - интенсивность освещения.

Indirect Multiplier - яркость отраженного света.

※ Baked

Baked Shadow Radius - угол наклона полутеней, для улучшения мягкости теней.

※ Realtime

Shadow Type - какого типа тени будут отбрасывать объекты:

- **Soft Shadows** - мягкие, гладкие
- **Hard Shadows** - резкие, квадратные, лесенкой
- **No Shadows** - без теней

Realtime Shadows при **Soft Shadows** / **Hard Shadows**

- **Strength** - насколько темными будут тени, при 0 такой режим теней не имеет смысла, так как ресурсы на обработку выделяются
- **Resolution** - разрешение, по умолчанию из настроек проекта (низкое - квадратные и размытые, ультра - качественные сглаженные)
- **Bias** - помогает настроить тени на неровных поверхностях, движок хранит карту теней для каждого пикселя
- **Normal Bias** - помогает настроить тени на неровных поверхностях, движок хранит карту теней для каждого пикселя
- **Near Plane** - минимальное расстояние между источником света и объектом, при котором тень не будет отбрасываться

Cookie - принимает в себя текстуру, эффект которой будет применяться на свет - волнистая текстура - свет под водой.

Size - размер применения текстуры для **Cookie**.

Draw Halo - определяет рисовать ли ореол света вокруг источника света, может имитировать свечение лампочки.

Flare - будут ли блики на свету от источника света, вид бликов задается текстурой, добавляется обычно только солнцу.

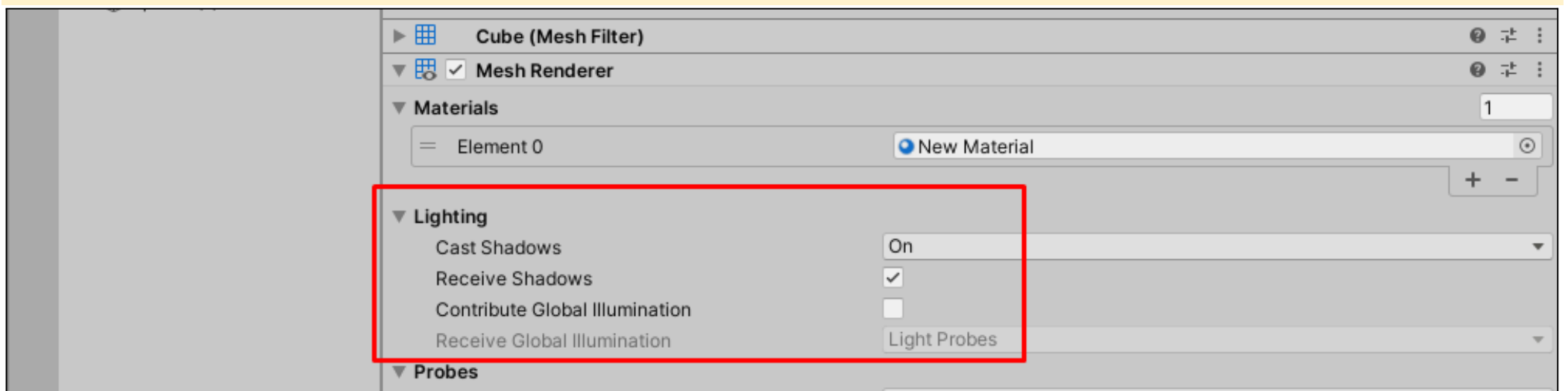
Render Mode - приоритет источника света.

- **Auto** - автоматический режим по умолчанию
- **Important** - важный, свет будет рендерить каждый пиксель отдельно, применять только к ключевым источникам света (фары машины)
- **Not Important** - рендерить не каждый пиксель, а относительно вершин модели

Culling Mask - выбрать слои, на которые будет воздействовать свет.

Чтобы объекты отбрасывали тени, у них должно быть включено свойство **Cast Shadows**.

Чтобы объекты могли принять чужую тень и отобразить ее на своей поверхности, у них должен быть поставлен флаг **Receive Shadows**.

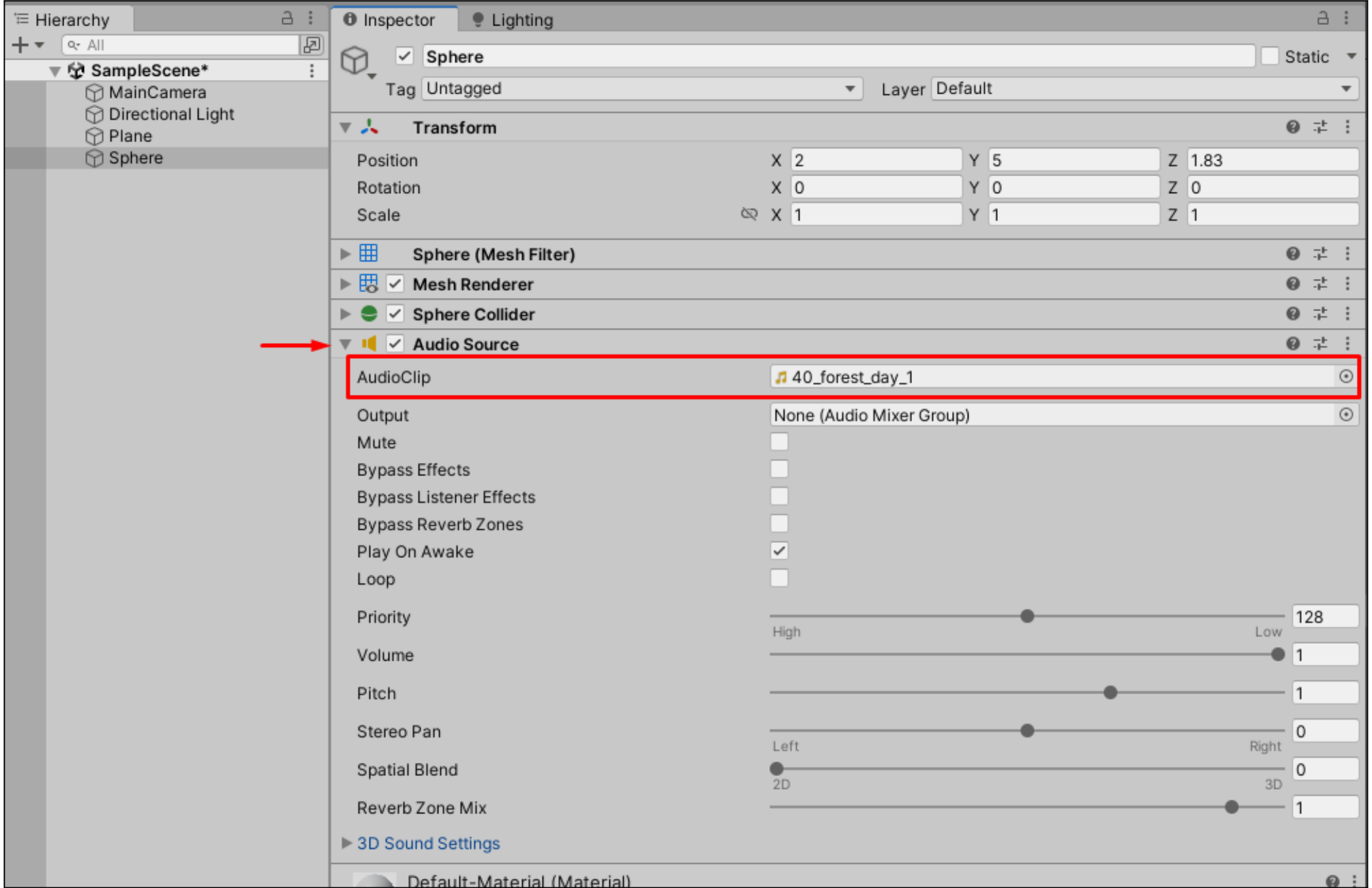


✂ Источники звука

Для имитации эффектов, зависящих от положения источника звука в пространстве, Unity требует чтобы все звуки исходили от объектов. Звуки улавливаются слушателем, подключенным к другому объекту (камере). Затем Unity симулирует эффекты расстояния и положения источника от слушателя, и воспроизводит результат пользователю. Unity не может рассчитывать эхо в зависимости от геометрии сцены, но эхо можно имитировать через добавление фильтров к объектам-источникам звука. Если источник звука быстро перемещается, то можно добавить эффект реверберации (машина в тоннеле). Микшер смешивает звуки разных источников и применяет эффекты.

Любому объекту надо добавить компонент [Audio Source], и он становится источником звука.

Ссылка на звуковой файл задает в поле **AudioClip**:



Output - куда / кому выводить звук, значение по умолчанию вполне справляется с обычной ситуацией.

Mute - звуковой файл проигрывается, но сам звук заглушен.

Bypass Effects - не применять эффекты, которые должны менять применяться к источникам звука.

Bypass Listener Effects - включить / выключить эффекты, применяемые к получателю звука.

Bypass Reverb Zones - включить / выключить зоны реверберации (реверберация - отражение звука).

Play On Awake - звук начинает воспроизводиться сразу после загрузки сцены, если отключено, то запускать воспроизведение надо командой из скрипта через метод **Play()**.

Loop - бесконечный повтор.

Priority - приоритет источника звука (0 самый важный, 256 наименее важный), 0 - чтобы избежать случайного переключения музыки.

Volume - насколько сохраняется громкость при удалении слушателя от источника звука на 1 метр.

Pitch - замедление / ускорение воспроизведения, 1 - нормальная скорость, при отклонении высота тона искажается.

Stereo Pan - как звук распределяется между левым и правым каналами стерео выхода.

Spatial Blend - как должен восприниматься звук - в 2D / 3D, в 3D источник звука учитывает положение слушателя в пространстве.

Reverb Zone Mix - количество звука, направляемого на реверберацию - для получения эффекта ближнего / удаленного источника звука.

Обычно сжатое аудио (Compression Format у аудио-файла в проекте) - **Vorbis** лучше использовать для длинных файлов - фоновой музыки или озвученных диалогов. Для коротких звуков - выстрелов - лучше использовать не сжатый **PCM**.

Поддерживаемые звуковые форматы:

.mp3

.ogg

.wav

.aiff / .aif

.mod

.it

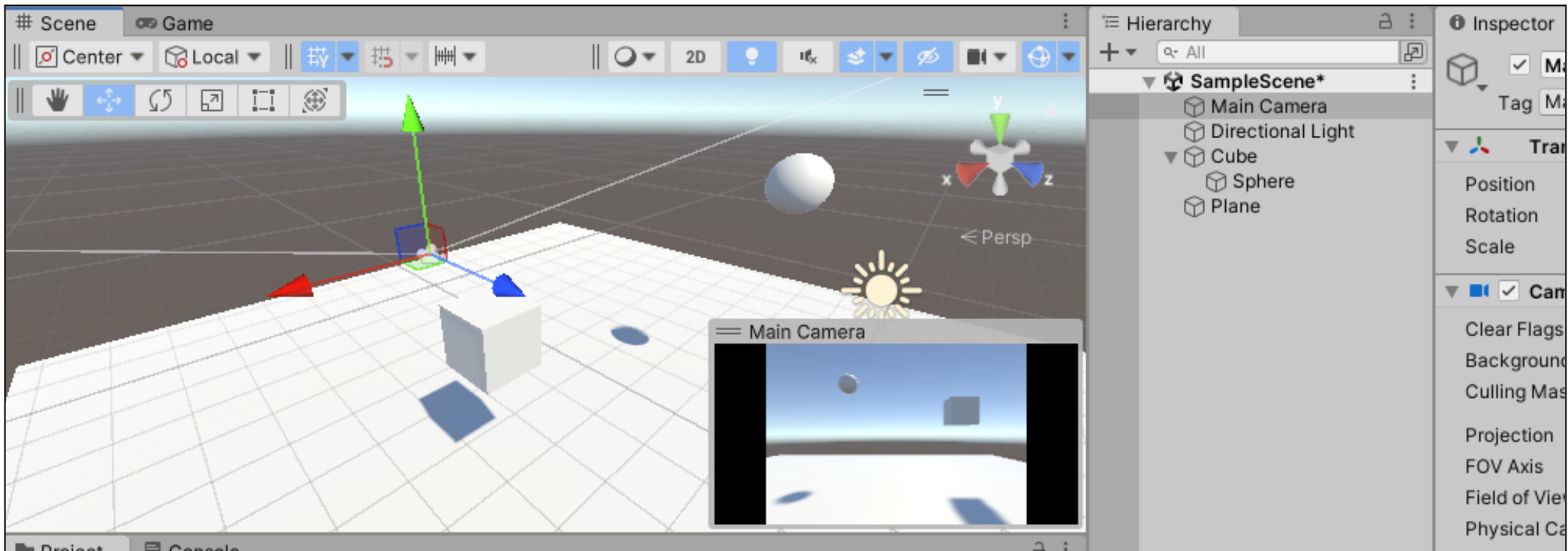
.s3m

.xm

Камера

Обычная **камера игрока** добавляется автоматически при создании **сцены**.

При выделении **камеры** на **сцене** отображаются границы обзора **камеры** - то, что увидит **игрок**, и окно предпросмотра.



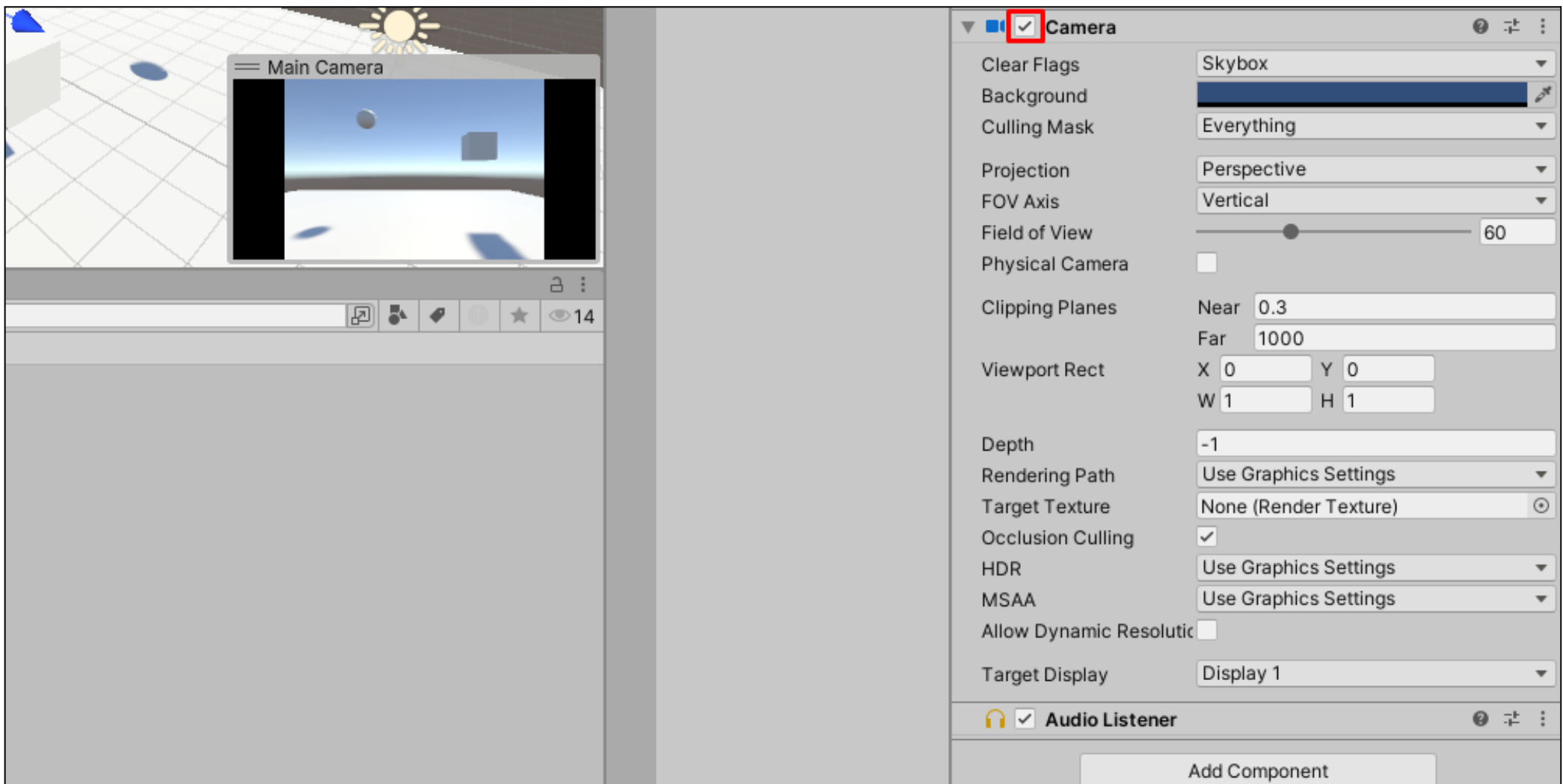
Посмотреть на сцену глазами **игрока** (через камеру игрока) можно в окне **[Game]**.

- Закрепить камеру за персонажем = игра с видом от третьего лица.
- Прикрепить камеру к голове персонажа = игра с видом от первого лица.
- Неподвижная камера = карточные игры.
- Перемещающаяся камера = стратегии.

✂ Настройка камеры

Камеру игрока можно настроить как и любой другой **компонент** в окне **[Inspector]**.

Флаг активности камеры - включает / выключает ее - управляя изображением в окне **[Game]**, но не выключает окно предпросмотра сцены.



Clear Flags - что будет отрисовано там, где нет объектов:

- Skybox** - небо
- Solid Color** - однородный цвет
- Depth only** - то что было на прошлом кадре, но обновляя глубину картинки
- Don't clear** - камера будет целиком оставлять изображение с прошлого кадра (создает эффект размытия, обычно не используется)

Background - цвет заливки для **Solid Color**.

Culling Mask - выбрать слои, которые камера будет показывать игроку. В шутерах часто решают проблему отображения оружия. Оружие с руками рендерят отдельной камерой и накладывают поверх изображения от основной камеры. Так оружие не проходит сквозь стены.

Projection - режимы работы у них разные связанные параметры:

- [Perspective]** - перспективная камера с объемным отображением
- [Orthographic]** - ортография - камера для плоских изображений

✂ **Perspective** - перспективная камера с объемным отображением.

FOV Axis - горизонтальная или вертикальная ориентация.

Field of View - угол обзора.

Physical Camera - эмуляция объектива у камеры.

※ **Orthographic** - ортография - камера для плоских изображений.

Size - общий размер прямоугольной области, которую захватывает камера.

Clipping Planes - отвечает за дальнюю и ближнюю плоскость камеры, насколько дальние и ближние объекты будут отрисовываться:

- **Near** - расстояние ближняя плоскость
- **Far** - расстояние до дальней плоскости

Viewport Rect - позволяет менять положение картинки на экране:

- **X, Y** - смещение влево / вправо, вверх / вниз
- **W, H** - ширина и высота видимого в камере пространства

С помощью этих параметров можно сделать черные полосы как в кино или мини радар.

Deapth (глубина) - порядок наложения изображения с нескольких камер, или очередь отрисовки. Можно совмещать с **Culling Mask**.

Пример: сначала выводится изображение с камеры -1, затем поверх изображение с камеры 0, потом 1 и т.д.

Rendering Path - какой вид рендера будет использовать камера:

- **Use graphics settings** использует настройки, которые стоят в проекте - **Forward**
- **Forward** - оптимально в 99% случаев
- Deferred
- Legacy Vertex Lit

Target Texture - сюда можно вставить текстуру, на которую будет выводиться изображение от этой камеры. А потом эту текстуру можно наложить на различные объекты. Так можно создать реалистичные мини карты, оптические прицелы, комнату охраны с мониторами, на которых будут двигающиеся картинки от охранных камер, зеркало заднего вида.

Occlusion culling - камера не будет обрабатывать объекты, закрытые другими объектами. Данные об объектах, которые не надо рендерить хранятся в ОЗУ видеокарты. Т.к. на мобилках и планшетах нет графического процессора, то вся нагрузка идет на центральный процессор (и там сожрет ресурсы), поэтому включать этот параметр для мобильных устройств не стоит.

HDR - использовать цвета широкого диапазона.

MSAA - не стоит использовать для дешевой камеры наблюдения или создания миникарты.

Allow Dynamic Resolution - динамическое разрешение, движок может понизить его при высокой нагрузке.

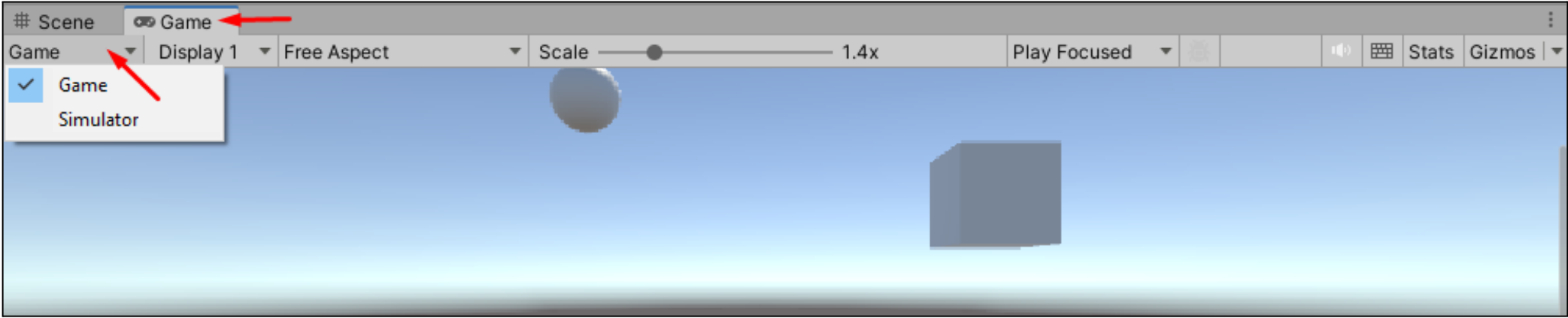
Target Display - на какой экран выводит изображение камера.

