Eadon Dice Roller

A dice rolling and task check system

Version 1.0

Table of Contents

Introduction

Eadon Dice Roller is a system to roll physically accurate dice on screen to simulate classic RPG mechanics. It includes a skill check component.

The following functionalities are implemented:

- Dice are rolled in a separate additively loaded scene with isolated physics
- Dice are physically accurate
- Multiple dice materials
- Customizable dice tray aspect
- Dice rolling sounds

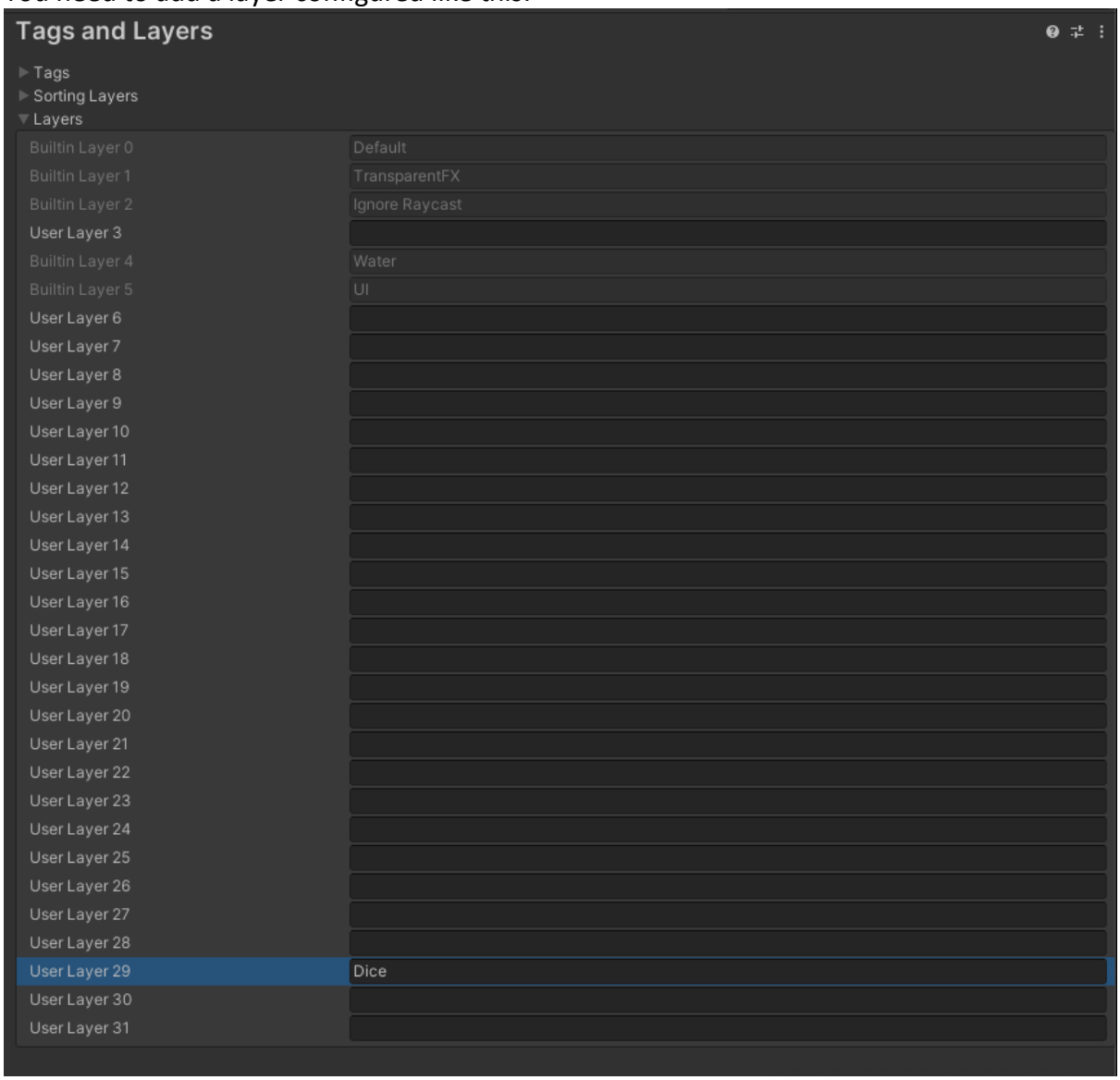
Changelog

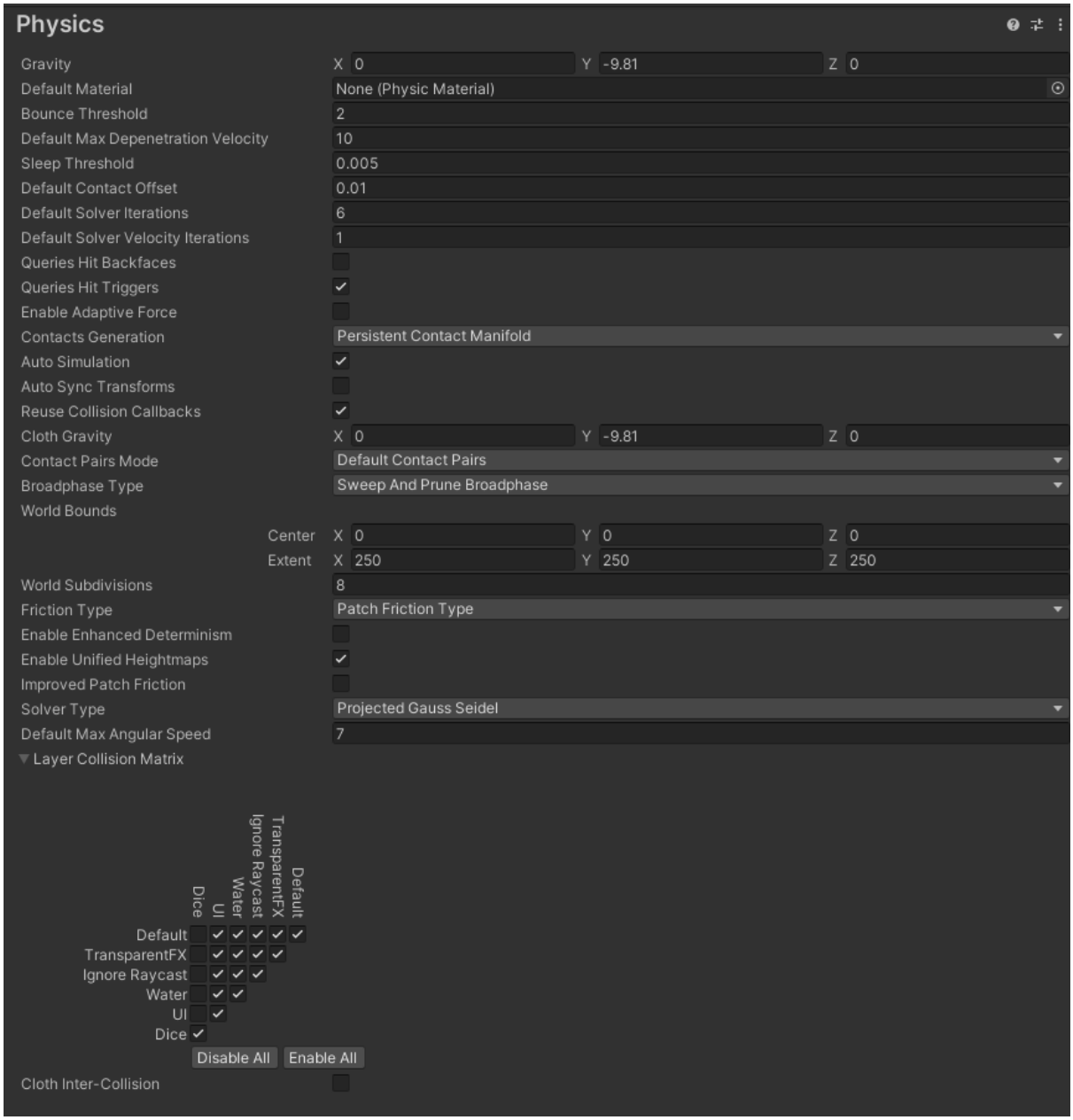
v 1.0	Initial release

Prerequisites

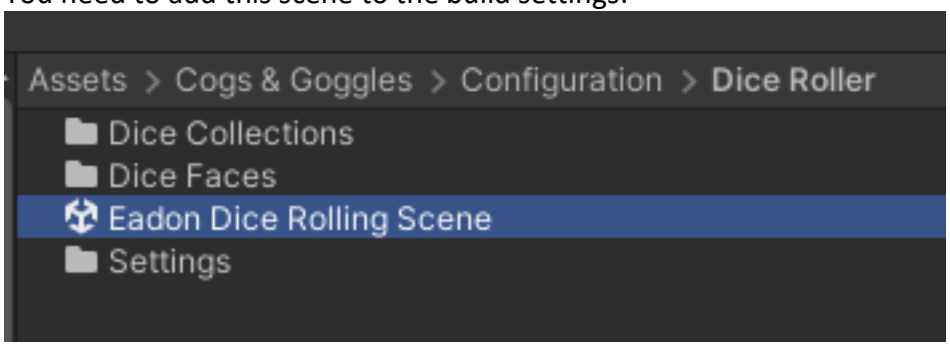
There are three prerequisite steps you need to perform to successfully run this asset:

1) You need to add a layer configured like this:





2) You need to add this scene to the build settings:



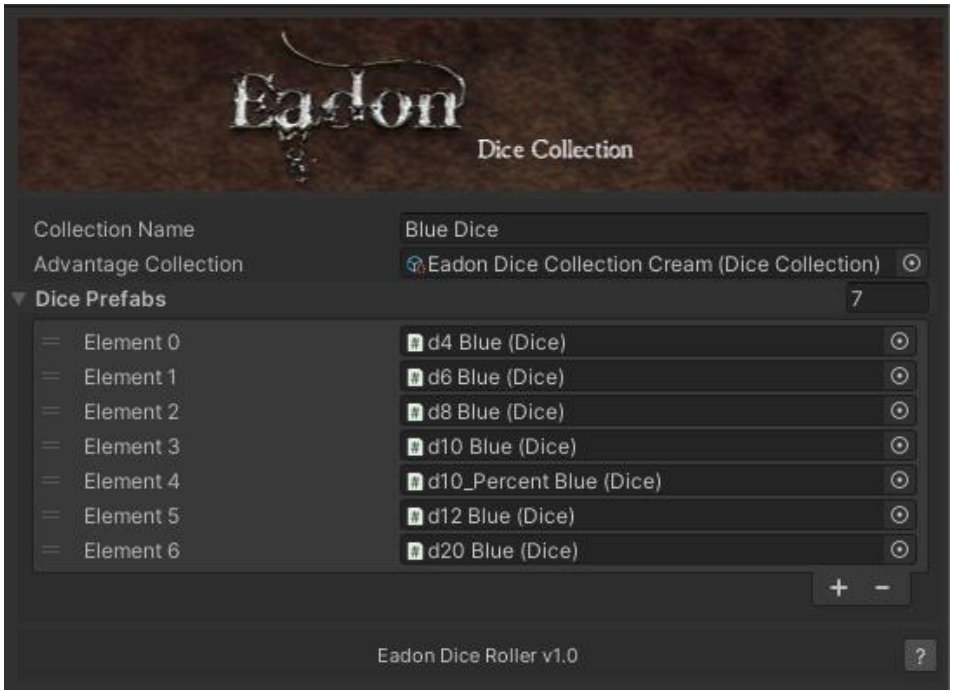
The demo scene and the dice prefab are preconfigured to use layer 29. If you need to use a different layer, please see the chapter on changing layers.

Configuration

Before the system can be used, it must be configured. Eadon Dice Roller is configured through a set of ScriptableObjects that determine what types of dice are available to the system, called dice collections. Dice collections can be created using the command found under the **Assets -> Create -> Eadon Dice Roller -> Dice Collection** menu.