[Game]

В окно [Game] выводится то, что увидит игрок, когда объекты сцены попадают в поле зрения его камеры.

У окна есть параметры, и значения первого параметра определяют доступ к другим настройкам.

- [Game] - имитация экрана с определенным разрешением
- [Simulation] - симуляция экрана какого-либо устройства



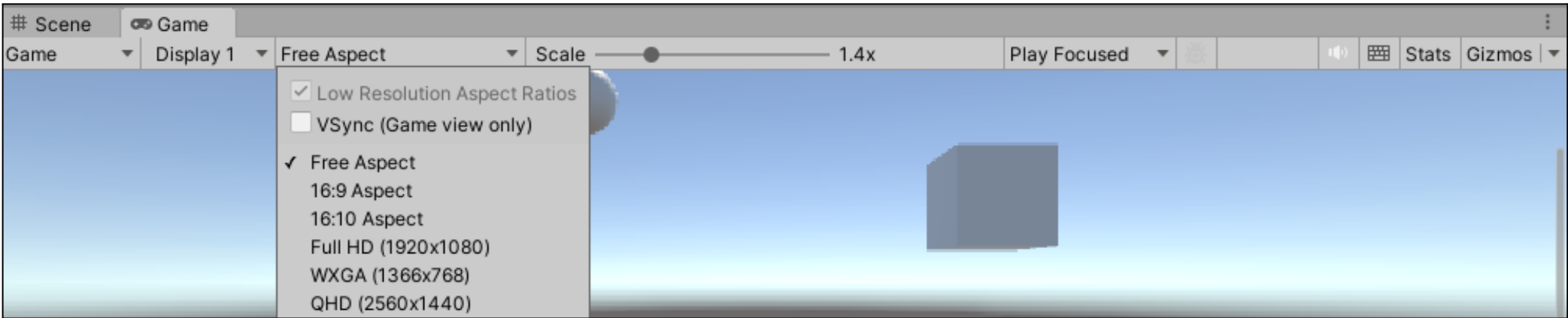
✂ Game

[Display 1] - **выбор дисплея** для отображения изображения (всего поддерживается 8).

Разным камерам можно задать вывод на разные дисплеи. Можно расставить несколько камер в разных точках под разными углами, задать им разные дисплеи, и переключаясь между камерами выбрать наиболее интересный ракурс.

[Free Aspect] - **выбор разрешения**. Это помогает при тестировании UI. Видно то, что увидит игрок на мониторах разного формата.

Если нужного разрешения или соотношения сторон нет, то можно его создать.



[Scale] - **слайдер размера экрана** - приближение / отдаление, удобно проверять есть ли визуальные дефекты.

[Play Mode] - **выбор поведения** при тестировании игры в редакторе. При включении PlayMode окно генерирует три типа поведения:

- **Play Focused** (по умолчанию) - при запуске фокус переключается на окно [Game], игра считывает нажатие кнопок на клавиатуре
- **Play Maximized** - **Play focused** + окно разворачивается на весь экран
- **Play Unfocused** - запускает тест, не переключая фокус на окно игры, это может быть полезно при проверке запуска катсцены, ролика, при котором ввод с клавиатуры не нужен

[Mute Audio] - отключить звук.

[Unity Shortcuts] - отключать считывание сочетаний клавиш Unity по время проверки игры.

[Stats] - включить / выключить окно статистики. Самым нужным в нем является FPS.

[Gizmos] - отображение границ объектов, надо чтобы отследить техническое отображение объектов, используется редко.

✂ Simulation

Симуляция экрана какого-либо устройства.

[Scale] - **слайдер размера экрана** - приближение / отдаление, удобно проверять есть ли визуальные дефекты.

[Fit To Screen] - **подогнать размер** эмулируемого девайса под размер окна.

[Rotate] - повернуть устройство (его дисплей).

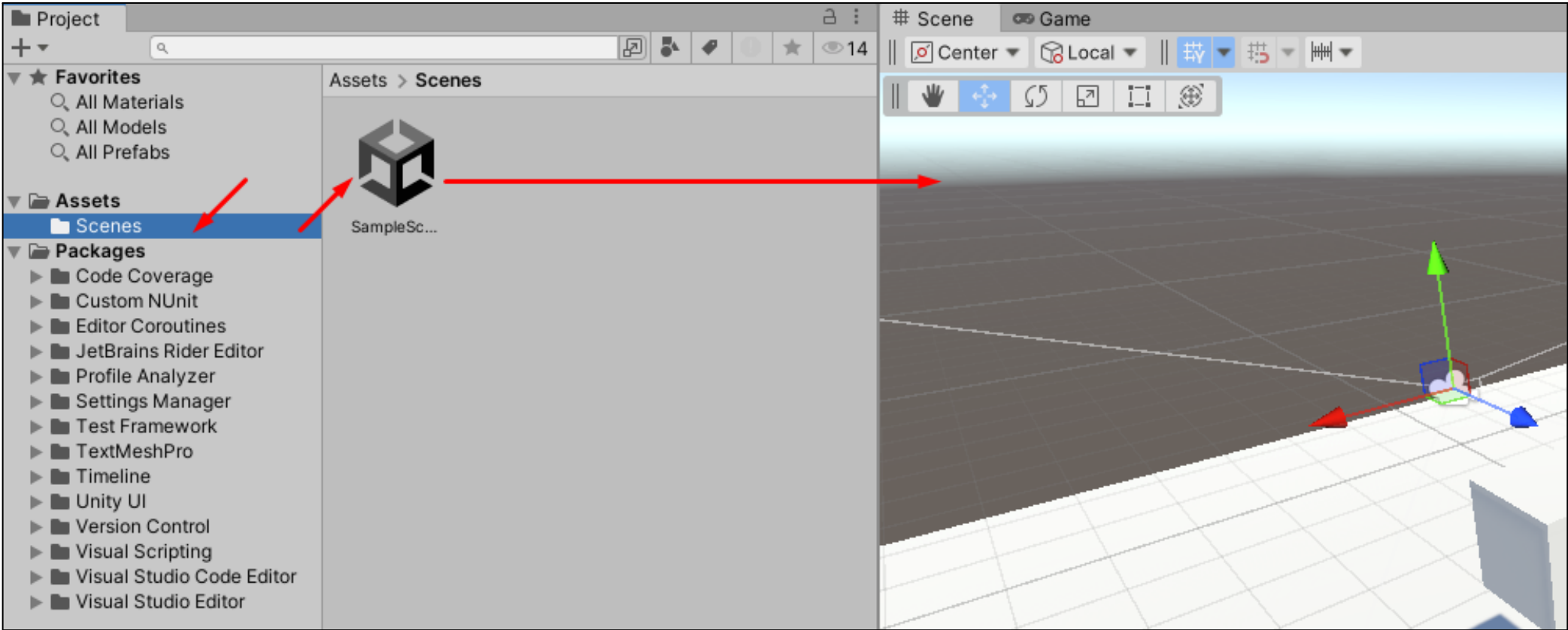
[Safe Area] - отобразить безопасную зону - ту часть экрана мобильного устройства, которую не перегораживает фронтальная камера.

[Play Mode] - **выбор поведения** аналогично как в режиме Game, но режим по умолчанию **Play Unfocused**.

[Control Panel] - **панель управления** позволяет выбрать язык эмулируемого устройства, выход в интернет.

[Project]

В окне проекта отображены все ресурсы проекта по аналогии с проводником Windows. Здесь окно перемещено, в нем файл текущей сцены.



✂ Типы ресурсов

3D модели - файлы **FBX** и другие (Unity подходят **DAE**, **FBX**, **OBJ**). **MESH** - модельная сетка 3D модели. Модель персонажа содержит все скины персонажа (за отображение конкретной части модели отвечает флаг активности нужного элемента).

Модель может хранить в себе анимации, которые она может воспроизводить. Типы анимации:

- **Legacy** - старый формат, который нужен в очень старых проектах и при экстремальной оптимизации
- **Generic** - применяется в основном к объектам и не гуманоидным существам, иногда к гуманоидным
- **Humanoid** - применяется к гуманоидным персонажам. Анимации работают в связке с аватарами

Префабы - способ создания шаблонного объекта из нескольких для повторного использования. При изменении параметра префаба все копии объектов на сценах также получают изменения. **Prefab Variant** наследует свойства и компоненты от оригинала, но может сохранять изменения как собственные отличия. Команда **Unpack** в контекстном меню префаба в сцене конвертирует префаб в обычный объект, который уже не будет связан с оригинальным источником.

Для создания префаба надо перетащить составной объект со сцены (или из окна иерархии) в окно проекта в нужную папку.

Материалы - определяют параметры и правила отображения объекта - способ визуализации поверхности. В том числе ссылки на используемые текстуры, информацию о мозаике, цветовые оттенки и прочее. Доступные параметры для материала зависят от того, какой шейдер используется в нем. Материал передается ссылкой в модель объекта. Модель может иметь как один, так и несколько материалов - по одному на каждый Mesh или участок модели.

Текстуры (спрайты) - растровые изображения - картинки в форматах **PNG**, **JPEG**, **PSD**. Подгружаем в проект нужное изображение, меняем настройки картинки, применяем картинку к материалу (материал получает ссылку на текстуру). Шейдер материала может использовать текстуру при расчете цвета поверхности объекта. В дополнение к основному цвету (альбедо) поверхности материала, текстуры могут предоставлять многие другие параметры поверхности материала - отражательная способность, шероховатость.

Спрайты можно использовать как самостоятельные игровые объекты - из них состоят все 2D игры.

Спрайты применяются в UI - окна и кнопки, но там используется компонент **Image** вместо **Sprite Renderer**.

Шейдеры - программа, которая выполняется на видеокарте, и определяет как будет рендериться объект (отрисовываться его поверхность на экране). Шейдер содержит математические вычисления и алгоритмы для вычисления цвета каждого визуального пикселя на основе входного освещения и конфигурации материала. Какие настройки имеет материал, будет ли на нем работать свет, будет ли он использовать текстуры и прочее - все это определяет шейдер, который используется в материале. Шейдер задает одну или несколько переменных-текстур, которые он ожидает на вход (будет использовать), а Инспектор позволяет назначить эти такие текстуры как входные параметры.

Стандартный шейдер автоматически добавляется новому материалу. Он использует сложные расчеты и хорошо справляется с отрисовкой символов, декораций, окружения, сплошных (камень, дерево, металл) и прозрачных объектов (стекло), твердых и мягких поверхностей.

- **FX** - водные и световые спецэффекты
- **GUI** - отображение графических интерфейсов пользователя, для текста
- **Mobile** - упрощенные шейдеры для мобильных устройств
- **Nature** - деревья и земная поверхность
- **Particles** - эффекты системы частиц
- **Skybox** - для рендеринга фоновой среды (неба)
- **Sprites** - для 2D
- **UI** - интерфейс пользователя
- **Unlit** - для рендеринга, который полностью обходит свет и тени
- **VR** - виртуальная реальность
- **Autodesk Interactive** - для обеспечения качественной визуализации 3D-объектов, созданных в Autodesk Maya или 3ds Max
- **Standard** - по умолчанию
- **Standard (Specular setup)** - для материалов с имитацией отражения света - металлических и полированных материалов

Стандартный шейдер заменяет большой список старых шейдеров (Legacy). Фактически стандартный шейдер это набор множества других, шейдеров, каждый из которых используется по ситуации. Использование того или иного шейдера зависит от того, какие опции были задействованы у стандартного шейдера. Это удобный способ оптимизации нагрузки.

Bumped Diffuse - текстура с рельефом без бликов.

Bumped Specular - текстура с рельефом с бликами.

Diffuse - просто текстура без рельефности и бликов.

Аудио - звуки и фоновая музыка, озвучка диалогов - подходят обычные файлы формата **MP3** и некоторые другие. Чтобы с ними работать нужны два компонента:

- **Audio Source** - определяет где воспроизводится звук, фоновая музыка и звуки, либо воспроизводится в определенной точке
- **Audio Listener** - передает звук в наушники, **объект с этим компонентом должен быть только один на сцене (камера игрока)**

Анимации - анимационные клипы, по схемам которых персонажи будут двигаться в зависимости от условий активации анимаций.

Скрипты - файлы с программным кодом. Писать собственные скрипты надо для реализации сложной логики взаимодействия между объектами. Средства визуального программирования позволяют сделать не все.

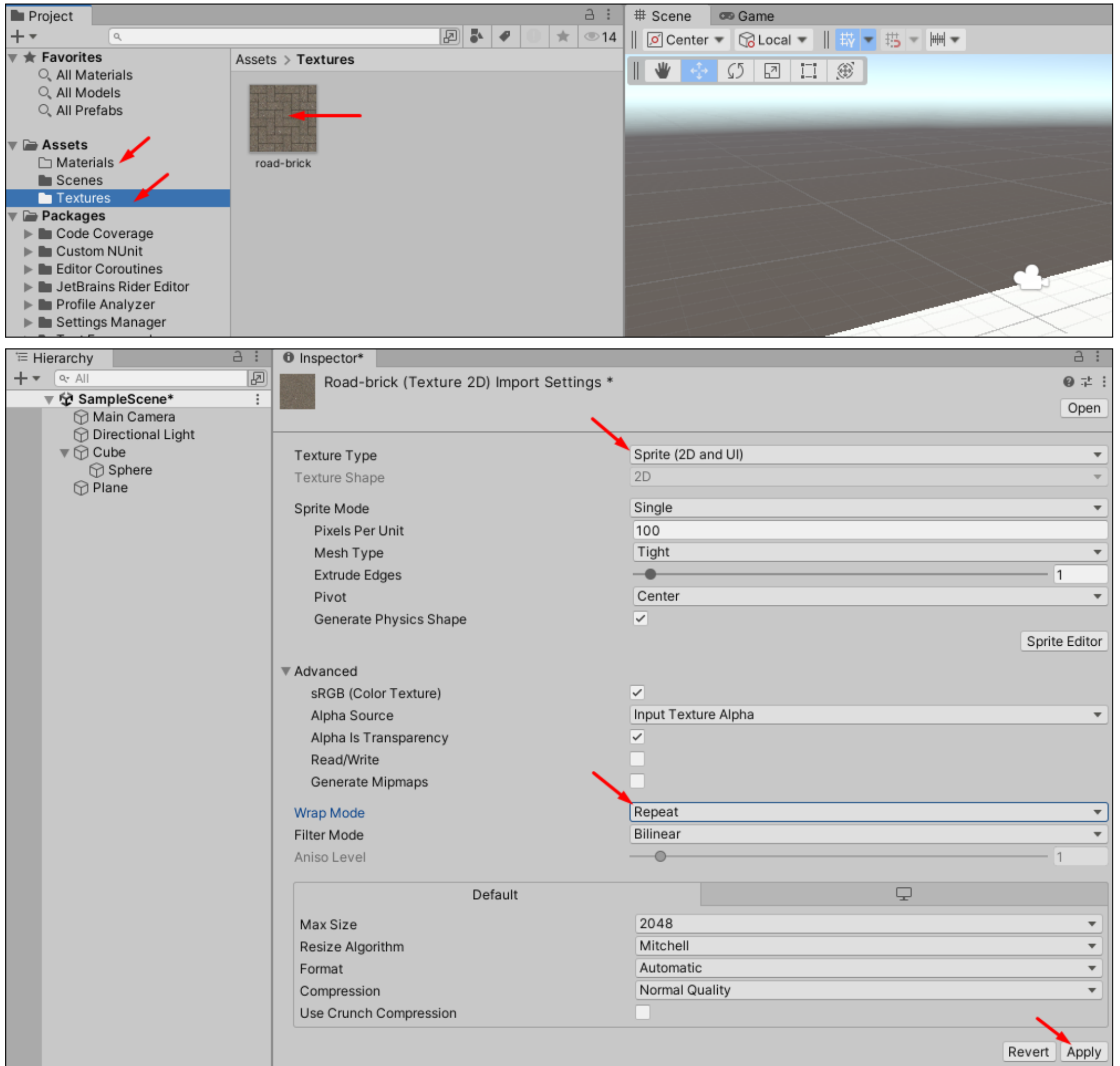
✂ Создание материала

Добавляем в структуру проекта в разделе **Assets** новые папки (Folder): **Materials** и **Textures**.

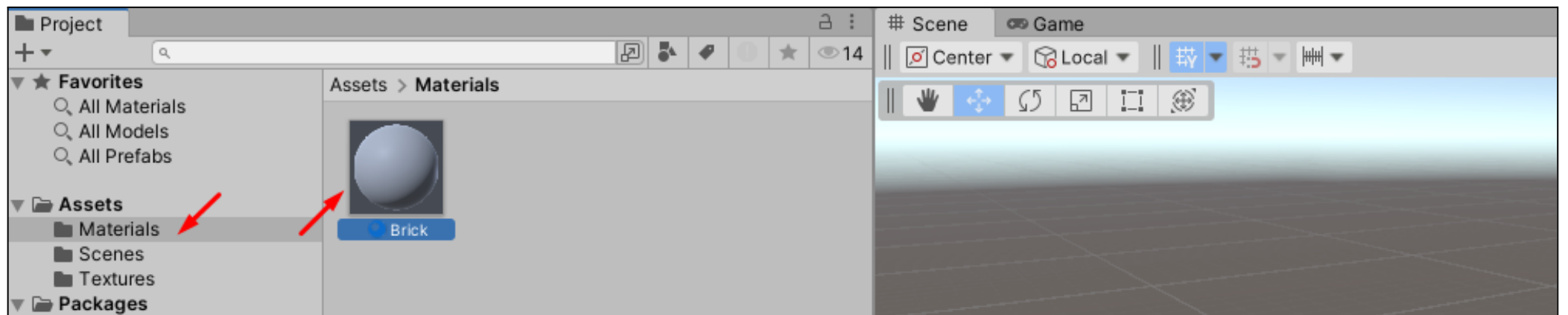
Скачиваем из интернета любое изображение для поверхности - плитка, доски и т.д.

Картинку перетаскиваем в папку текстур и меняем ее свойства в окне [**Inspector**].

Для сохранения ее изменений кнопка [**Apply**].



В папке материалов создаем новый материал - контекстное меню / **Create** / **Material**.



Открыть окно выбора текстуры - ткнуть в мелкий кружок, либо перетащить текстуру на материал.

Опция **Metallic** придает материалу металлический отблеск, а регулировка параметра **Smoothness** придает эффект гладкости и блеска.

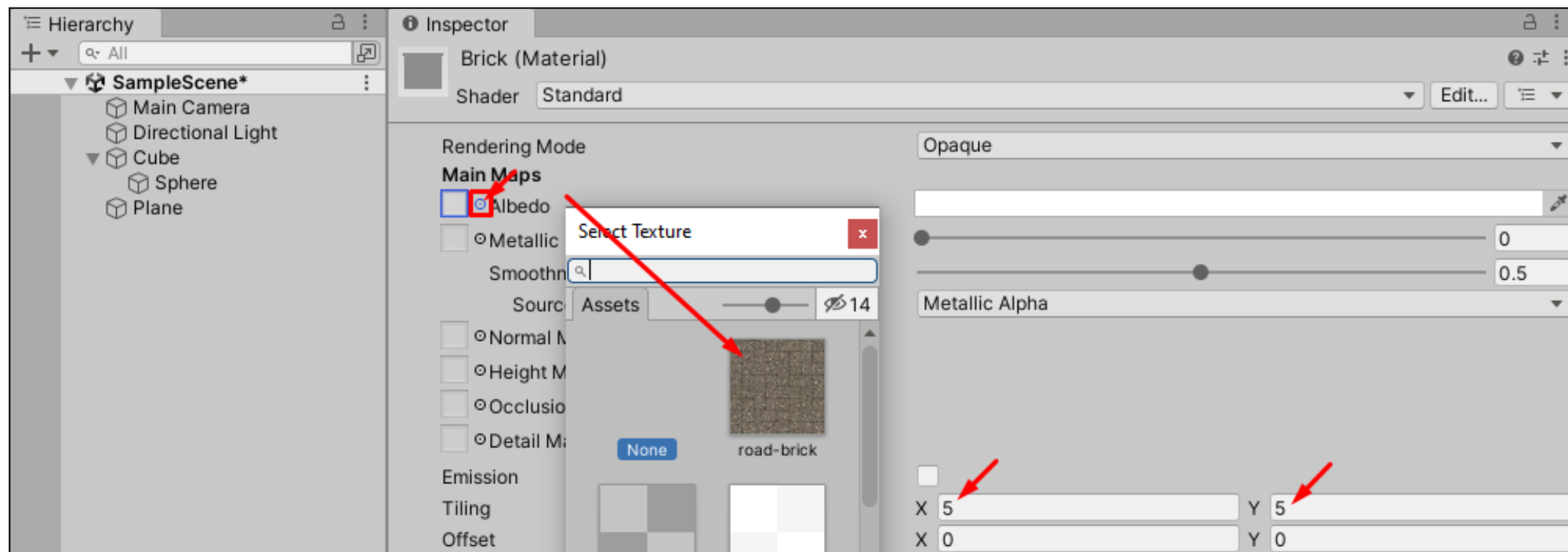
Опция **Normal Map** - карта неровностей на поверхности материала, имитация впадин (поглощение света) через рисунок.

Опция **Height Map** - карта высот на поверхности материала, имитация выступающих частей через рисунок.

Tiling - количество повторений текстуры, надо подбирать под каждый конкретный случай.

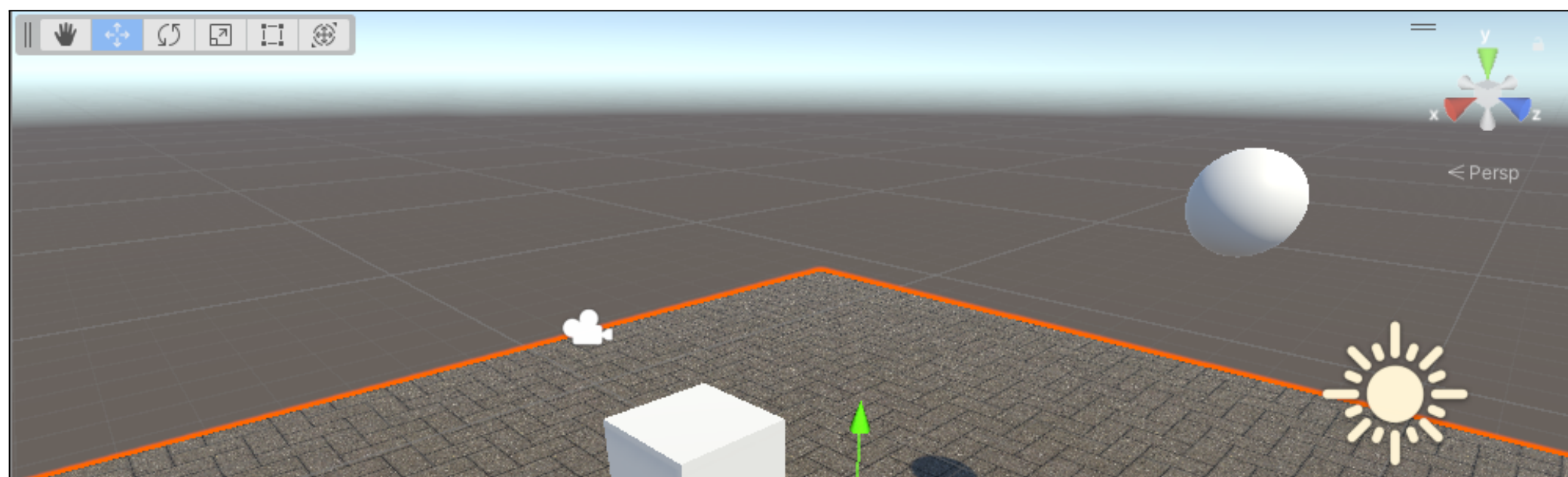
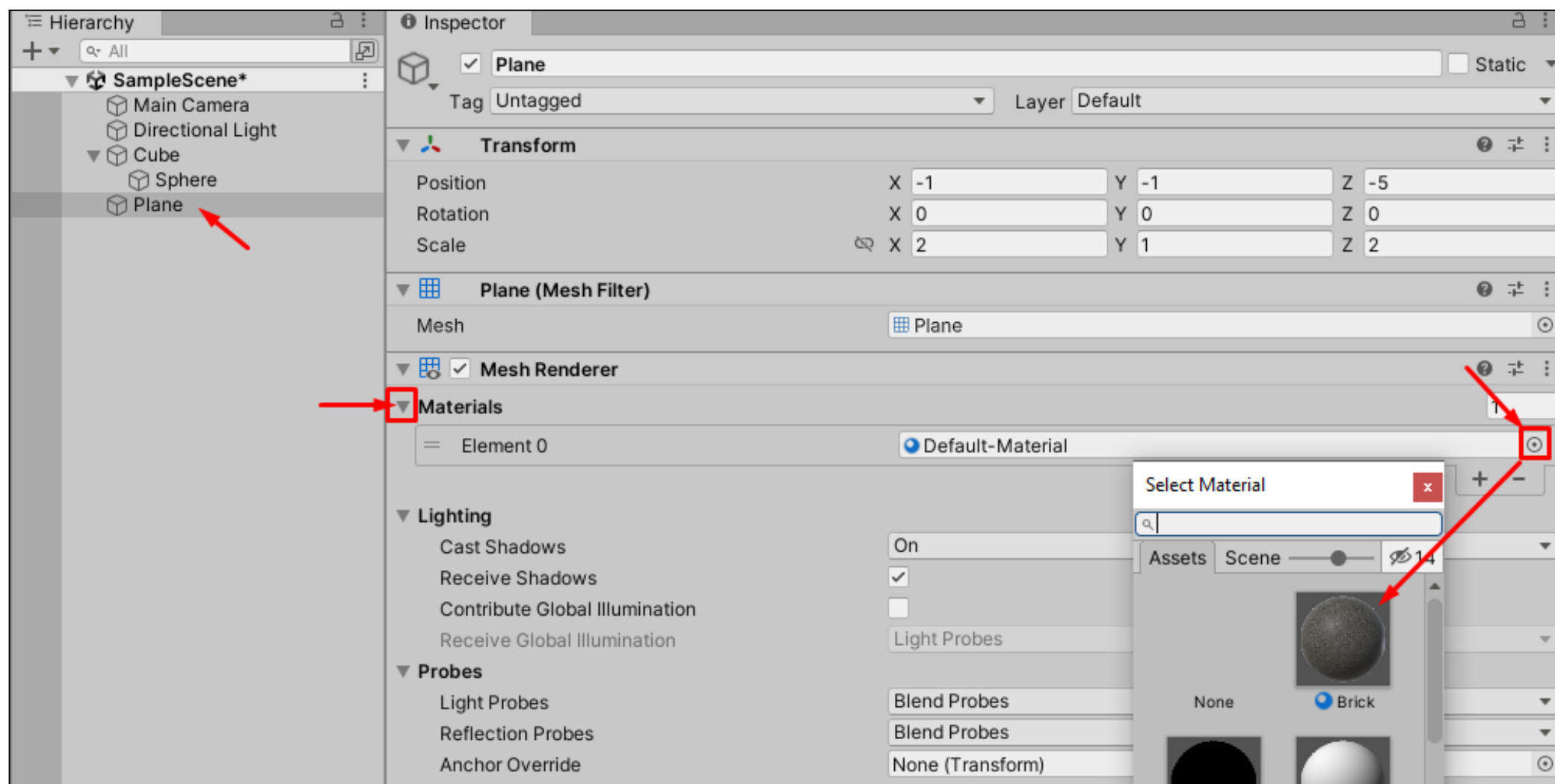
Режимы рендеринга **Rendering Mode**:

- **Opaque** - без прозрачности, по умолчанию
- **Cutout** - жесткие края между прозрачными и непрозрачными материалами, подходит для травы, листьев, ткани
- **Fade** - значения прозрачности затушевывать, убирая блики и отражения, но прозрачность остается
- **Transparent** - части объекта прозрачны, от них отражается свет, появляются блики



Для применения настроенного материала к объекту достаточно перетащить материал на нужный объект в окне [**Scene**].

Либо выбрать объект в окне [**Inspector**], развернуть свойство **Materials**, снова через малый круг раскрыть список, и выбрать материал. Получается уложенный плиткой пол.

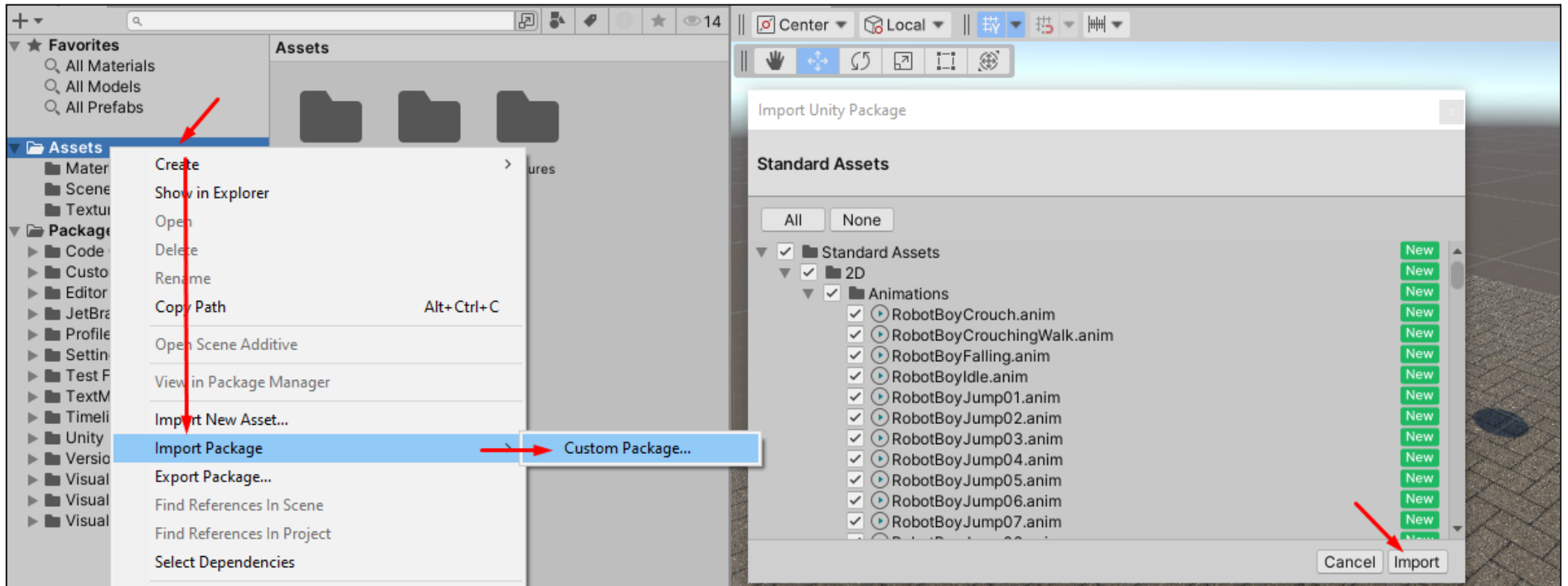


✂ Импорт ресурсов

На сайте **издательства Наука и техника** в разделе **Материалы к книгам** из блока **Компьютерная литература** скачать и распаковать архив:
[21. UNITY. Полное руководство, 2-е изд._Корнилов А. В.](#)
<https://nit.com.ru/page/materialy-k-knigam-3>

При импорте ресурсов из сторонних источников можно скачать программный код с вирусами / майнерами и т.д., поэтому Unity будет предупреждать об опасности, если он запущен с правами администратора. Стандартный источник ресурсов – официальный магазин.

Импорт ресурсов из пакетов – в контекстном меню папки **Assets** или в меню редактора **Assets / Import package / Custom package**. Найти среди распакованных ресурсов в папке Assets файл **Standard Assets.unitypackage**, подтвердить импорт ресурсов – кнопка [**Import**].



Так как пакет не новый, то при импорте Unity сообщит о найденных ошибках в ресурсах, и сам поправит проблемные детали. Если Unity не может исправить ошибку, то импортируемый объект может потерять свойства, текстуры и т.д. Некоторым объектам может потребоваться починка перед использованием.

При перетаскивании файла с диска в окно области [**Project**] файл КОПИРУЕТСЯ (а не перемещается) в папку **Assets**, и созданная копия становится доступной для использования.

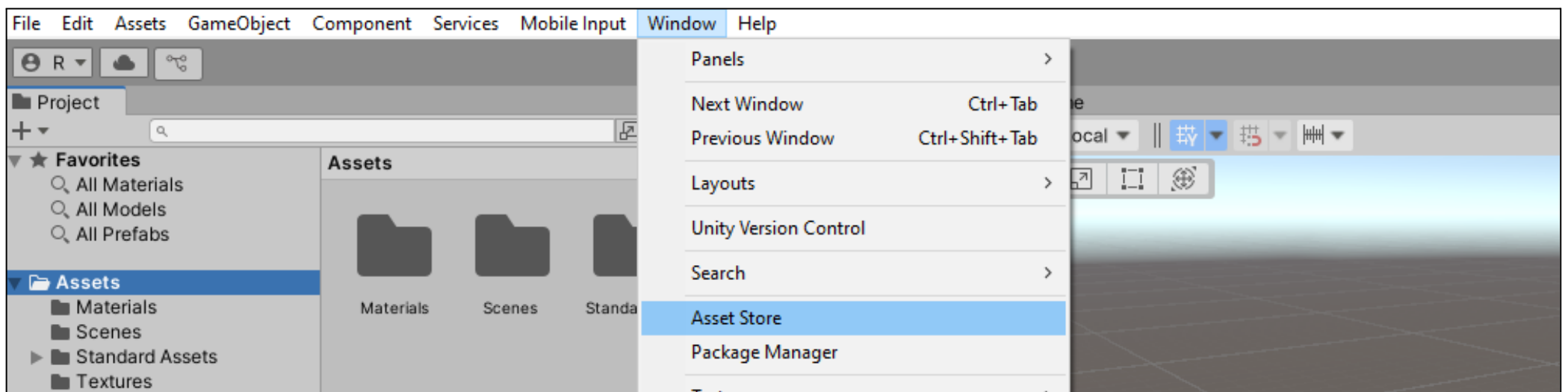
В большинстве случаев элементы в области [**Project**] это файлы на диске компьютера, поэтому при их удалении в окне проекта эти файлы перемещаются в корзину, но оригинал, из которого они были скопированы остается.

Если какой-либо файл из аудио или графического редактора был создан / сохранен сразу в папку проекта **Assets**, то при удалении в окне проекта этого файла-ресурса, будет удален единственный экземпляр этого ресурса!

Удаленные файлы можно найти в корзине, если они удалялись кнопкой [**Delete**] без модификатора [**Shift**].

✂ Магазин ассетов

Открыть официальный магазин ассетов можно по ссылке из Unity, либо сразу по прямой ссылке: <https://assetstore.unity.com/>



Чтобы добавлять в свой профиль ассеты необходимо залогиниться под учетной записью, созданной при установке Unity.

Далее с помощью фильтров найти любой бесплатный ассет (**Free**) и добавить его в свой профиль – [**Add to My Assets**], [**Accept**].

<https://assetstore.unity.com/packages/3d/environments/lowpoly-environment-nature-free-medieval-fantasy-series-187052>

<https://assetstore.unity.com/packages/3d/vegetation/environment-pack-free-forest-sample-168396>

3D2DAdd-OnsAudioAIDecentralizationEssentialsTemplatesToolsVFXSale

Over 11,000 five-star assets

Rated by 85,000+ customers

Supported by 100,000+ forum members

Every asset moderated by Unity

Home > 3D > Environments > Lowpoly Environment - Nature Free - MEDIEVAL FANTASY SERIES



Lowpoly Environment - Nature Free - MEDIEVAL FANTASY SERIES

Polytope Studio★★★★★(94) | ♥(8224)

FREE

2195 views in the past week

Add to My Assets

License agreementStandard Unity Asset Store EULA

License typeExtension Asset

File size52.1 MB

Latest version1.1.1

Latest release dateJan 2, 2025

Original Unity version2021.3.40

SupportVisit site

В пакетном менеджере найти нужный ассет и запустить его загрузку [Download], после загрузки импортировать [Import].

Unity Asset Store

The Asset Store has moved

You can visit the Asset Store website from 2020.1 onwards, and also import your purchases from the Package Manager.

Search online

Always open in browser from menu

Importing your assets from the Package Manager

To download or view your purchases, click the Package Manager button below.

Open Package Manager

Package Manager

Aluminum Can "Standard" 1.0

Gridmap Maker 1.1.0

HexMap Tools 1.1.1

Leonardo.Ai 1.1.2

Lowpoly Modular Armors - Free - MEDIEVAL FANTASY SERIES 1.2.1

Low Poly RPG Fantasy Weapons Lite 1.0

Low Poly Shaders 1.0.1

Maze Generator V 1.2.2

Lowpoly Environment - Nature Free - MEDIEVAL FANTASY S... 1.1.1

Lowpoly Environment - Nature Free - MEDIEVAL ...

1.1.1 · January 02, 2025

Polytope Studio

View in Asset Store | Publisher Website | Publisher Support

Download

Supported Unity Versions 2021.3.40 or higher

Package Size Size: 52,15 MB (Number of files: 127)

Purchased Date June 11, 2025

Обычный сценарий при работе с ассетами из магазина:

- Создать новый проект
- Импортировать в него интересующий ассет
- Посмотреть содержимое
- Отобрать понравившиеся материалы, модели и прочее, и экспортировать
- Экспортированные элементы импортировать в рабочий проект

✂ Экспорт ресурсов

Экспортировать можно ресурсы, хранящиеся в папках проекта (не из окна сцены). Выбрать нужный файл, в контекстном меню команда [Export Package]. Откроется окно, в котором надо проставить проставить флажки (обычно надо снять флаг со скриптов), и [Export].

При экспорте ресурсов, особенно префабов, нельзя выключать опцию Include dependencies, иначе не все части сборных объектов будут экспортированы – могут потеряться материалы, текстуры и т.д.

✂ Прогулка по сцене

Среди всего в импортированных ресурсах есть анимированная вода:

Assets / Standard Assets / Environment / Water / Water / Prefabs / WaterProDaytime

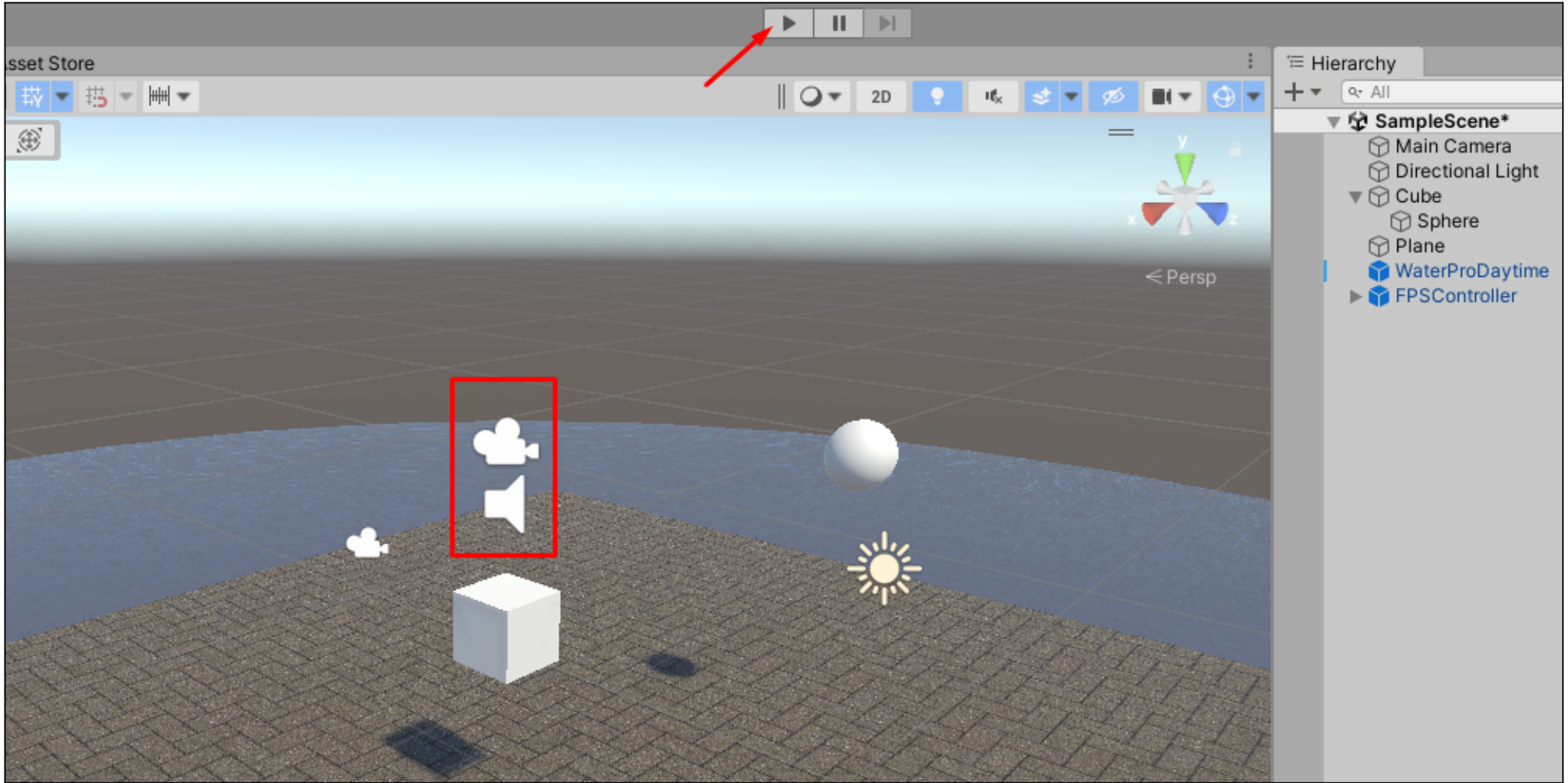
Чтобы создать объект на сцене из импортируемых ресурсов – надо перетащить его из папки-источника на саму сцену, или в список объектов в окне иерархии объектов. Помещенную воду отрегулировать по высоте, чтобы она была ниже пола, и увеличить размер.