Dice Collections

A DiceCollection looks like this:

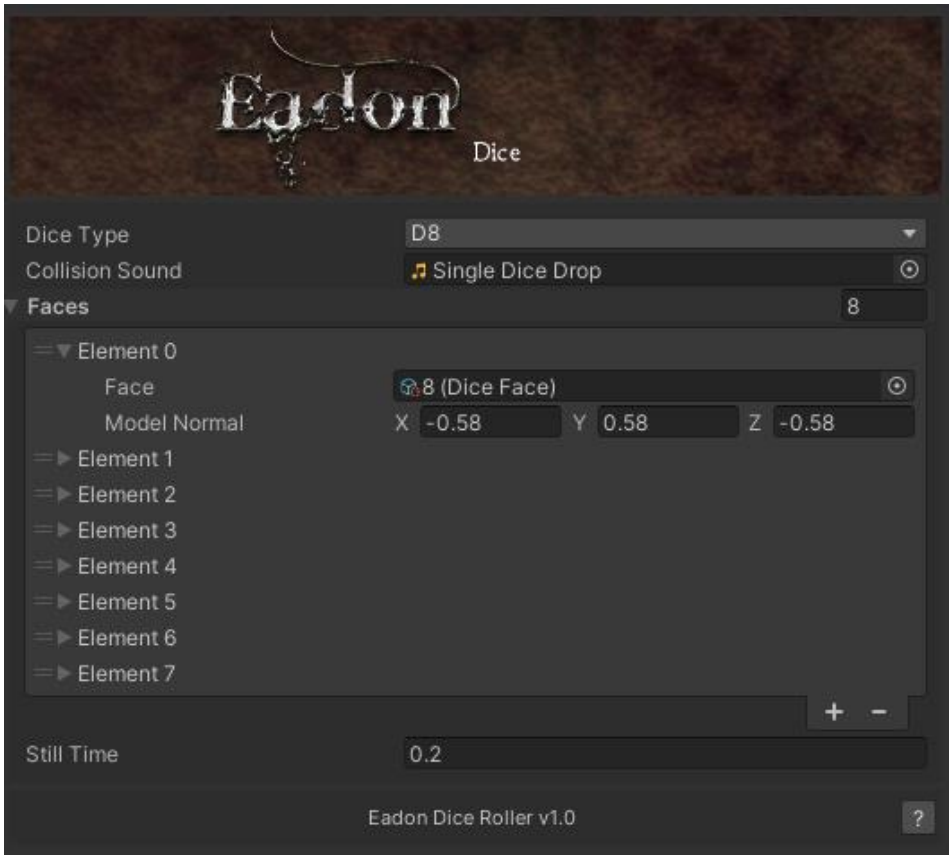


The fields are:

Field name	Purpose
Collection Name	The name of the collection. Should be unique across the range of collections in your project
Advantage Collection	The collection used to display advantage/disadvantage dice in a roll (see the chapter on Roll System
Dice Prefabs	The list of dice prefabs in the collection

Dice

A Dice looks like this:

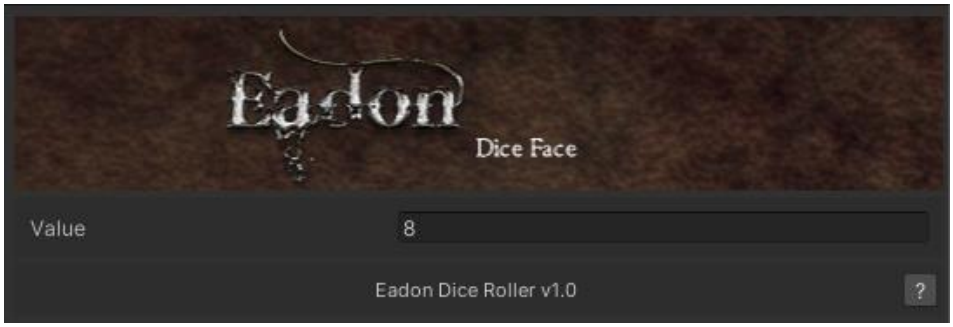


The fields are:

Field name	Purpose
Dice Type	Choice of D4, D6, D8, D10, D12, D20 and D100
Collision Sound	The sound to play when a dice collides with another or the borders of the roll area
Faces	A list of the faces (value and normal direction)
Still Time	The time after which a dice is considered stationary and not rolling when the rigidbody velocity is close to zero. Higher values can improve the simulation but increase the time it takes to get a roll result

Dice Faces

A DiceFace contains the value of the face, and looks like this:

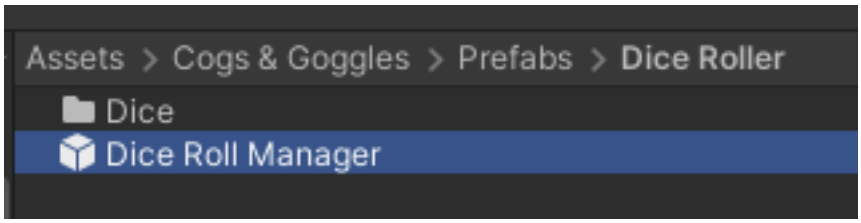


The fields are:

Field name	Purpose
Value	The numeric value of the face

Roll system

In order to trigger rolls, you need to drag the **Dice Roll Manager** prefab into your scene. Yhe prefab is located here:



It contains the DiceRollManager component, which looks like this:

Field name	Purpose
Dice Roll Scene	The name of the dice rolling scene
Dice Collections	The available dice collections
Rolling Table Background	The list of all possible rolling table background images

The DiceRollManager is the component that receives roll requests, loads the rolling scene if needed, triggers the physical dice roll and receives the result. You can request dice rolls from your code through a simple API.

Dice Roll Request API

The component requesting the roll needs to implement an interface in order to be notified of the roll events.

The interface is this:

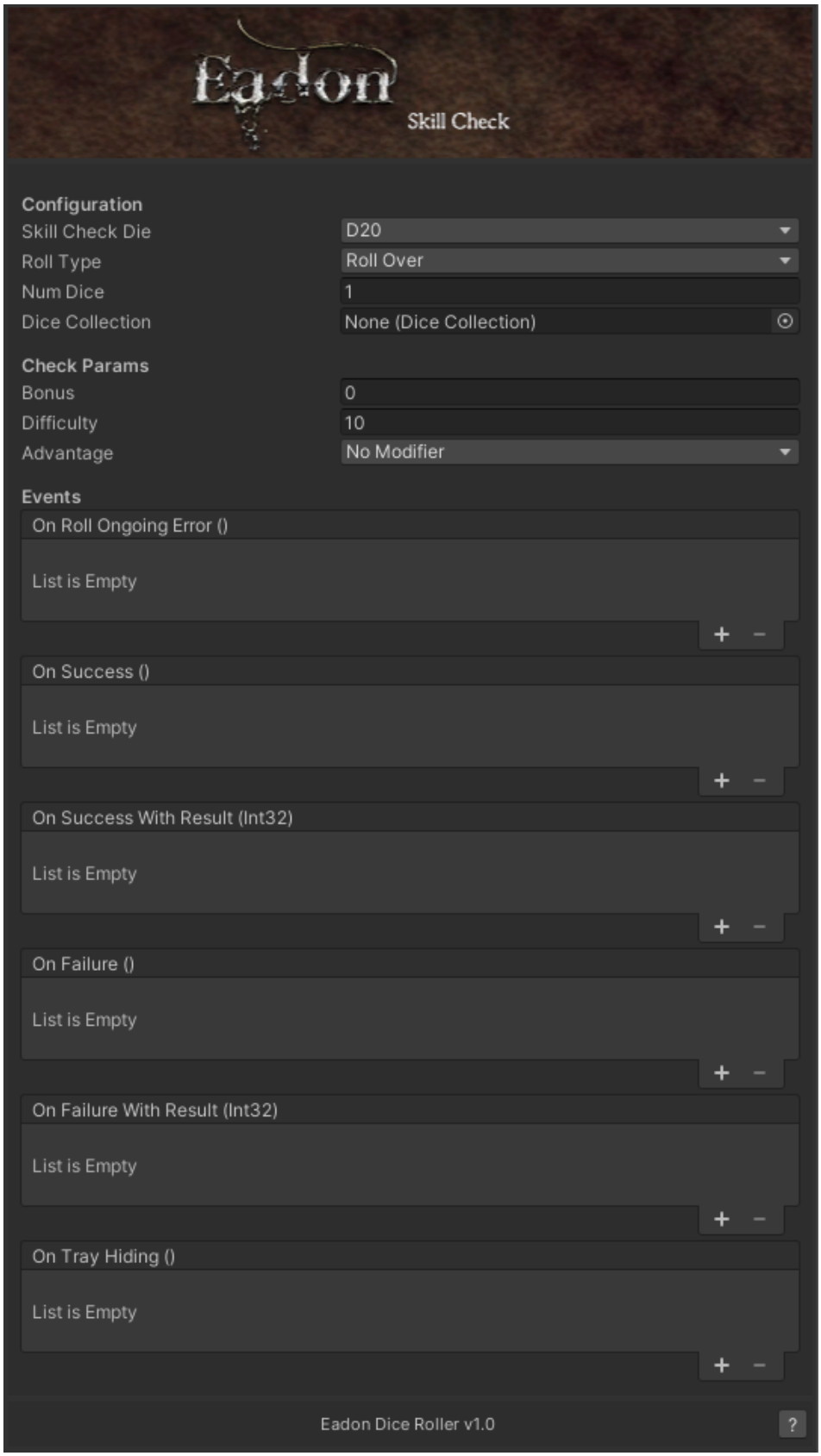
```
using System.Collections.Generic;
using Eadon.DiceRoller.Data;

namespace Eadon.DiceRoller.Components
{
    public interface IRollRequester
    {
        void ProcessResults(List<DiceResult> results);
        void TrayFading();
    }
}
```

Your component needs to implement the two methods. The first receives a list of DiceResult data structures (score and dice type) and is invoked when the roll is done (i.e. when all the dice are stationary). The second is invoked when the dice rolling tray starts to disappear.

As an alternative, if you don't want to implement the API you can use the SkillCheck component detailed in the next chapter.

Skill Check

The SkillCheck component implements a skill system that let you invoke skill checks quickly and provides events. It looks like this:



The fields are split into three groups: Configuration (which determines how a skill roll works), Check Params (which determines the detail of the roll) and Events.

Field name	Purpose
Skill Check Die	The type of the dice to roll
Roll Type	Choice of Roll Over, Roll Equal Or Over, Roll Under, Roll Equal Or Under
Num Dice	How many dice to roll (values are added together)
Dice Collection	The dice collection to use (if blank, the first in the Dice Roll Manager is used)
Bonus	The bonus to use for the roll
Difficulty	The target number for the roll
Advantage	If the character has any advantage/disadvantage
On Roll Ongoing Error	This event is triggered if you request a roll while another roll is ongoing
On Success	This event is triggered if the roll is successful
On Success With Result	This event is triggered if the roll is successful and passes the roll result
On Failure	This event is triggered if the roll is a failure
On Failure With Result	This event is triggered if the roll is a failure and passes the roll result
On Tray Hiding	This event is triggered when the dice tray starts to disappear

The advantage/disadvantage system works like this:

- If the character has advantage, an extra dice is rolled, then the lowest is discarded
- If the character has disadvantage, an extra dice is rolled and the highest is discarded
- Otherwise the normal number of dice is rolled and there is no modification

To trigger a roll, the component exposes two public methods:

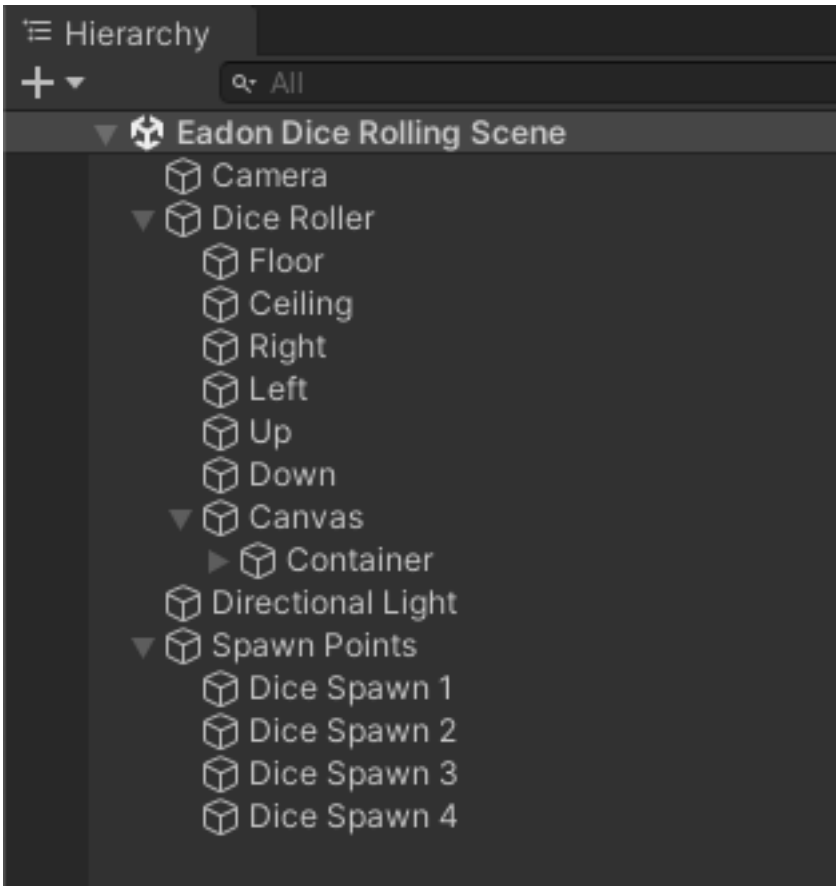
```
public void CheckSkill()

public void CheckSkill(int currentBonus, int
currentDifficulty, AdvantageType currentAdvantage);
```

The first triggers the skill check using the values in the inspector (which you can change at runtime by getting a reference to the SkillCheck component), while the second lets you pass the actual roll details in the arguments.