Из другого раздела добавить управляемую камеру для прогулки по сцене:

Assets / Standard Assets / Characters / FirstPersonCharacter / Prefabs / FPSController

Такая камера-бродилка позволяет перемещаться по сцене с помощью клавиш – WASD и прыгать – пробел.

Запуск тестовой прогулки по кнопке [Play] (запустить или прервать – Ctrl + P):

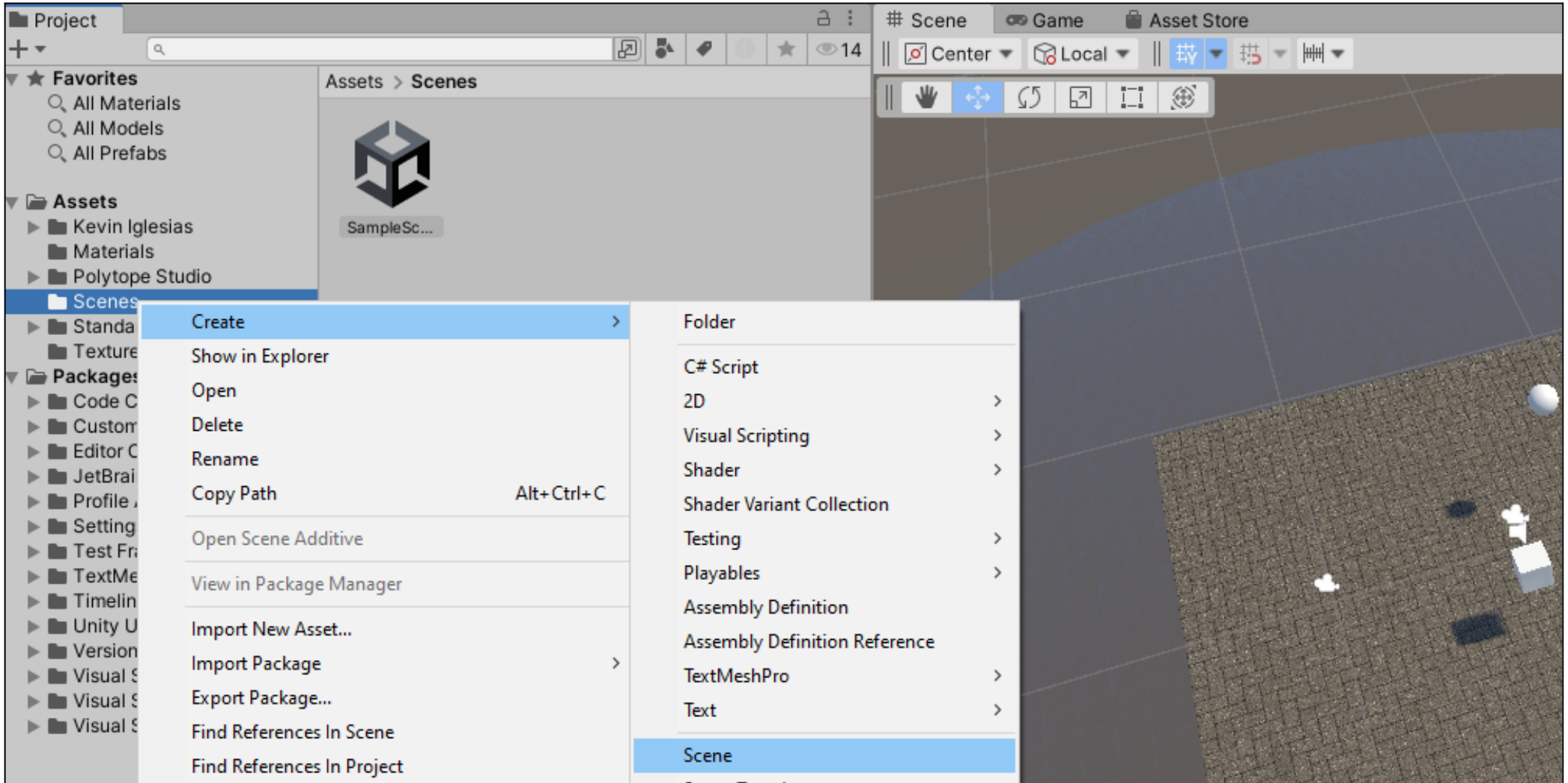


Из такой тестовой прогулки становится ясно что плавать по воде нельзя, а камера проваливается под воду и падает в бездну.

Во время режима **Play** любые изменения параметров объектов сцены применяются на ходу, что значительно облегчает тестирование. После остановки все изменения, проведенные после запуска режима **Play**, автоматически сбросятся.

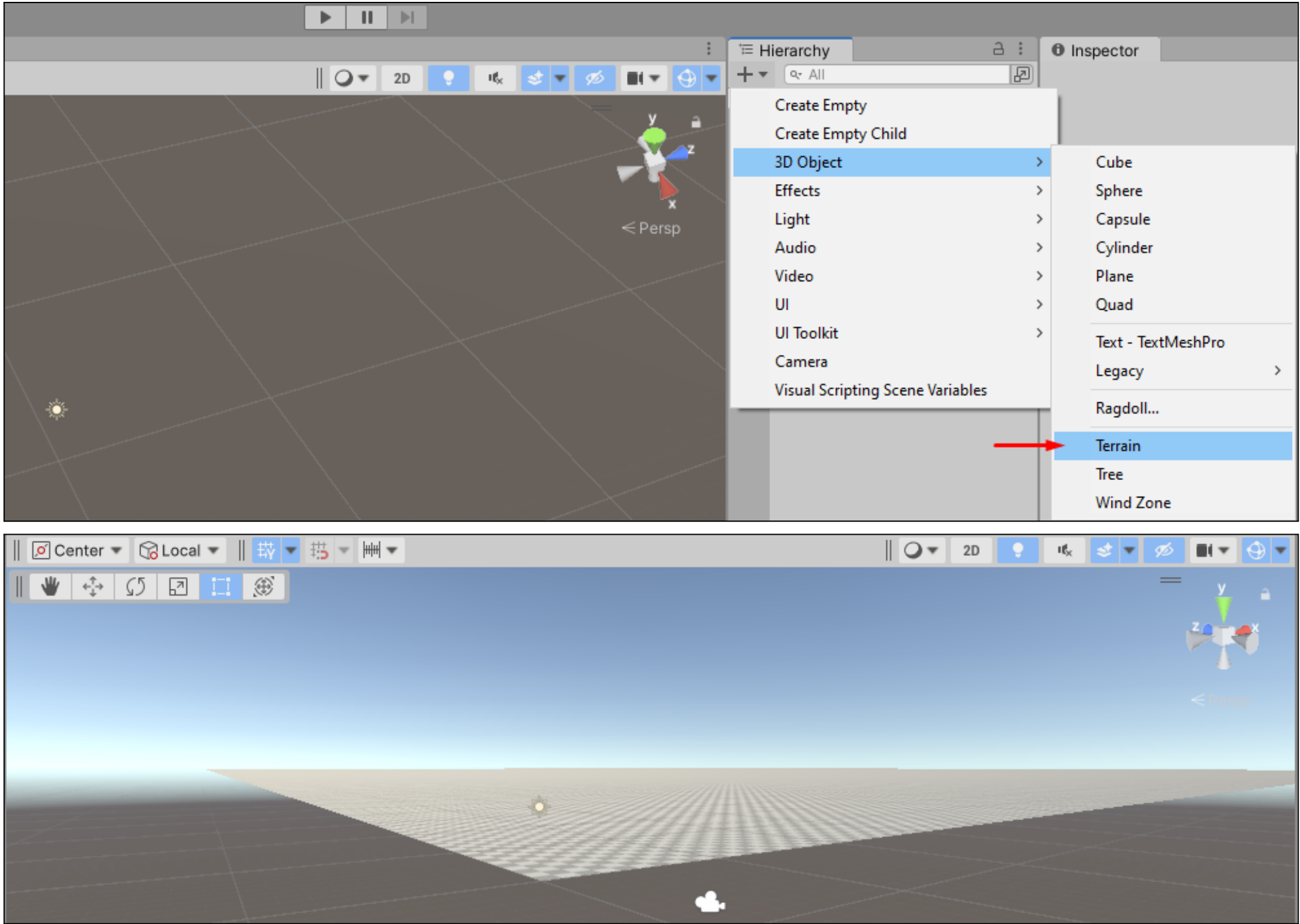
Новая сцена

Создать новую сцену, дать ей имя и открыть:

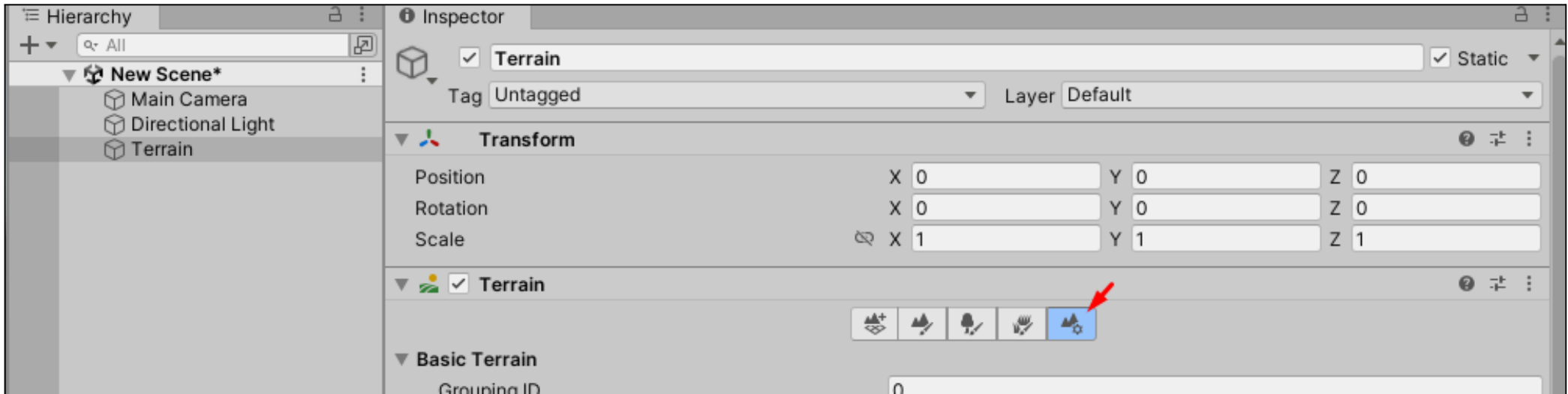


✂ Создание ландшафта

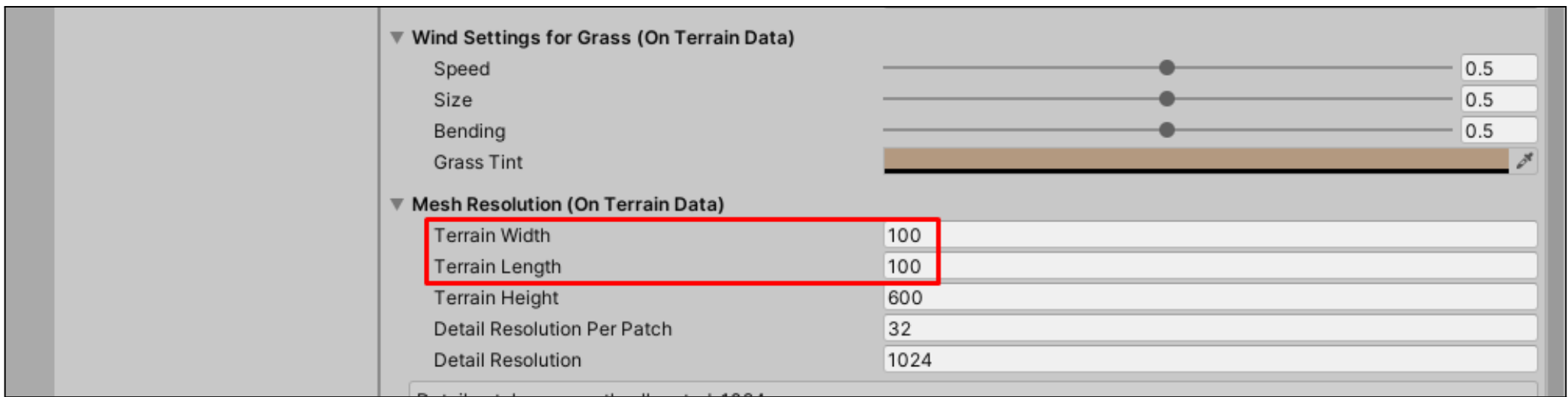
На новой сцене создать объект **Terrain**, это тип поверхности, который можно изменять прямо в окне сцены:



У объекта ландшафта несколько разделов свойств в окне инспектора, с хорошо говорящими иконками. Размеры редактируются в настройках [**Terrain Settings**]:



Общий размер поля уменьшен:



✂ **Изменение ландшафта**

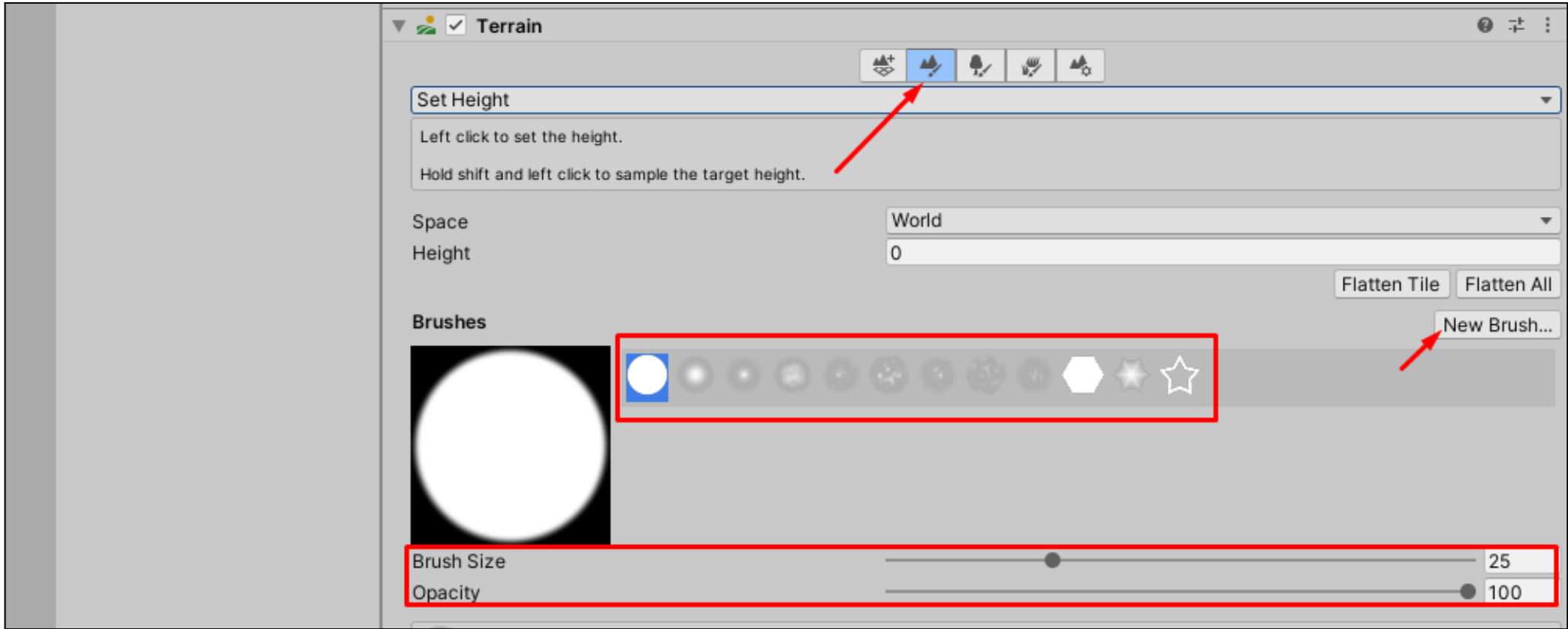
Остальные разделы настроек относятся к редактированию поверхности с помощью различных инструментов: **Terrain Tools**.

[**Paint Terrain**] - раздел с параметрами (формами) кистей для изменения рельефа - редактирование высоты отдельных участков.

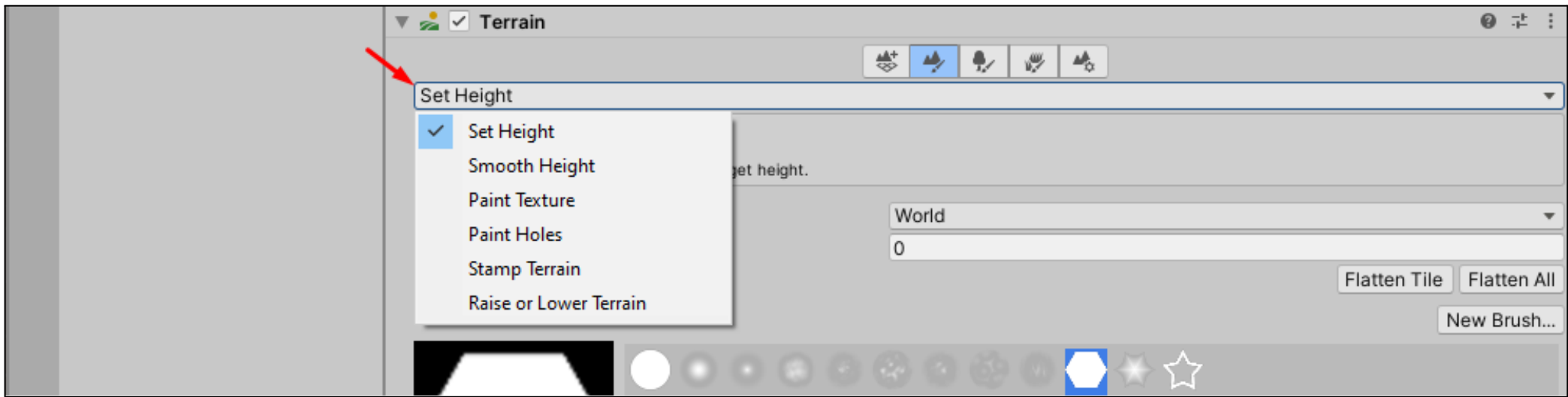
[**New Brush**] - загрузка новой кисти (на текущем этапе это не требуется).

Brush Size - радиус кисти.

Opacity - непрозрачность кисти - влияет на резкость вносимых изменений.



У редактирования высоты несколько режимов:



Set Height - установка заданной высоты (значение параметра **Height**), чем прозрачнее кисть, тем более плавный переход (склон).

- **Flatten Tile** задает указанную высоту всему участку поверхности
- **Flatten All** задает указанную высоту всем объектам в сцене на выбранном участке поверхности

Smooth Height - сгладить перепады высоты.

Paint Texture - раскраска поверхности текстурой - потребуется добавить текстуру через функцию [**Edit Terrain Layers**],[**Add layer**]. Первая добавленная текстура закрасит весь участок поверхности целиком.

Paint Holes - вырезать участок поверхности (стереть) **ЛКМ** / **Shift ЛКМ** - отменить стирание.

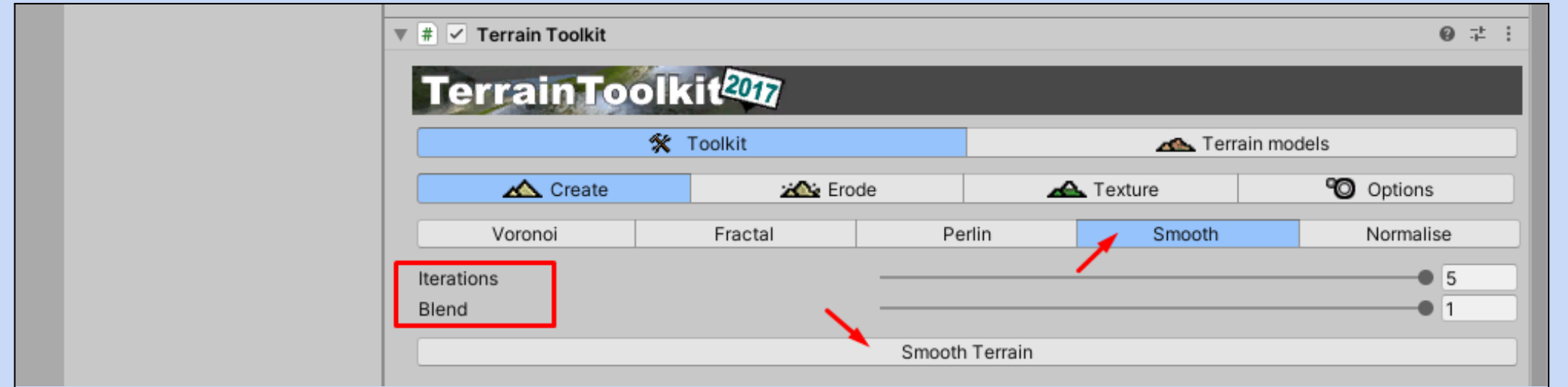
Stamp Terrain - выдавливание участка по рисунку кисти на заданную высоту. **Max <--> Add** регулирует тип поднятия - максимальная высота, **ДО которой** надо поднять участок, или высота, **НА которую** надо поднять редактируемый участок рельефа.

Raise or Lower Terrain - регулировка высоты через **ЛКМ** / **Shift ЛКМ** - вверх / вниз.

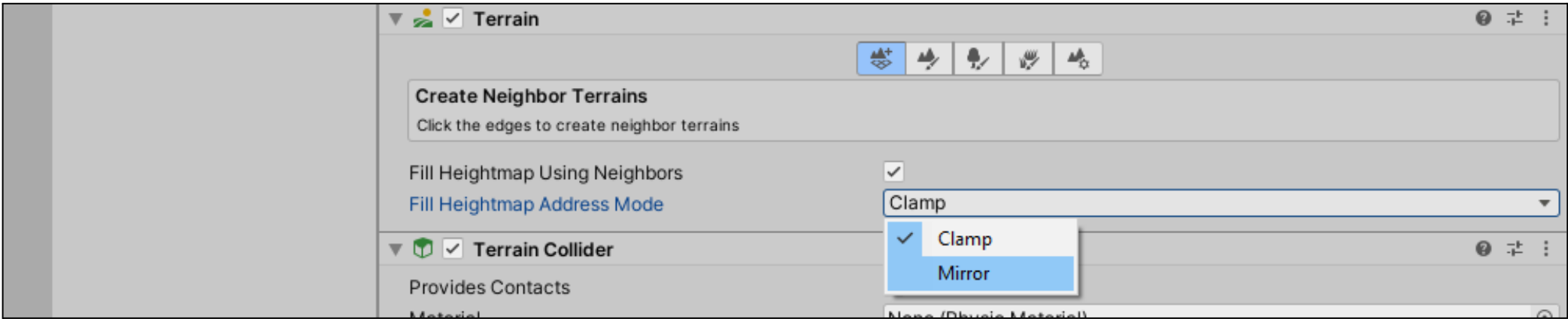
Опционально: Ассет для расширения работы с высотами: **Terrain Tool Kit 2017** (есть в скачанном архиве Terrain Tool Kit 2017.unitypackage)

После его импорта надо выбрать в меню редактора **Component / Terrain / Terrain Toolkit**. Загруженный ассет добавит каждому объекту типа Terrain новый компонент **Terrain Toolkit**, с помощью которого можно генерировать и редактировать ландшафт процедурно.

Пример: сглаживание применяется ко всей поверхности объекта:



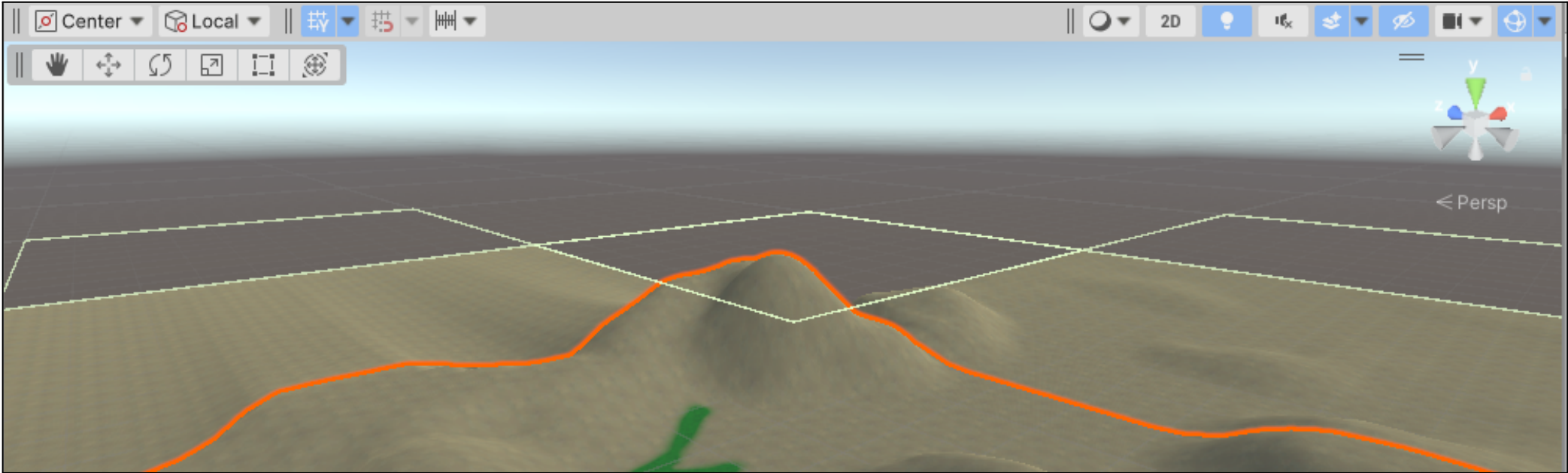
[Create Neighbor Terrains] - автоматическая отрисовка новых участков поверхности по клику на прилегающую зону:



Новые участки будут добавляться в папку размещения исходного объекта **Terrain** со сгенерированными именами.

Тип отрисовки **Clamp** продолжает соседний участок, сглаживая уровень высоты на границе до общего уровня.

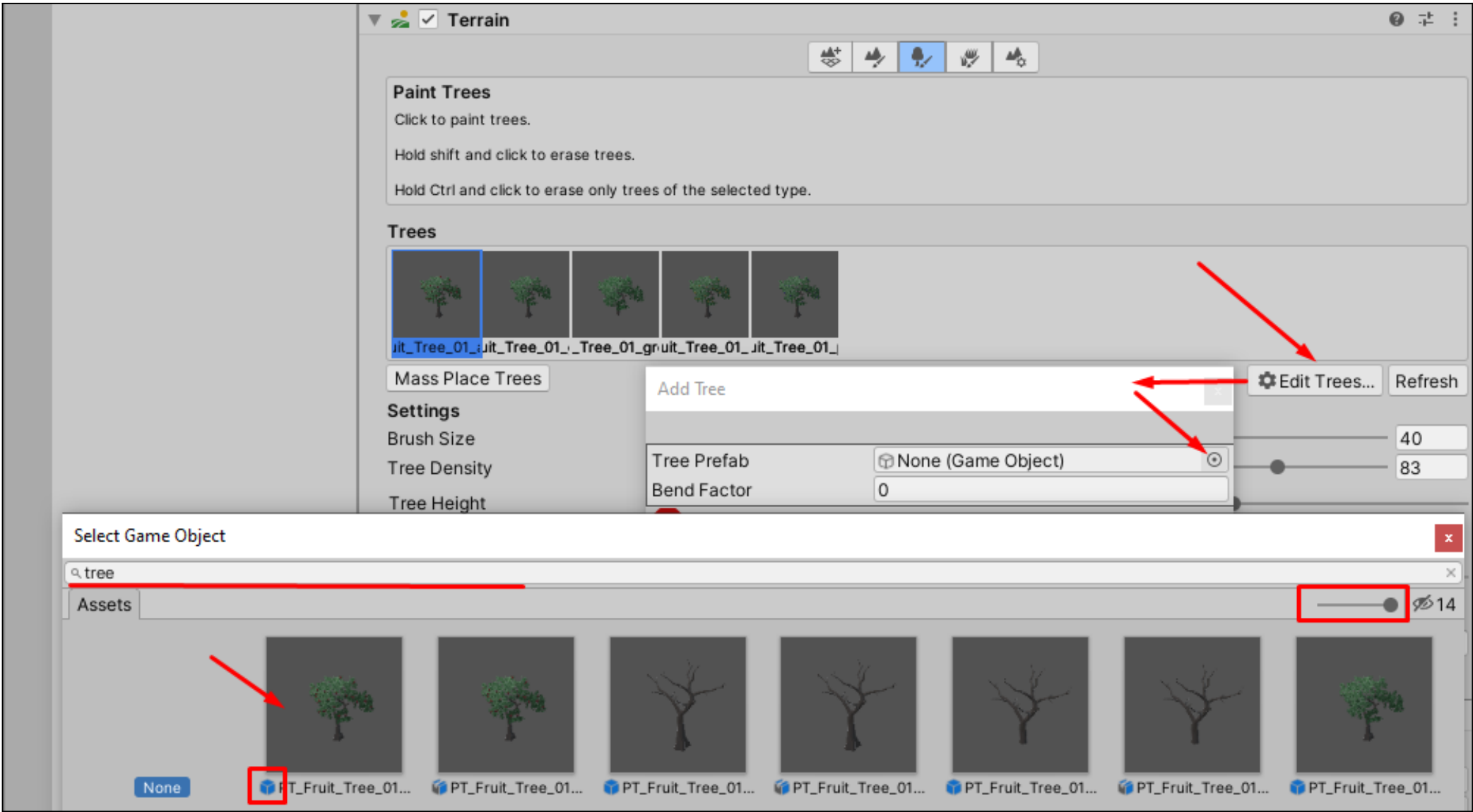
Тип отрисовки **Mirror** создает похожие перепады высоты и складки рельефа.



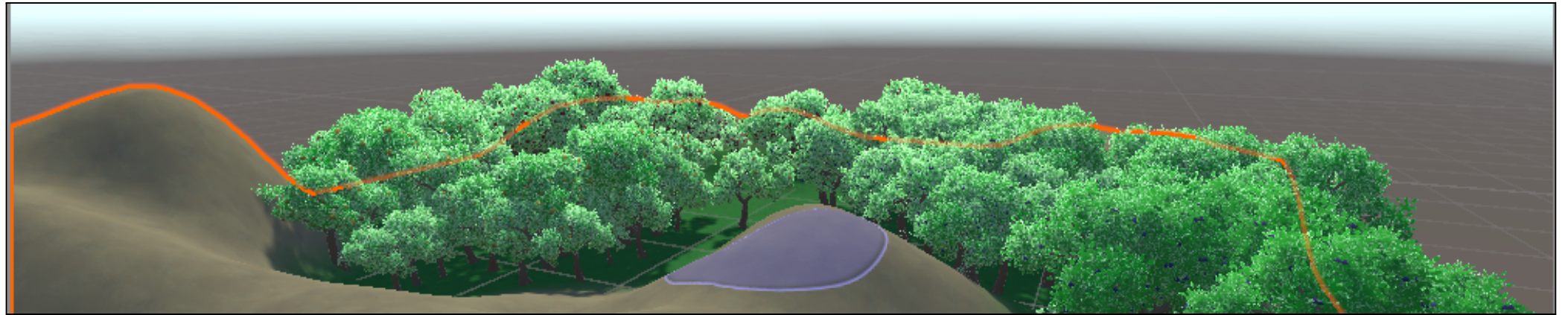
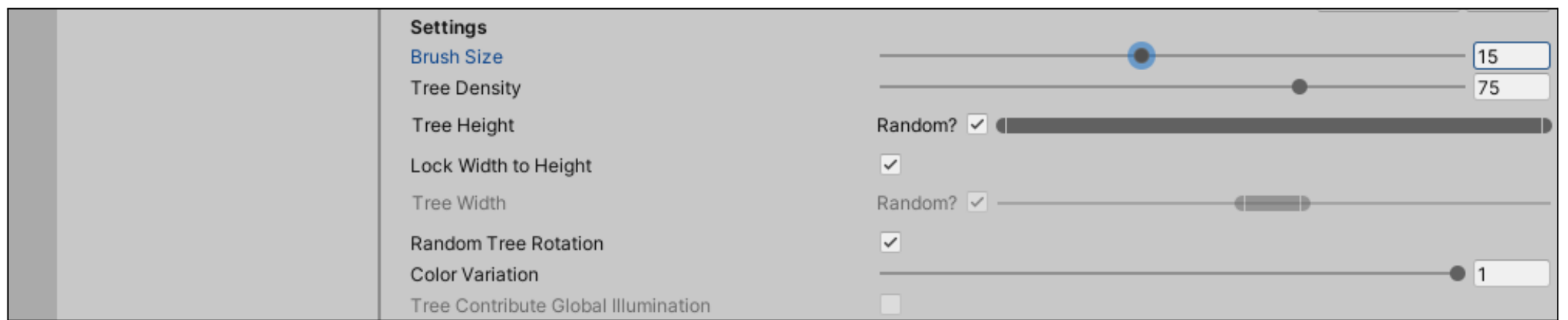
[Paint Trees] - добавление деревьев.

Чтобы добавлять деревья кистью, надо сначала подобрать используемые модели - [Edit Trees] / [Add Trees].

Открыть окно поиска, в окне набрать **tree** и выбрать подходящие модели:



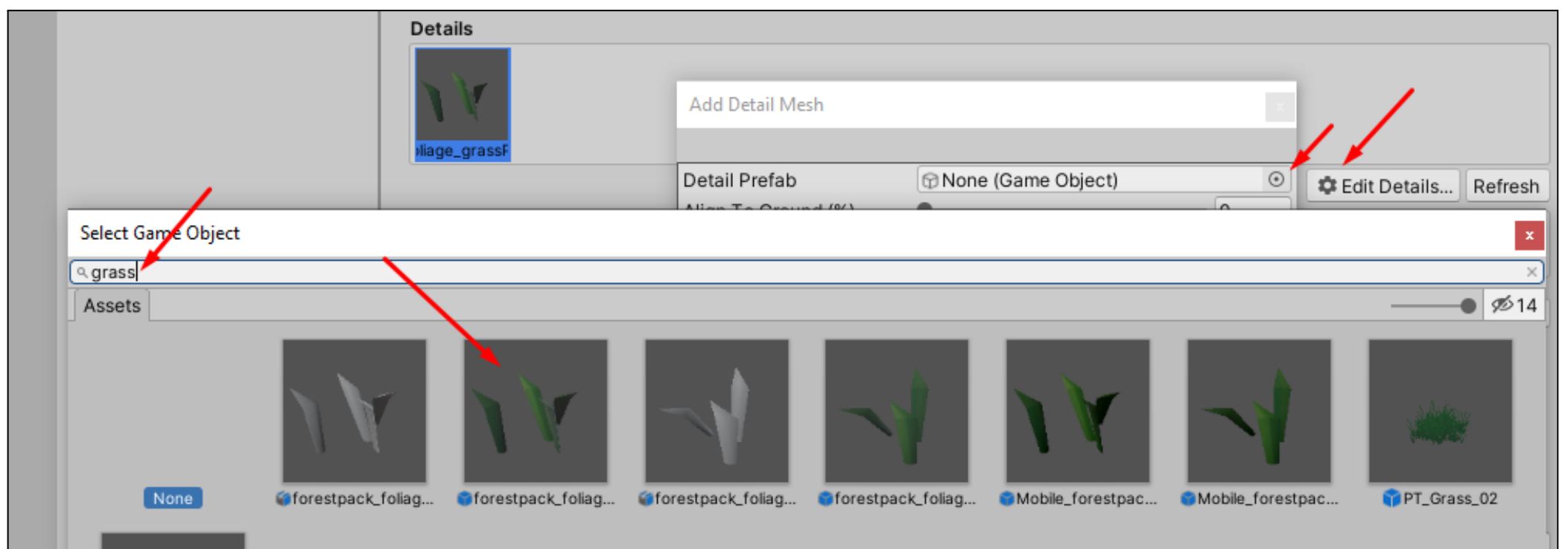
Выбрав модель дерева (предпочтительно выбирать префаб) надо задать размер кисти и плотность посадки деревьев. Определив диапазон случайного разброса по размерам, можно легко размещать деревья на местности кистью (**ЛКМ** разместить / **SHIFT + ЛКМ** стереть):



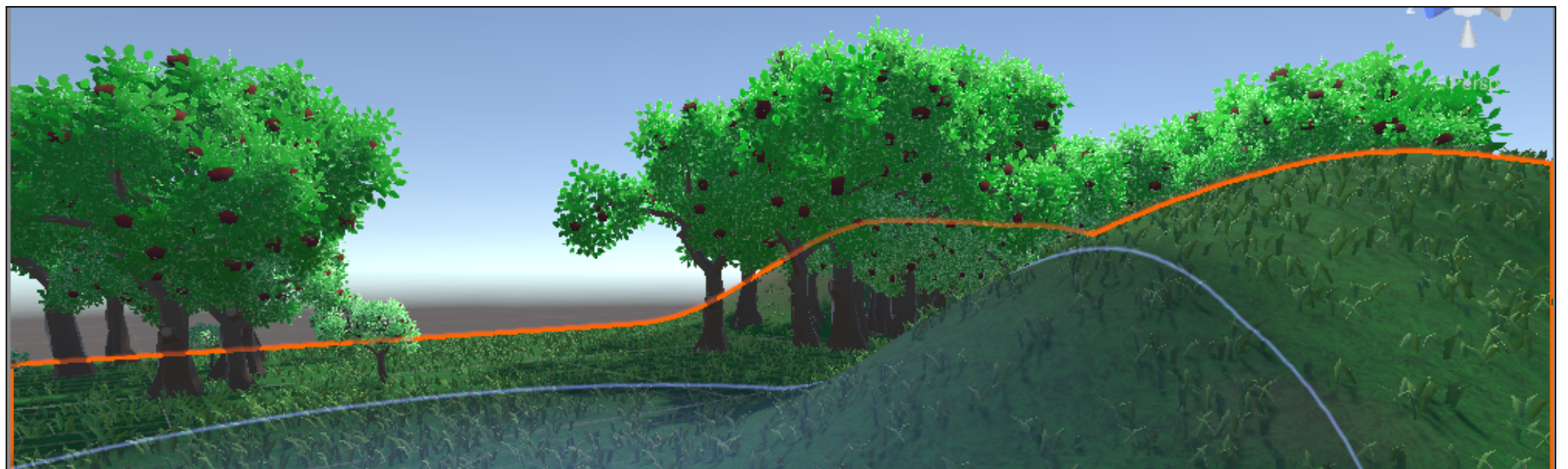
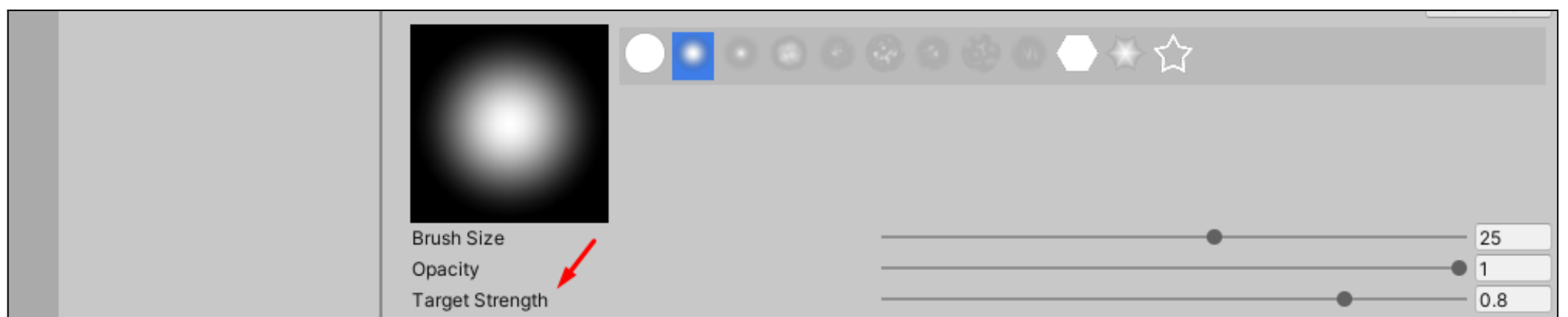
[**Paint Detail**] – кисть для размещения мелких деталей – травы, кустарников, камней и прочих небольших объектов.