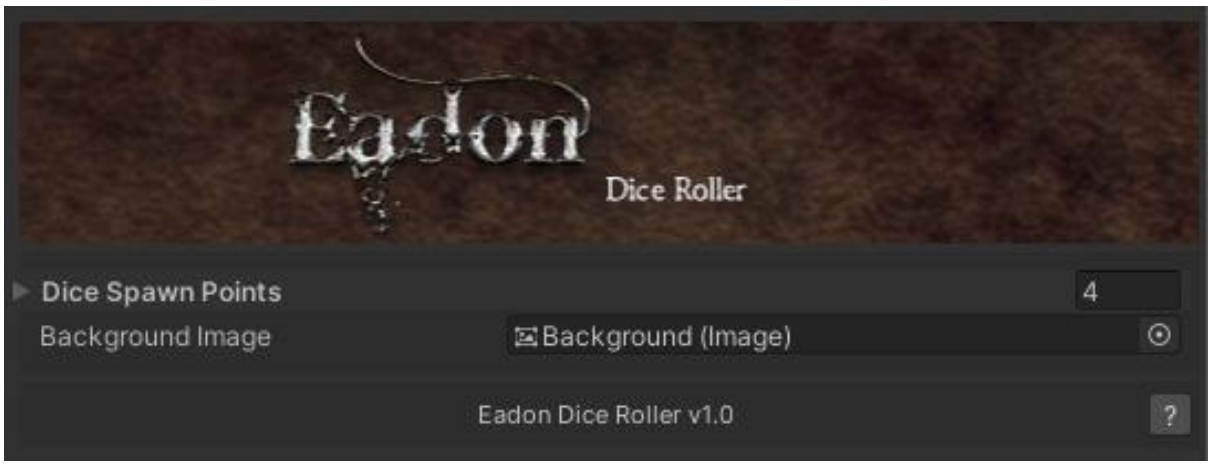
Rolling Scene Setup

The actual rolling of the dice happens in a separate scene which is loaded additively the first time you invoke a roll.

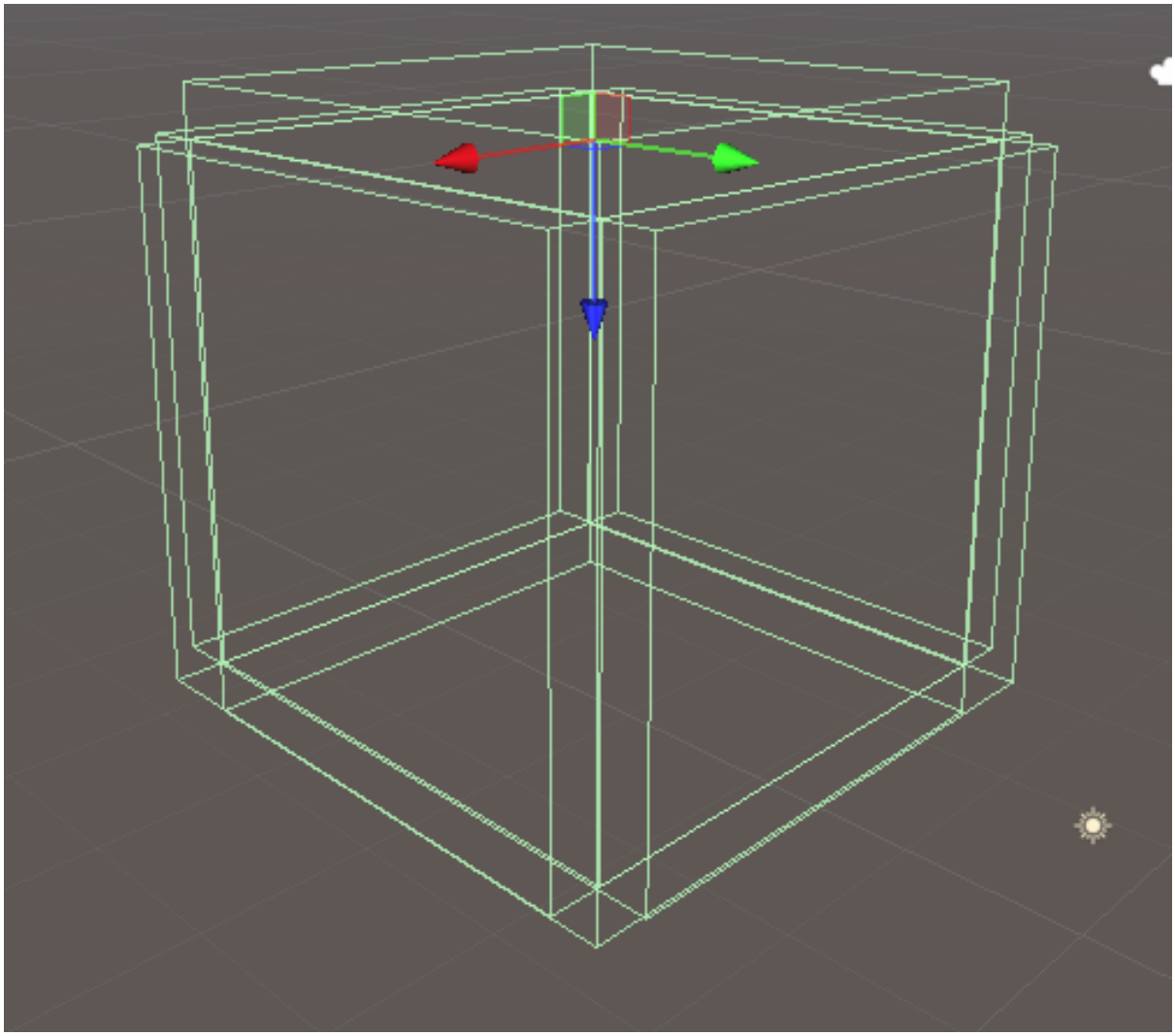
The scene is quite simple. It contains an orthographic camera, light only affecting the Dice layer and the Dice Roller setup:



The DiceRoller component contains references to the dice spawn points and the rolling table image:



The actual rolling of the dice happens within 6 colliders set up to look like a box:



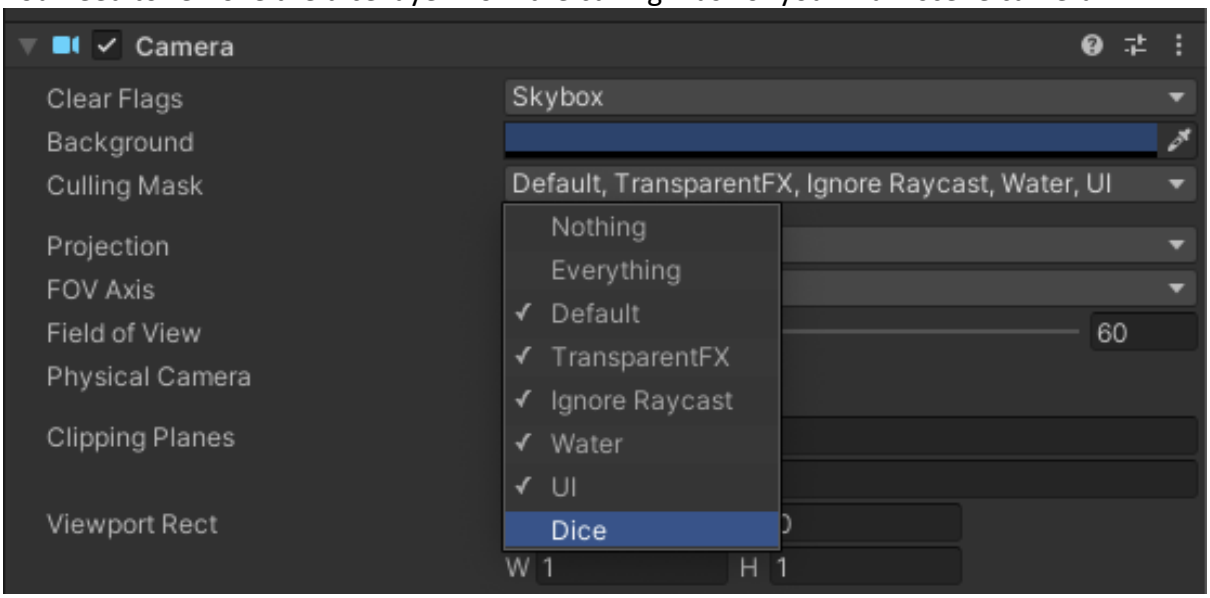
If you roll more dice that there are spawn points, the system will cycle through the spawn points as needed.

Changing Layers

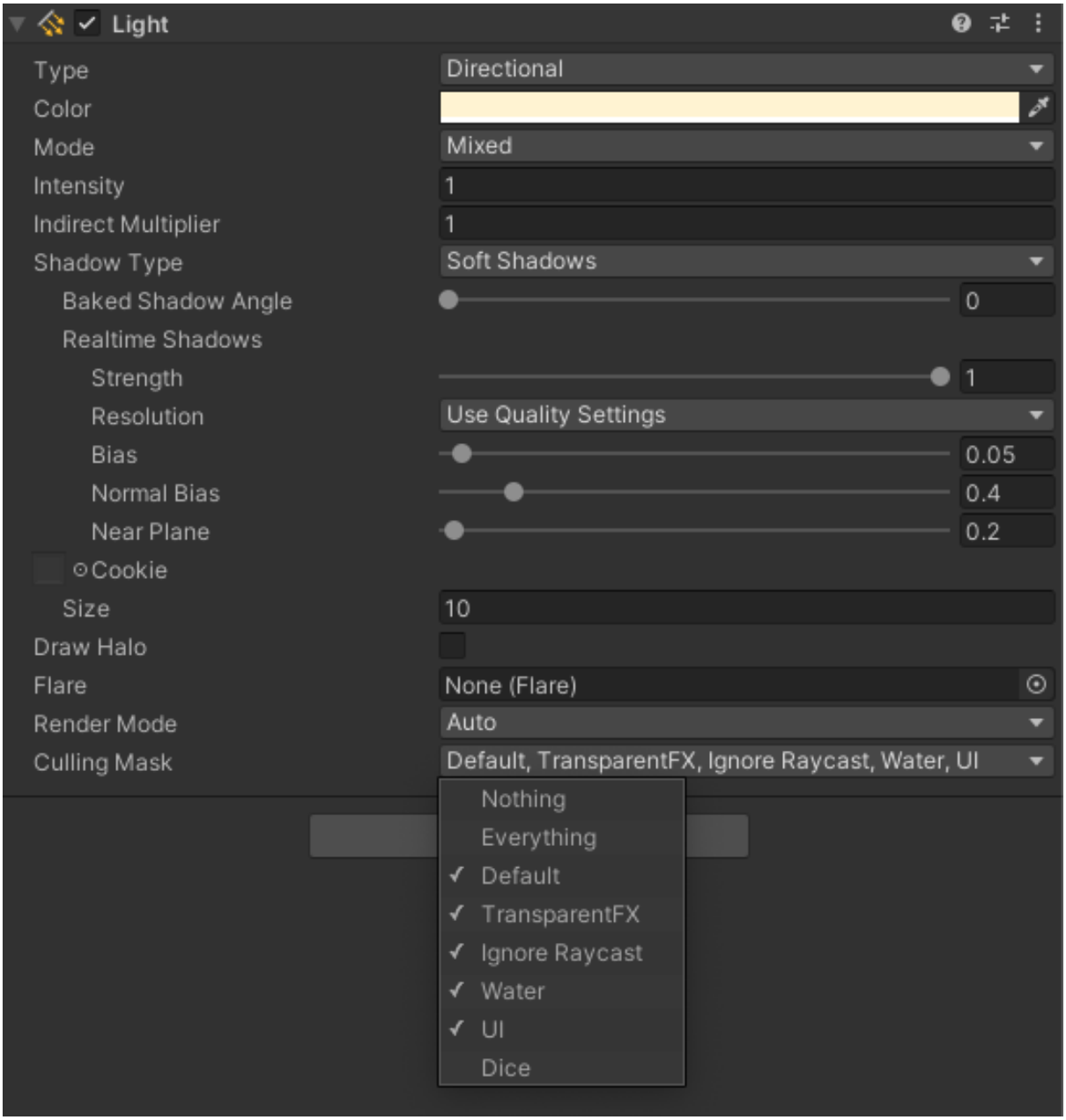
If you don't want to use the predefined layer 29 (as mentioned in the Prerequisite section), you can change it to any other layer. Please note that the dice rolling happens in a separate scene, but in the same physical space as your main scene. The layer you choose for the dice should only collide with the dice layer to avoid colliding with main scene objects.