Трава добавляется по аналогии с моделями деревьев с небольшим отличием:

- **Add Detail Mesh** добавляет объемную модель
- **Add Grass Texture** добавит плоское 2D изображение травы

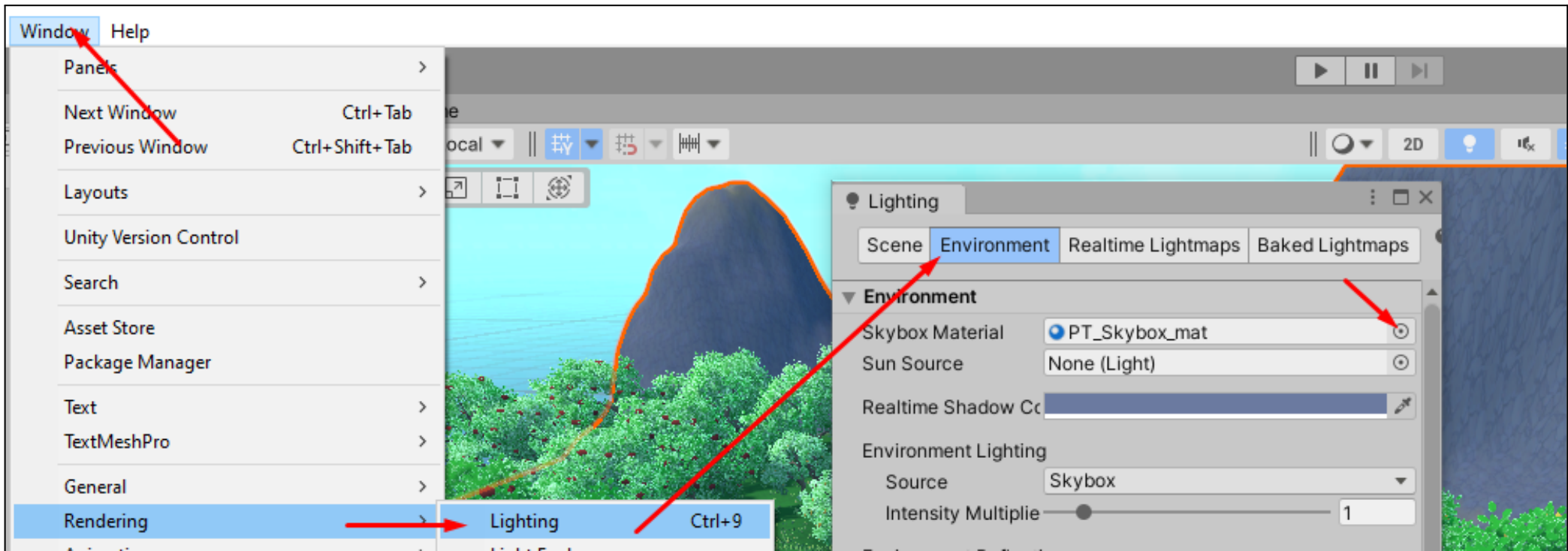


Плотность размещения объектов задается параметром кисти **Target Strength**:

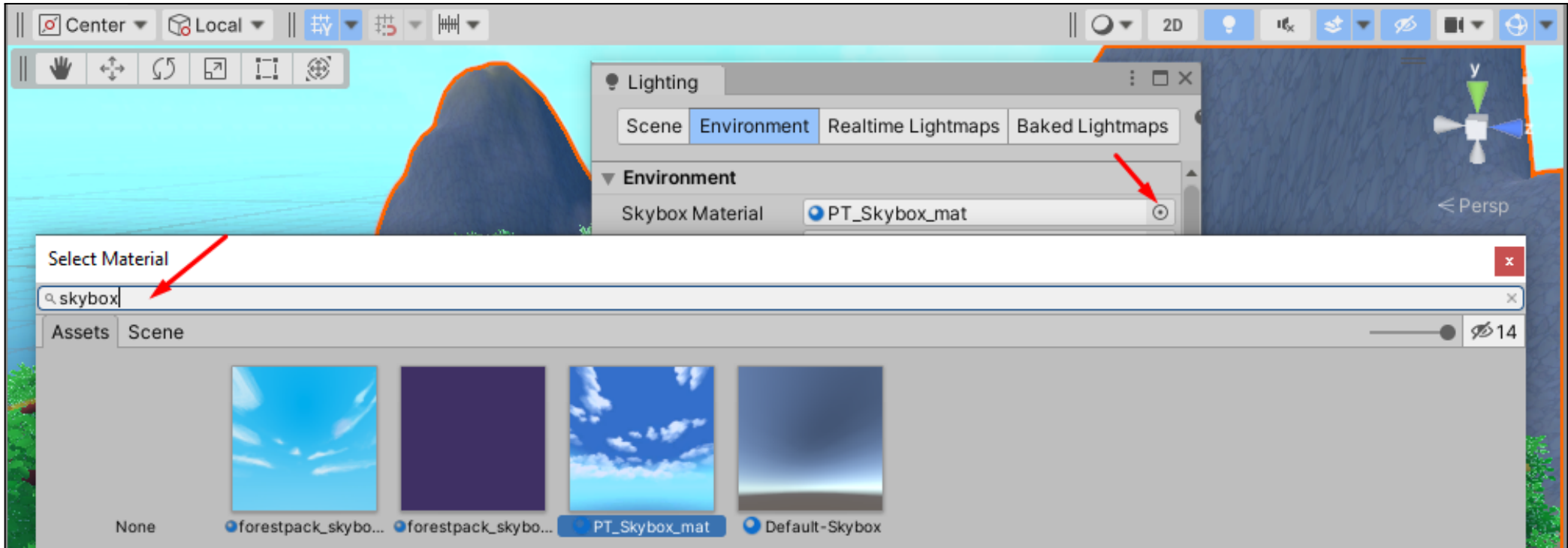


✂ **Настройка неба**

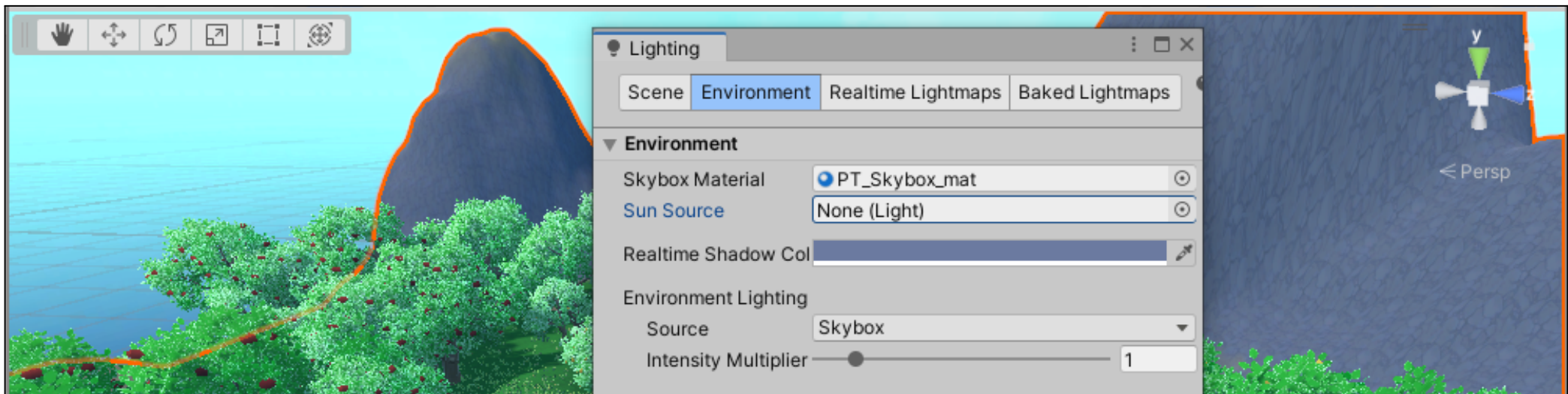
Меню редактора **Window / Rendering / Lighting**, в открывшемся окне настраивается общее фоновое освещение для сцены. Открыть вкладку **Environment** и далее выбор материала для неба:



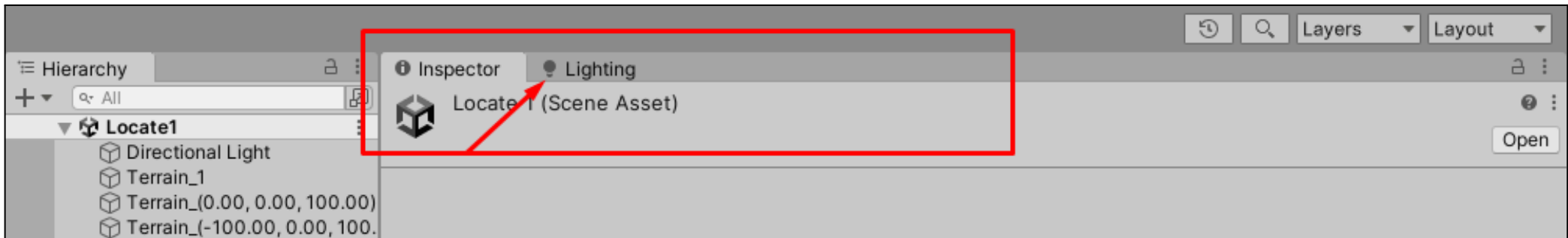
Задать стандартное наименование для поиска **Skybox** и выбрать один из вариантов:



Параметр **Intensity Multiplier** определяет насколько сильно цвет неба влияет на цвет объектов сцены. Его можно отключить, если в **Source** выбрать параметр **Gradient** или **Color** и указать ручные настройки.

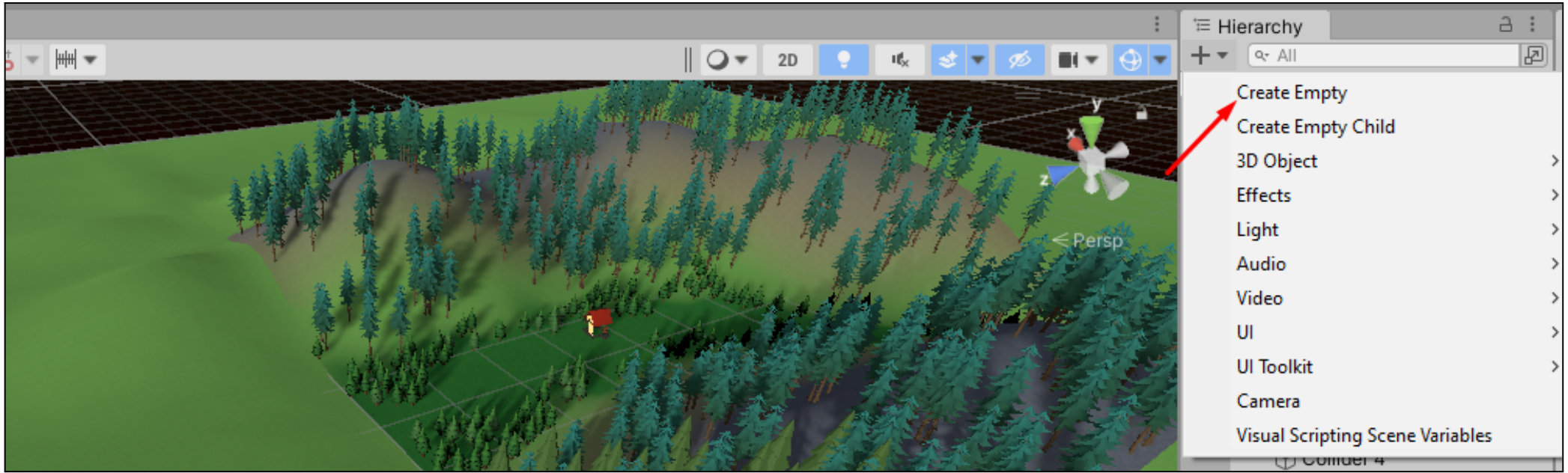


Окно [**Lighting**] целесообразно сразу закрепить на экране, как одну из альтернативных вкладок рядом с инспектором. Помимо света в нем настраиваются тени, спецэффекты и прочее.

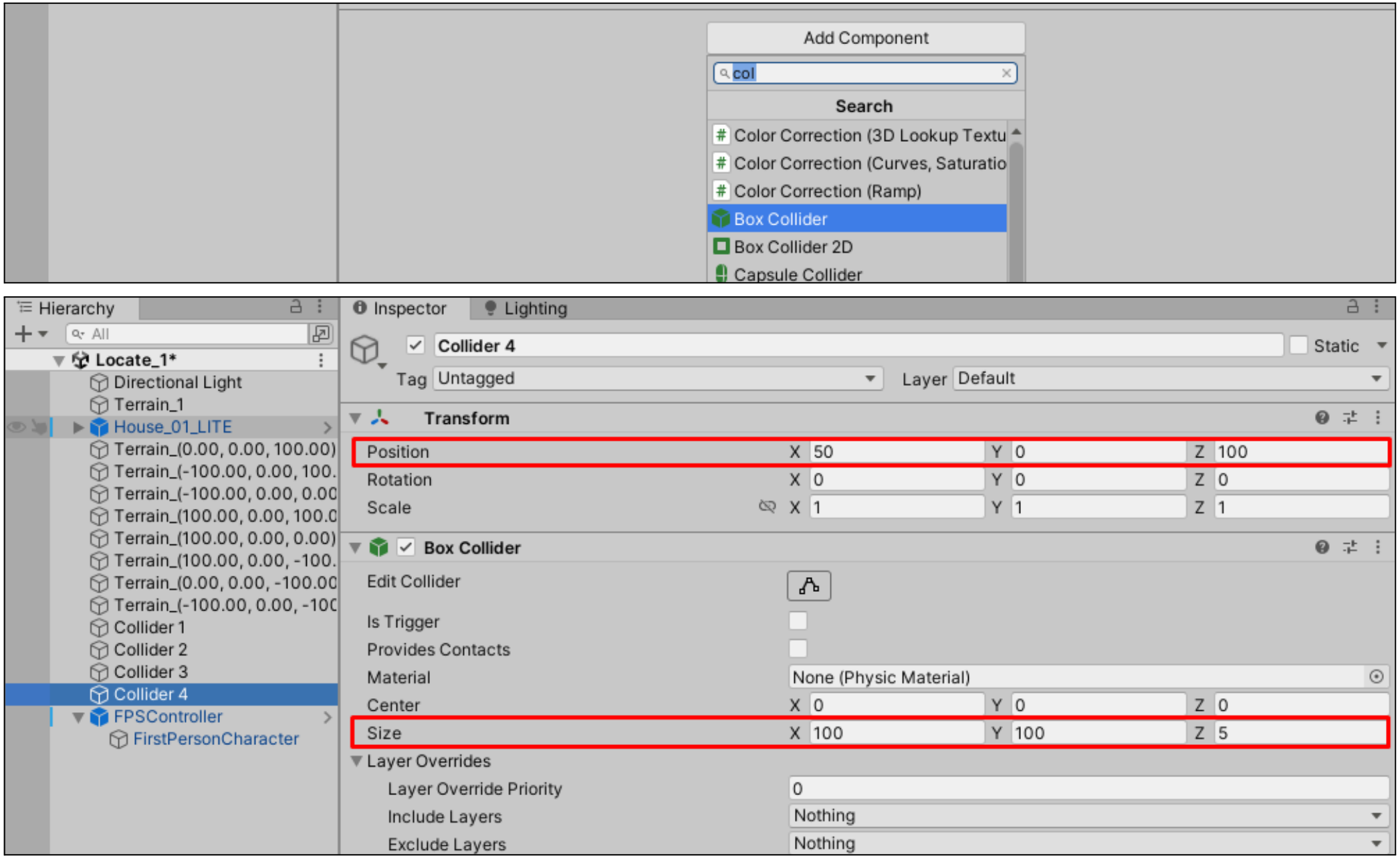


※ Стена из коллайдера

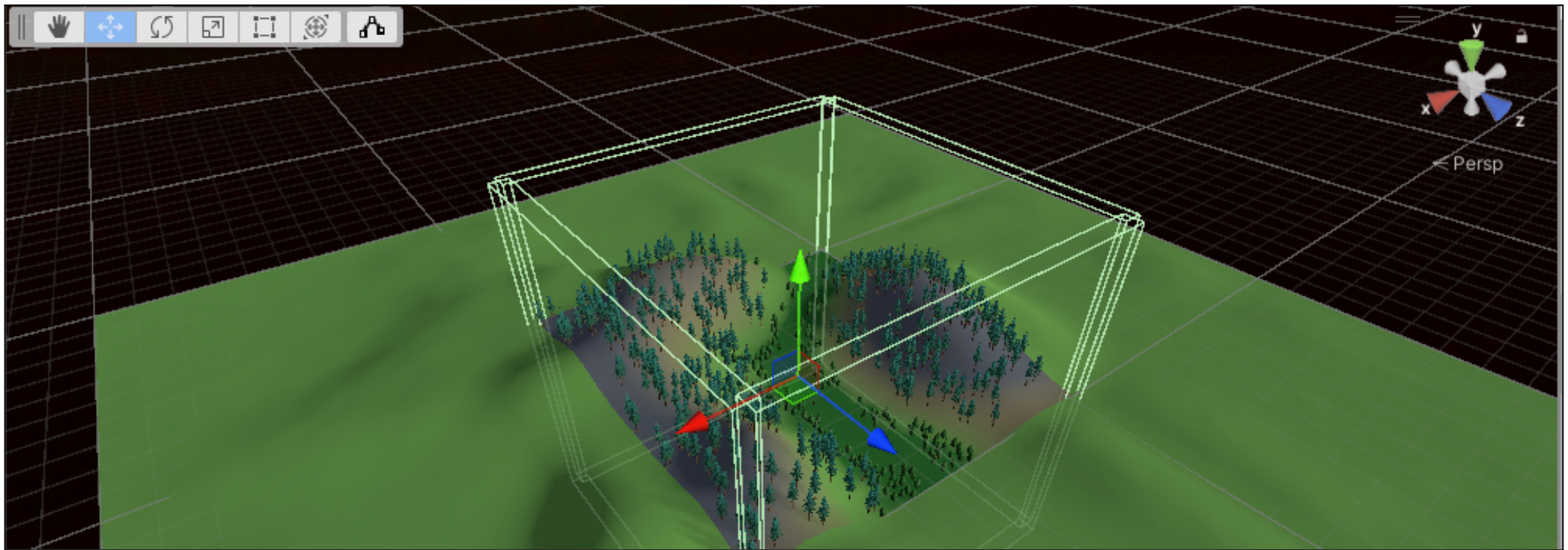
Невидимые стены вокруг игровой зоны создает коллайдер. Для создания коллайдера сначала создается пустой объект [Create Empty](#):



Этому объекту добавляется компонент [Box Collider](#), и ему выставляются **координаты** и **размеры** таким образом, чтобы получилась стена:



Выставление заграждений из таких стен создает границу локации, и при попытке выйти за пределы локации персонаж упрется в невидимую стенку - в один из блоков-коллайдеров:



※ Коллайдер

Компонент **Collider** это физические границы объекта, они могут не совпадать с его визуальной формой - мешом. Кубу можно дать коллайдер сферы, и тогда он будет катиться как мяч. Если у объекта нет коллайдера, то сквозь него могут проходить другие объекты.

Если объект не должен двигаться в сцене (пол, стены), то он является **статическим**, и добавлять ему компонент **Rigidbody** не надо. Не надо после запуска игры скриптами менять положение статических коллайдеров изменяя параметры **Transform** - это сильно влияет на производительность движка. Коллайдеры на объекте с компонентом **Rigidbody** называют **динамическими**.

У базовых объектов типа куб, сфера и капсула (цилиндр) набор параметров почти одинаковый.

Коллайдеры базовых объектов являются **примитивными**.

Cube / Sphere / Capsule Collider

Edit Collider - редактирование границ коллайдер доступно только для трех базовых форм объекта.

Is Trigger - превращает объект в зону взаимодействия, которая отслеживает попадание в нее других объектов с коллайдерами, и не отталкивает их, обычно в паре с этим параметром отключают отображение визуальной формы - вода для плавания, чекпоинты в гонках.

Provides Contacts - когда он активен в систему физики Unity записываются все события о его взаимодействии; когда он выключен, в систему записываются только действия, определенные разработчиком (стандартная ситуация).

Material - определяет **физические** свойства (**физический материал**) объекта (сила трения, упругость и т.д.).

Center - задает отклонение от центра объекта, по умолчанию все смещения = 0.

- **Radius** - для сферы и капсулы - радиус.
- **Size** - для куба размеры по трем осям.
- **Height** - для капсулы - высота.
- **Direction** - для капсулы - вдоль какой оси вытянута капсула.

Layer Overrides - матрица столкновений.

Terrain Collider - коллайдер объекта **Terrain**, его отличительные свойства:

Terrain Data - ссылка на объект, к которому применяется коллайдер.

Enable Tree Colliders - включить или отключить коллайдеры деревьев.

Mesh Collider - коллайдер объекта с произвольной моделью (не примитивный) - строится по MESH модели объекта (затратно по CPU).

Convex - упрощает исходную форму такого коллайдера, округляя его контуры, рисуя контур по внешним вершинам (выпуклый коллайдер).

Cooking Options - опции оптимизации.

Mesh - ссылка на Mesh.

Форма **Convex** коллайдера (выпуклого), приближенная к форме сложного объекта, более выгодна в плане производительности, чем оригинальный Mesh-коллайдер. **Примитивные** формы коллайдера наиболее производительны. Из примитивных коллайдеров можно собрать составную форму, которая будет повторять общие контуры сложного объекта, но быть легче Mesh-коллайдера - получится аналог **Convex** коллайдера.

Однако надо учесть нюансы:

Примитивные коллайдеры в составной фигуре плохо работают со смещениями, вращениями и неоднородными масштабами в иерархии преобразования.

Mesh-коллайдер обычно НЕ может столкнуться с другим Mesh-коллайдером. Mesh-коллайдер можно преобразовать в **Convex** (выпуклый), тогда он будет сталкиваться с другим Mesh-коллайдером.

Хорошее общее правило - использовать Mesh-коллайдер для **геометрии сцены**, а для **движущихся объектов** использовать примитивные и составные коллайдеры.

С коллайдерами тесты связаны **физические материалы** (**Material**), они должны отражать свойства поверхности. Лед гладкий и хрупкий, по нему скользят другие объекты. Резина шершавая и упругая. У льда низкое трение и прочность, резины высокие трение и прочность.