Once you identify the layer you want to use, you need to:

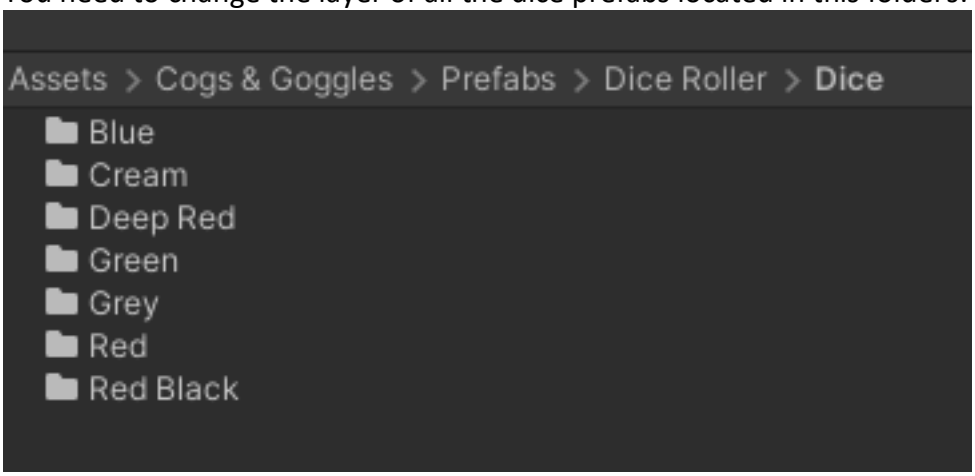
1) Add the layer to the layer list (as shown in the Prerequisite section)
2) Set up the physics collisions (as shown in the Prerequisite section)
3) You need to remove the dice layer from the culling mask of your main scene camera:



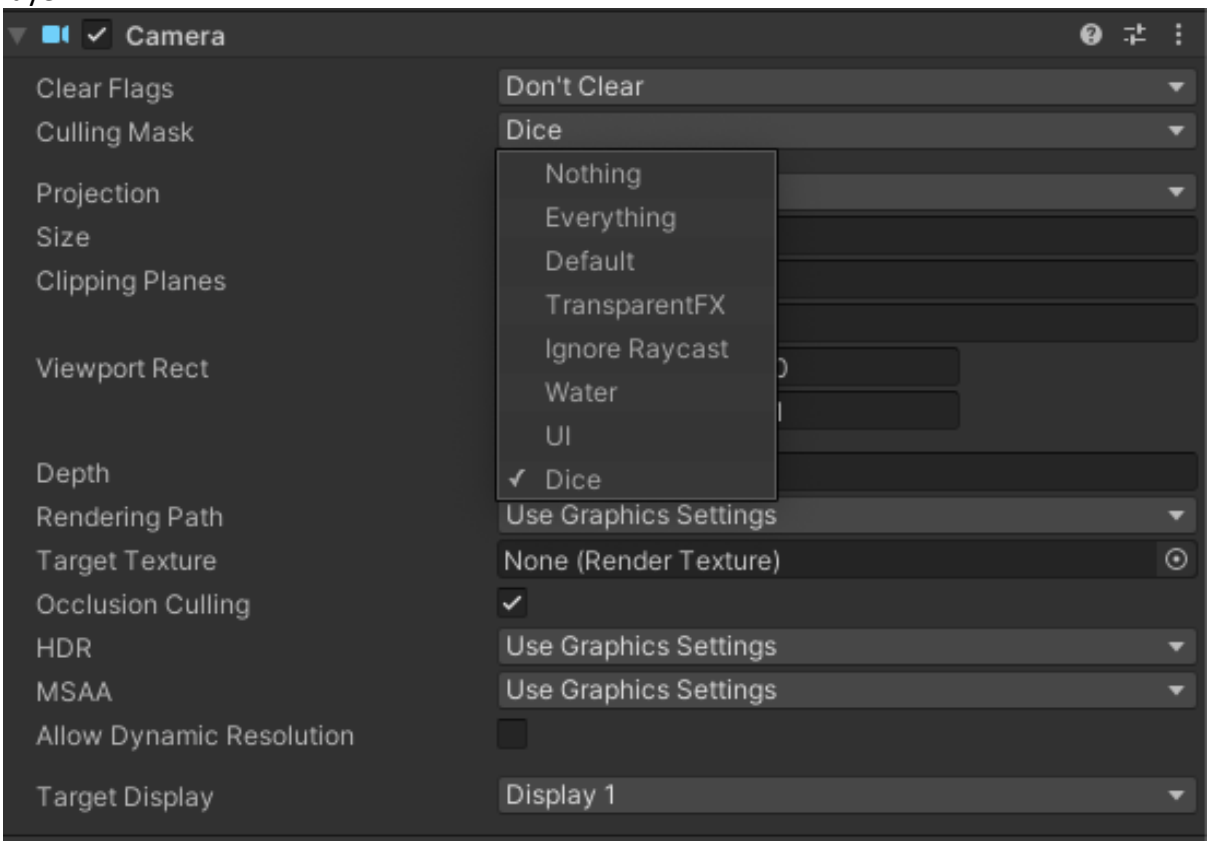
4) You need to remove the dice layer from your scene lights (or the dice will get double light):