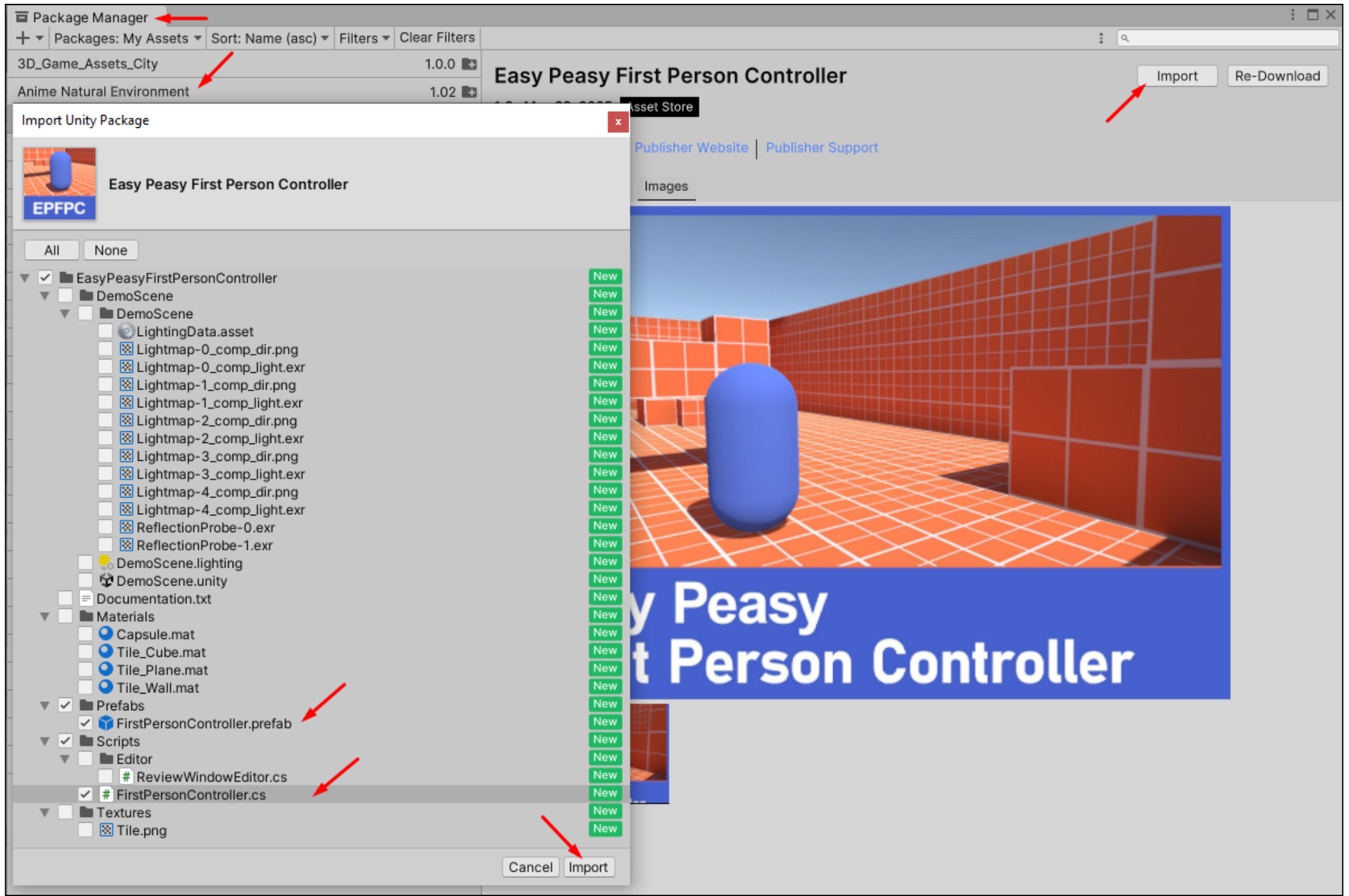
Коллайдер в режиме **Is Trigger** имитирует зону, поэтому чаще всего объект с этим параметром лишен визуальной формы. Такой коллайдер используется для запуска событий и по умолчанию игнорируется движком.

Тестовый персонаж

Основа персонажа (или аналогичного объекта) - **модель, анимация, камера, управление**. Однако для проверки карты уровня полноценный персонаж не обязателен. В качестве тестовой модели достаточно загрузить любой примитивный контроллер для персонажа из магазина ассетов.

Бесплатный пример, из которого нужно минимум всего два объекта:

<https://assetstore.unity.com/packages/tools/physics/easy-peasy-first-person-controller-317073>



После импорта разместить на карте модель **FirstPersonController**, сделать запуск для проверки.

УПРАВЛЕНИЕ:

- Движение: WASD или клавиши со стрелками
- Бег: левый Shift+W
- Приседание: левый Control
- Скольжение: левый Control во время бега
- Прыжок: пробел

Управление персонажем возможно из-за наличия на нем скрипта и компонента [**Character Controller**].

✂ Контроллер персонажа

Компонент **Character Controller** дает персонажу простой коллайдер в форме капсулы, который всегда находится в вертикальном положении, частично имитирует физику (импульс игнорируется). У него особые функции для назначения скорости и направления объекта, и он не требует наличия компонента **Rigidbody** (физическое тело).

Персонаж с этим компонентом не может проходить сквозь **статические коллайдеры** в сцене - не проваливается сквозь пол, и не ходит сквозь стены. При движении он может отталкивать объекты с **Rigidbody**, но когда они толкают его, персонаж не получает импульс.

Персонаж будет подниматься по лестницам, если высота ступеньки ниже значения свойства **Step Offset**,

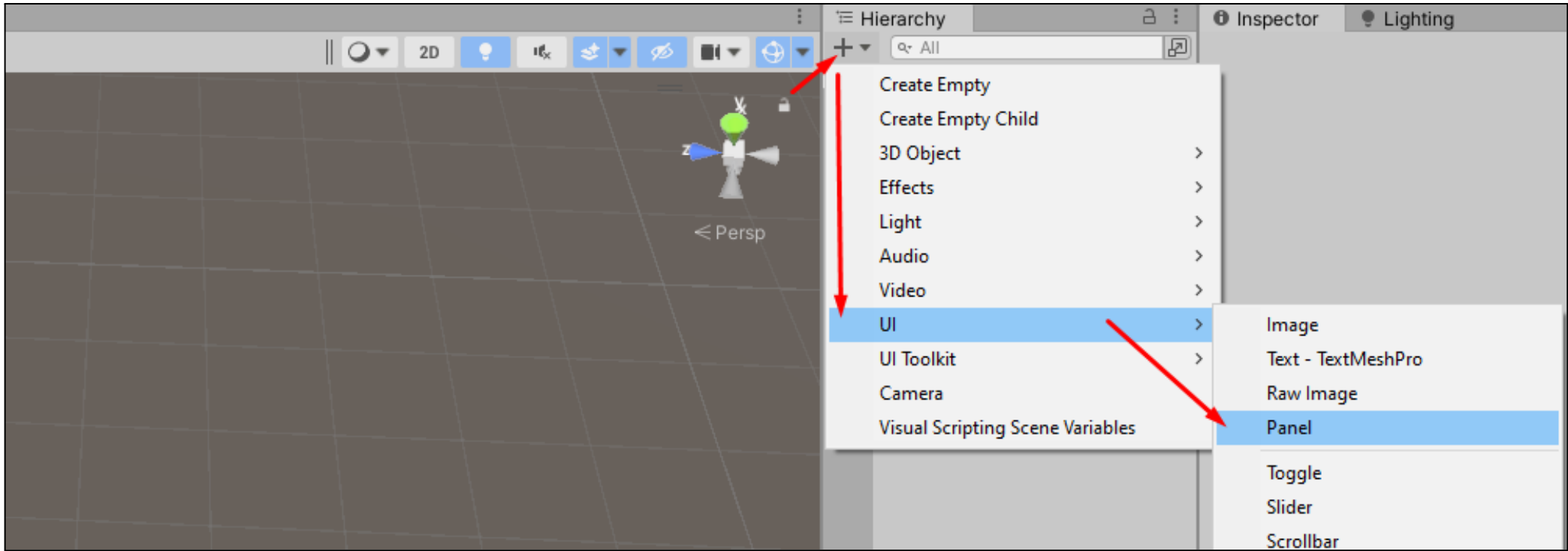
Персонаж будет подниматься по склону если угол наклона у склона меньше значения свойства **Slope Limit**.

Свойство **Skin Width** позволяет другим объектам (коллайдерам) слегка пересекать контроллер. Это уменьшает тряску при соприкосновениях и защищает от застревания. Рекомендуемое значение более **0.01** и больше чем **Radius / 10**.

Уменьшить тряску при движении позволяет увеличение **Min Move Distance**, но чаще всего его оставляют 0 или близком значении - 0.0001.

Меню игры

Создать новую сцену, переименовать ее в **MainMenu**, открыть сцену и добавить в окне иерархии объектов **UI / Panel**:



Объект **Panel** автоматически создал холст **Canvas** в сцене. У объекта **Canvas** есть режимы рендеринга **Render Mode**:

- **Screen Space - Overlay** - Canvas напрямую привязан к экрану, при изменении размера экрана или разрешения, Canvas также изменит свой масштаб на соответствующий. UI будет отрисовываться поверх других объектов. При этом важен порядок размещения объектов в иерархии. Холст должен находиться на самом верху в иерархии, иначе он может пропасть из виду.
- **Screen Space - Camera** - холст прикрепляется к камере. Для такой настройки надо обязательно указать камеру, которой будет соответствовать холст. Холст будет менять свой масштаб в зависимости разрешения и размера экрана, а также от области видимости камеры. Важно размещение относительно других объектов - на переднем плане будут объекты, которые ближе к камере, независимо от того относятся они к UI или нет.
- **World Space** - холст размещается как обычный объект без привязки к камере и экрану, может быть ориентирован любым образом. Размер задается с помощью **RectTransform**, но то, как его будет видно во время игры будет зависеть от положения камеры игрока.

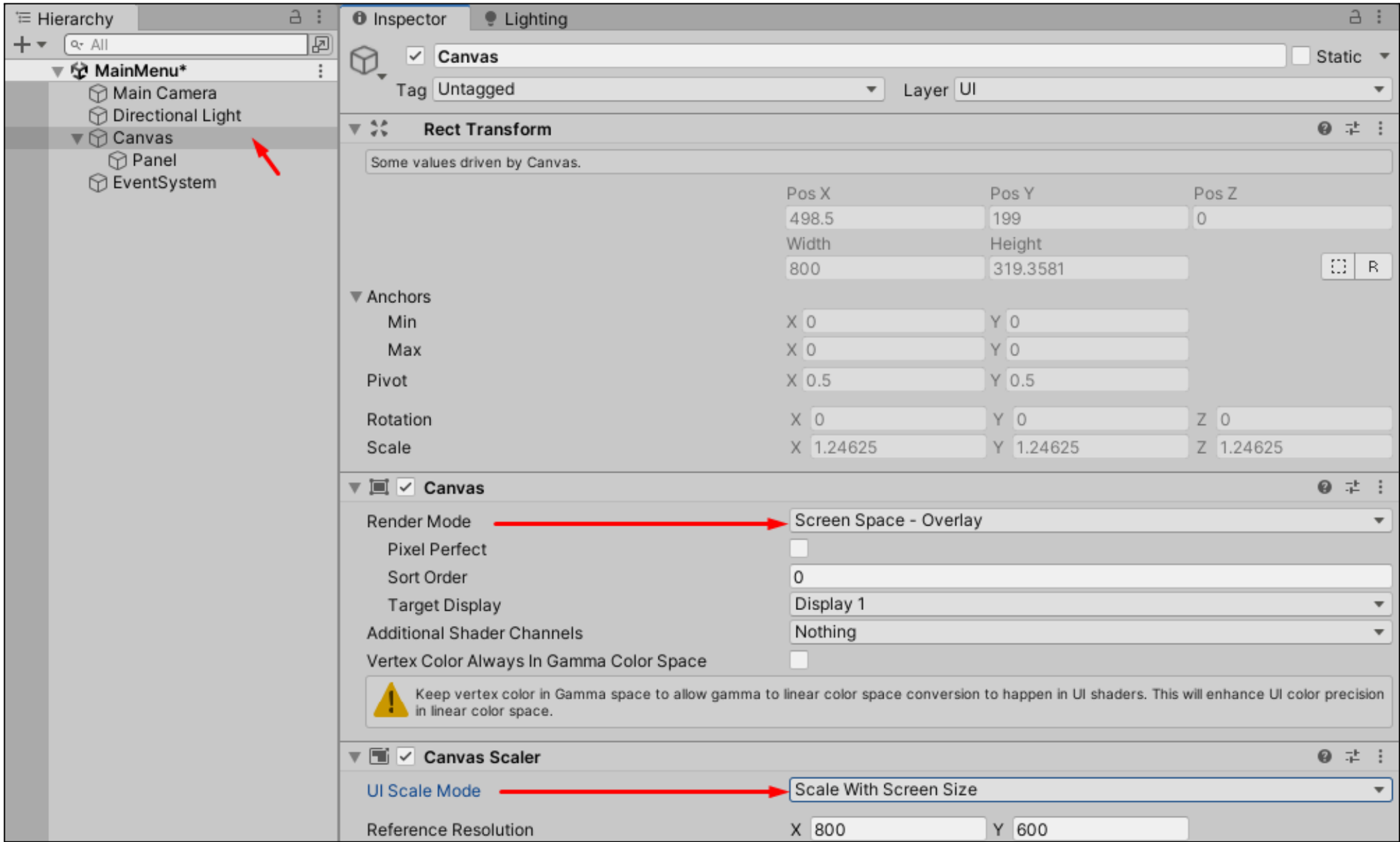
Render Mode установить на **Screen Space - Overlay** или **Screen Space - Camera**.

Свойство **UI Scale Mode** - как будет происходить масштабирование пользовательского интерфейса в зависимости от разрешения экрана или размеров окна приложения. Как элементы интерфейса будут адаптироваться к различным экранам и разрешениям.

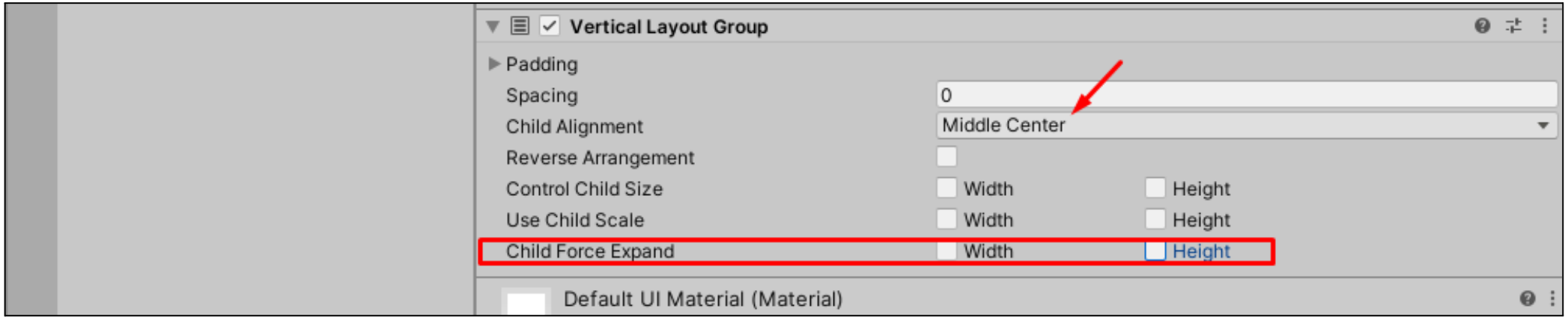
- **Constant Pixel Size** (постоянный размер в пикселях): Элементы сохраняют размер в пикселях независимо от разрешения экрана. С высоким разрешением элементы интерфейса будут выглядеть меньше, а на устройствах с низким разрешением - больше.
- **Scale with Screen Size** (масштабировать): Элементы масштабируются в зависимости от размера экрана.
- **Constant Physical Size** (постоянный физический размер): Масштабирование элементов интерфейса происходит в зависимости от физических размеров экрана, сохранить одинаковые размеры на устройствах с разными плотностями пикселей.

UI Scale Mode установить на **Scale With Screen Size** - заполнение во весь экран.

Поменять параметры объекта **Canvas**:



Объекту **Panel** добавить компонент **[Vertical Layout Group]**.
Новому компоненту свойство **Child Alignment** установить в **Middle Center** - все дочерние элементы будут выравниваться по центру экрана, и снять флаг у свойства **Child Force Expand**, чтобы дочерние элементы выстраивались друг под другом.

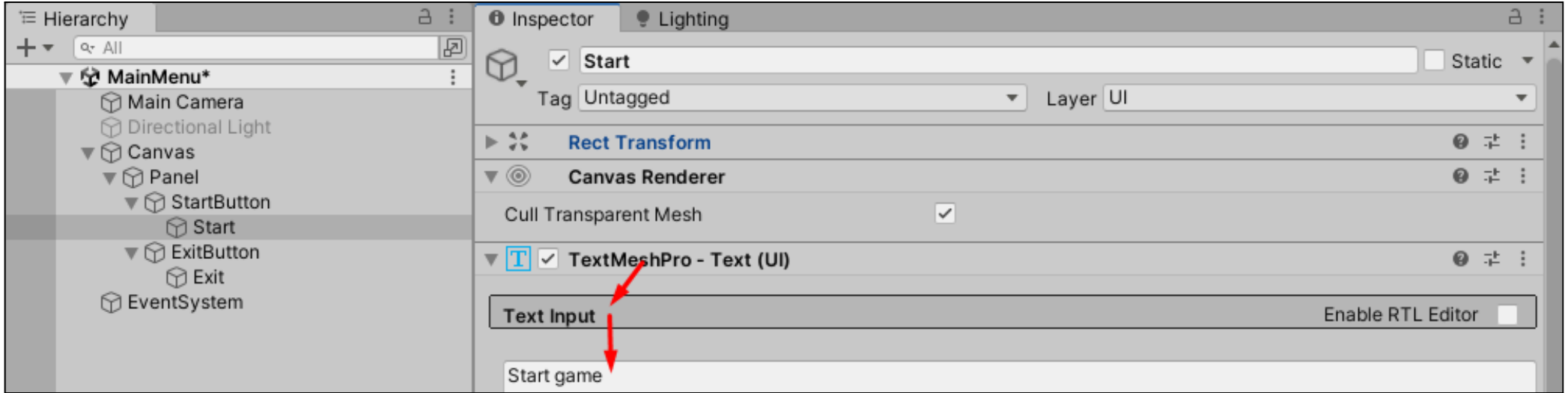


У объекта **Panel** открыть контекстное меню и выбрать **UI / Button - TextMeshPro** - добавить две кнопки - **Button - TextMeshPro**.

❌ **Кнопки меню**

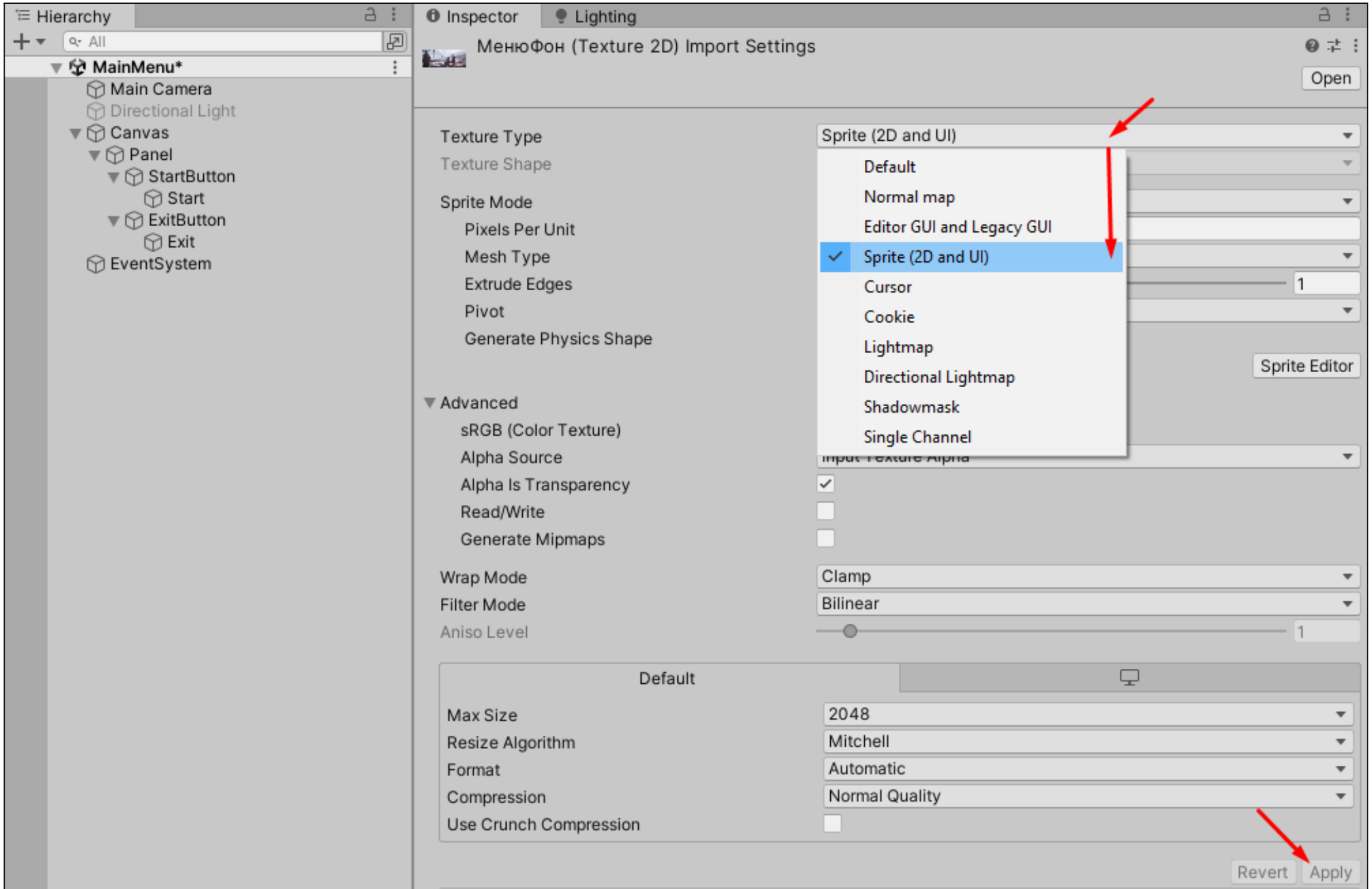
Верхнюю кнопку переименовать в **StartButton**. У ее дочернего элемента **Text (TMP)** сменить имя на **Start**, в поле **Text Input** ввести надпись для кнопки - **Start game**.
Нижняя кнопка по аналогии будет **ExitButton**, с текстом **Exit**.