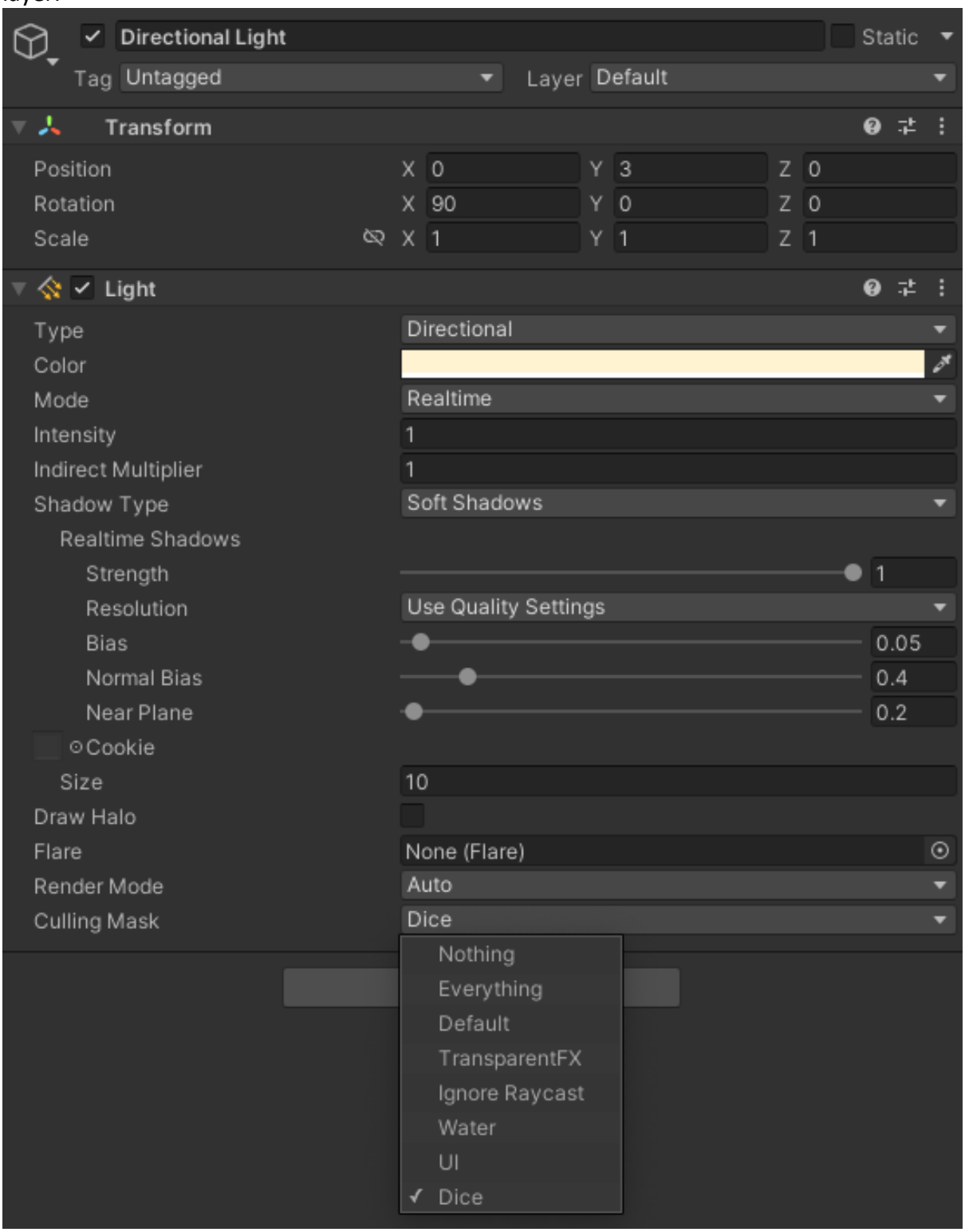
5) You need to change the layer of all the dice prefabs located in this folders:



6) Open the dice rolling scene
7) In the dice rolling scene you need to set the layer of the Dice Roller game object to the new layer (including children)
8) In the dice rolling scene you need to set the Culling Mask of the camera to the new layer:



9) In the dice rolling scene you need to set the Culling Mask of the light to only the new layer:



License

You agree that Cogs & Goggles own all rights, title and interest in this Asset, including without limitation all applicable Intellectual Property Rights. "Intellectual Property Rights" means any and all intellectual property rights wherever in the world and whenever arising (and including any application), including patent laws, copyright, trade secrets, know-how, confidential information, business names and domain names, computer programs, trademark laws, service marks, trade names, utility models, design rights, semi-conductor topography rights, database rights, goodwill or rights to sue for passing off, and any and all other proprietary rights worldwide. You agree that you will not, and will not allow any third party to,

(i) copy, sell, license, distribute, transfer, modify, adapt, translate, prepare derivative works from, decompile, reverse engineer, disassemble or otherwise attempt to derive source code from the Asset, unless otherwise permitted,

(ii) take any action to circumvent or defeat the security or content usage rules provided, deployed or enforced by any functionality (including without limitation digital rights management or forward-lock functionality) in the Asset,

(iv) remove, obscure, or alter Cogs & Goggles' or any third party's copyright notices, trademarks, or other proprietary rights notices affixed to or contained within the Unity Asset Store or Assets.

YOU EXPRESSLY UNDERSTAND AND AGREE THAT YOUR USE OF THE ASSET IS AT YOUR SOLE RISK AND THAT THE ASSET IS PROVIDED “AS IS” AND “AS AVAILABLE” WITHOUT WARRANTY OF ANY KIND, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW. IN PARTICULAR, COGS & GOGGLES, ITS SUBSIDIARIES, HOLDING COMPANIES AND AFFILIATES, AND ITS LICENSORS DO NOT REPRESENT OR WARRANT TO YOU THAT:

(A) YOUR USE OF THE ASSETS WILL MEET YOUR REQUIREMENTS,

(B) YOUR USE OF THE ASSETS WILL BE UNINTERRUPTED, TIMELY, SECURE OR FREE FROM ERROR,

(C) ANY INFORMATION OBTAINED BY YOU AS A RESULT OF YOUR USE OF THE ASSETS WILL BE ACCURATE OR RELIABLE, AND

(D) THAT DEFECTS IN THE OPERATION OR FUNCTIONALITY OF ANY SOFTWARE PROVIDED TO YOU AS PART OF THE ASSETS WILL BE CORRECTED.

YOUR USE OF THE ASSET IS AT YOUR OWN DISCRETION AND RISK AND YOU ARE SOLELY RESPONSIBLE FOR ANY DAMAGE TO YOUR COMPUTER SYSTEM, OR OTHER DEVICE, OR LOSS OF DATA THAT RESULTS FROM SUCH USE.

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, COGS & GOGGLES FURTHER EXPRESSLY DISCLAIMS ALL WARRANTIES TERMS OR CONDITIONS OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO ANY IMPLIED WARRANTIES TERMS AND CONDITIONS OF MERCHANTABILITY, SATISFACTORY QUALITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.