При первом создании кнопок, то в открывшемся окне **TMP Importer** надо запустить импорт ресурсов, и надпись загрузится в кнопку.

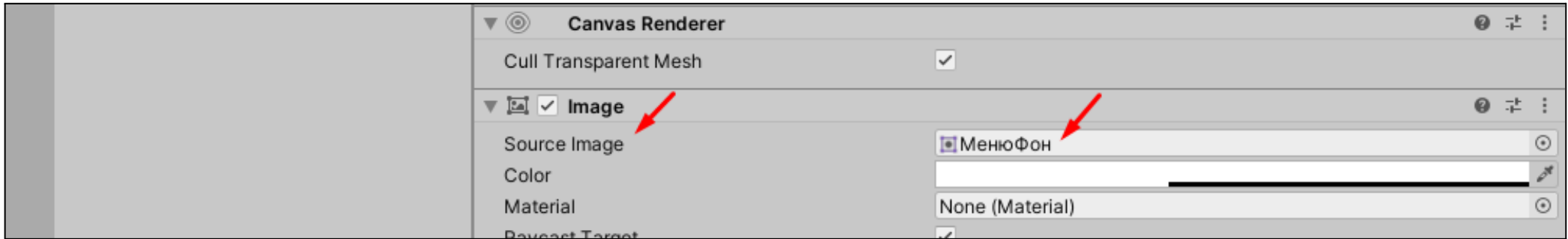


❌ **Фоновая картинка**

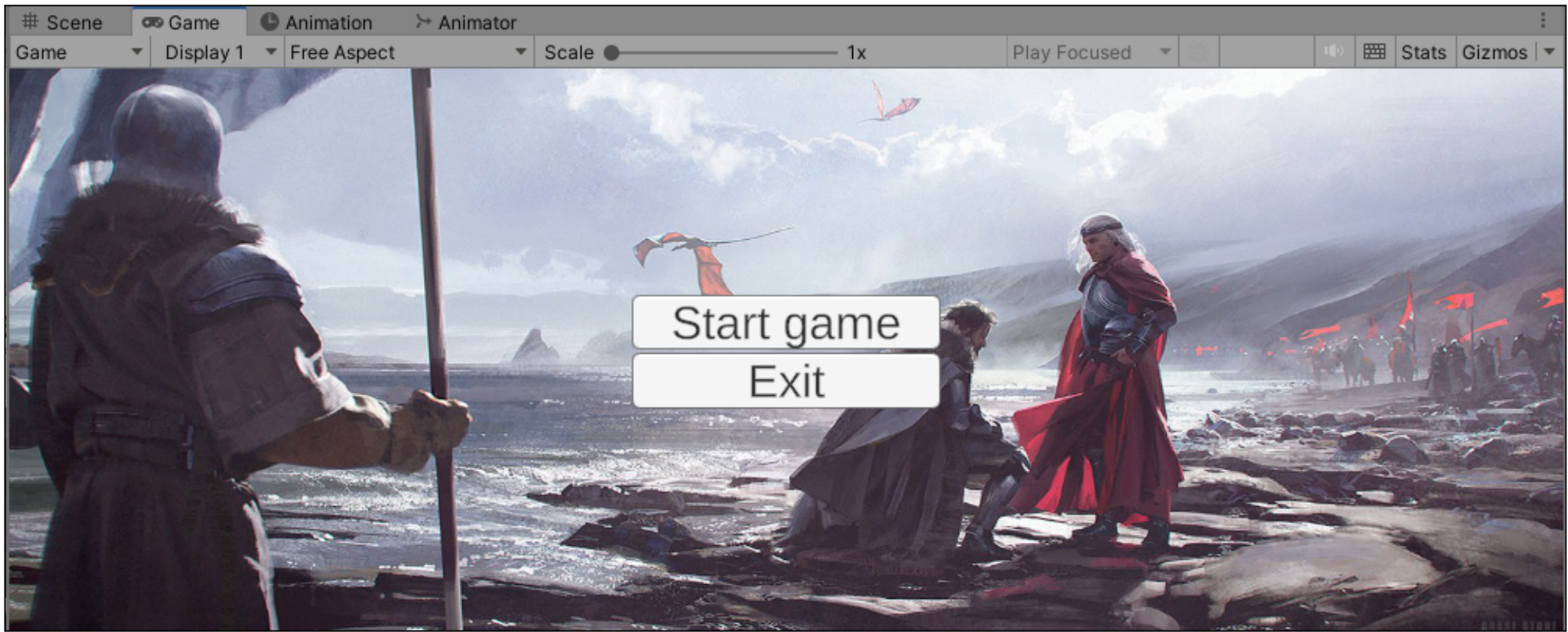
Добавить в проект любую картинку для фона меню. Изображение надо конвертировать в **Sprite (2D and UI)**, и принять изменения **[Apply]**:



После преобразования изображение можно использовать как фон для объекта **Panel1** - свойство **Source Image**:



В камере сцены свойство **Clear Flags** установить **Depth only**, чтобы фон сцены не отрисовался в кадре, так меню выглядит интереснее:



❌ **Скрипт запуска**

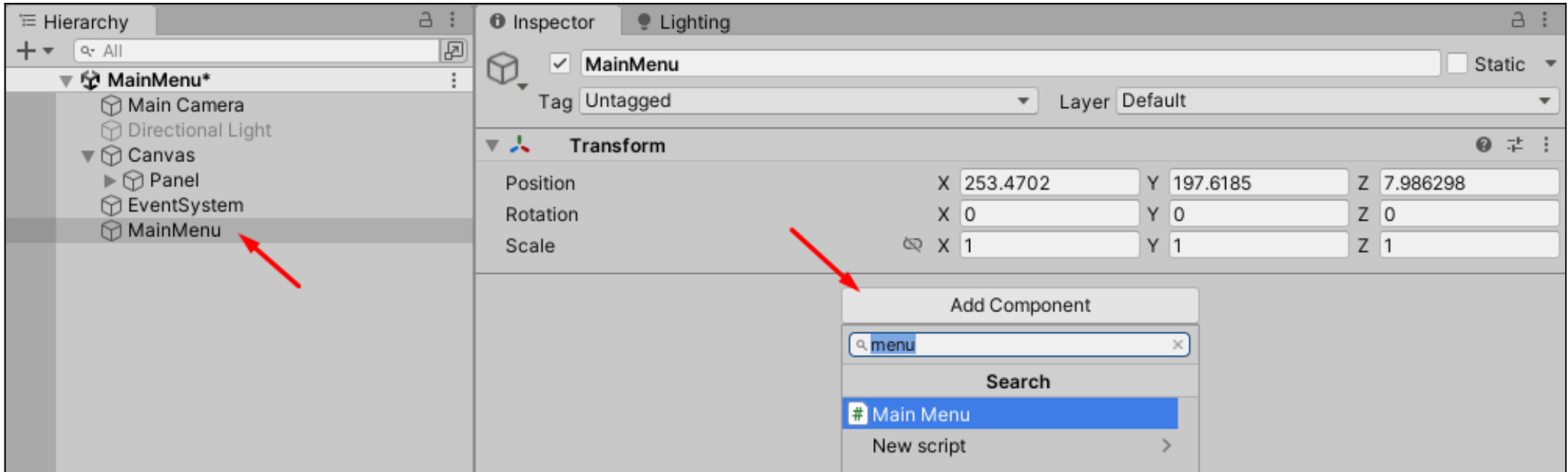
Создать **скрипт** для меню - объект типа **C# Script**, и переименовать в **MainMenu**.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour
{
    public void PressedStartGame()
    {
        SceneManager.LoadScene("GameScene"); // GameScene - указать имя сцены, в которую надо загружать игрока
    }

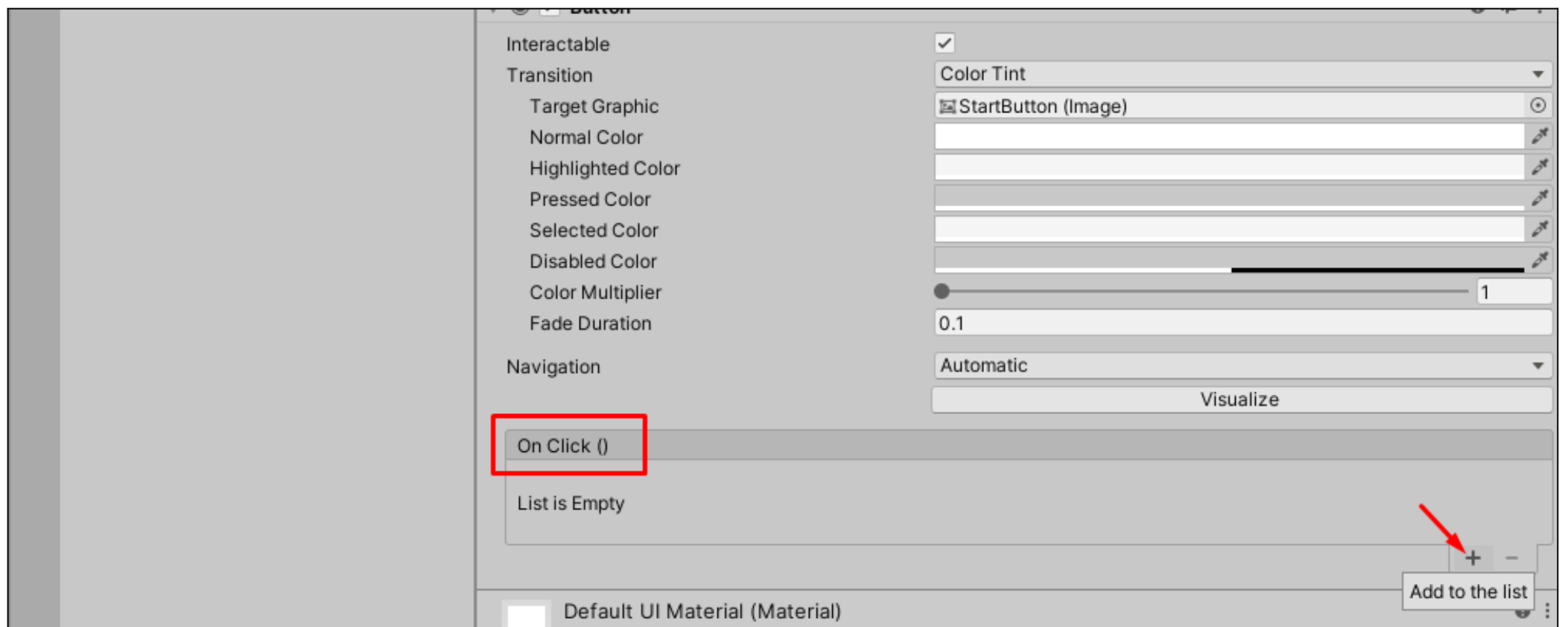
    public void PressedExit()
    {
        Application.Quit();
    }
}
```

В сцене **MainMenu** создать пустой объект **Create Empty**, переименовать в **MainMenu** и добавить ему компонент - созданный скрипт:

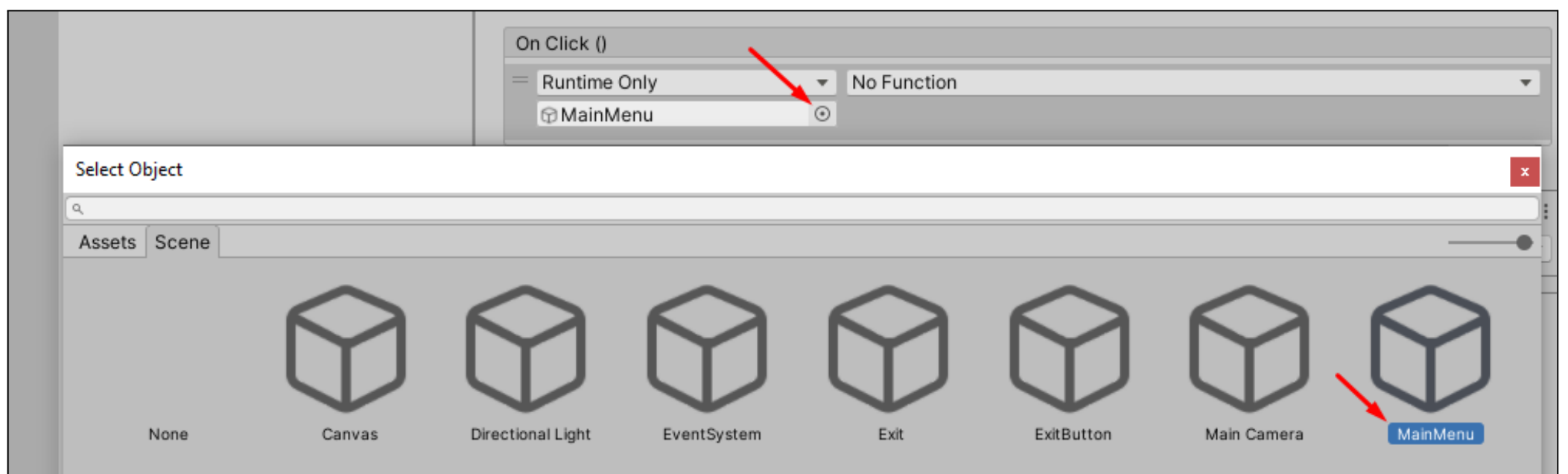


Объекты сцены можно сгруппировать в иерархии для удобства, но объект **Panel1** должен оставаться вложен в **Canvas**, а кнопки в панели.

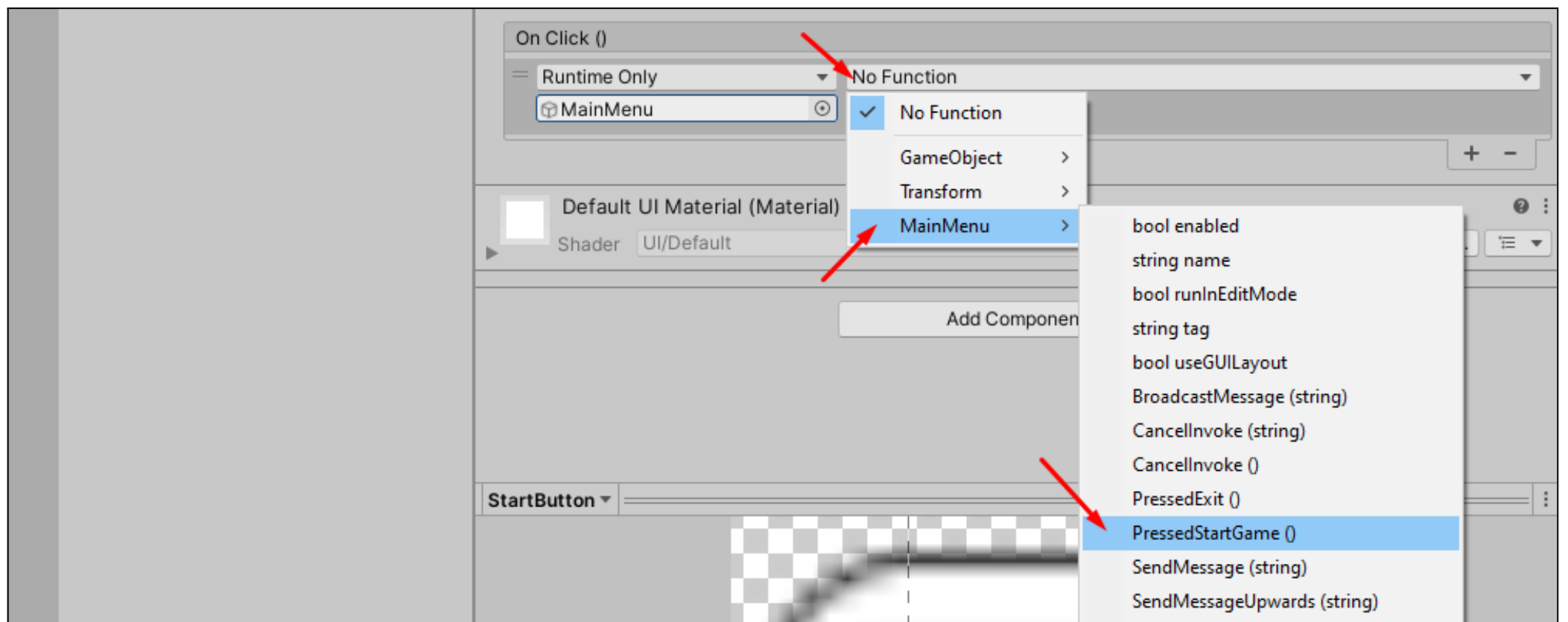
У объекта-кнопки **StartButton** в области **On Click()** добавить слот:



В нижнем окне созданного слота в качестве целевого объекта выбрать объект **MainMenu**:



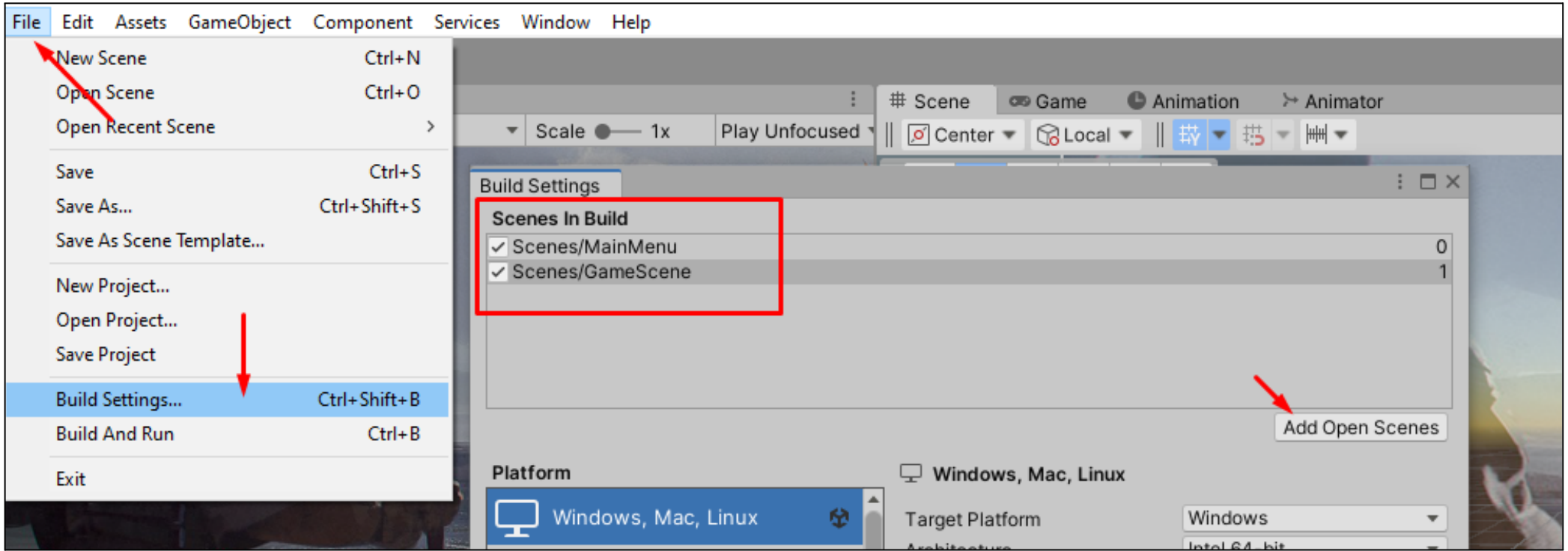
Выбрать функцию из созданного ранее скрипта, которая отвечает за загрузку сцены - **PressedStartGame()**:



Аналогично сделать для кнопки **Exit** - функция **PressedExit**.

※ Параметры сборки

В меню редактора **File / Build Settings** добавить задействованные сцены – сцена с меню, и сцена, в которую будет загружен игрок. Через кнопку **[Add Open Scenes]** можно добавить активную сцену, поэтому потребуется последовательно заходить в нужные сцены. Сцены можно перетаскивать из окна проекта в окно настройки сборки:



После проделанных действий при запуске игрового режима в сцене с меню на экране отобразится созданные кнопки. По нажатию на кнопку **[Start game]** будет загружена сцена GameScene. Кнопка **[Exit]** при тесте из Unity ничего не сделает.

※ Элементы UI

- Canvas** – холст
- Event System** – обработчик событий взаимодействия с интерфейсом
- Image** – картинка
- Text** – текстовая надпись – неинтерактивный фрагмент текста
- Raw Image** – выводит любую текстуру, а не только спрайт (конвертировать в спрайт не обязательно)
- Panel** – панель
- Toggle** – переключатель (чекбокс)
- Slider** – ползунок (при изменении текущего значения генерирует событие **OnValueChanged**)
- Scrollbar** – полоса прокрутки
- Scroll View** – прокрутка полотна текста
- Button** – кнопка
- Dropdown** – выпадающий список (для выбора пользователем одного элемента из списка, генерирует событие **OnValueChanged**)
- Input Field** – поле для ввода текста