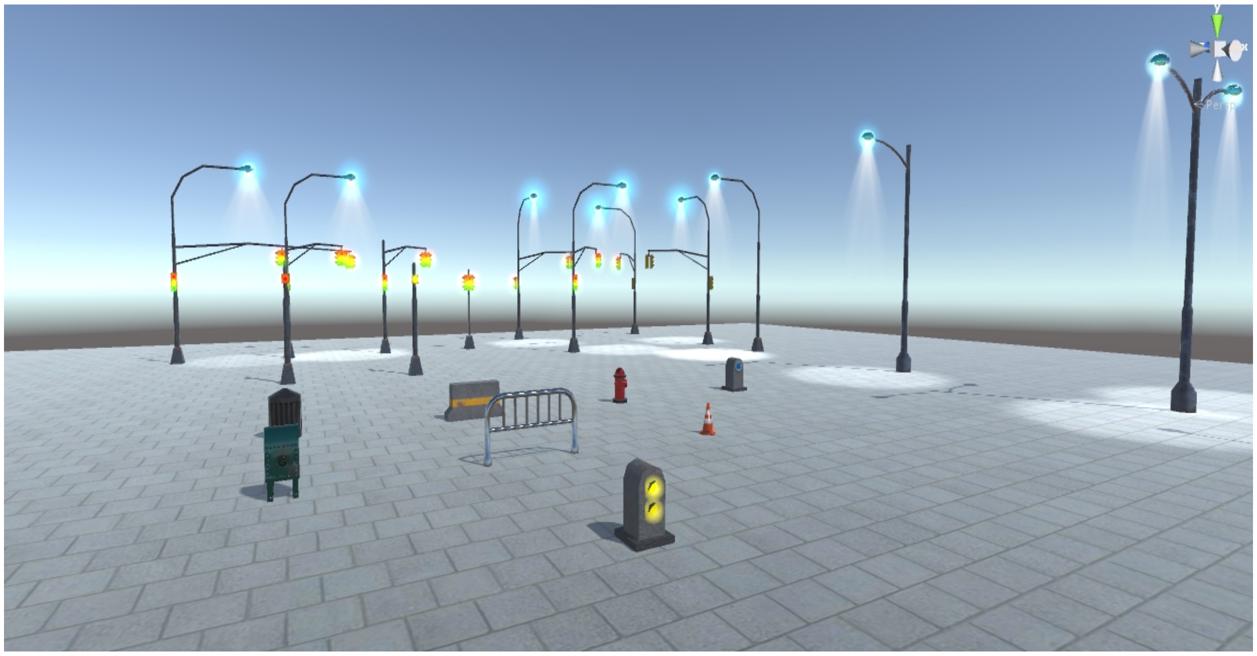
Dynamic Components

Documentation



Introduction

This manual contains information about the prefabs and scripts you get with the package. To start with, a demo scene is provided with selected prefabs. You just need to drag and drop prefabs in your scene as they all are pre set up and do not need any further adjustments from the users side, however, the working of scripts will be discussed in this manual. You need to collide objects in order to see the dynamic effects.

All prefabs are located in the **PackageAssets > Prefabs** folder

There are further 2 subfolders viz. **StreetComponents** and **TrafficLightSystems**.

The **StreetComponents** folder contains all prefabs while **TrafficLightSystems** folder contains working prefabs for 4 way crossing traffic lights.

All 3D models are located in the **3DModels** folder. They are Sketchup (skp) files and contain meshes for different LODs and colliders.

Demo Scene

While in demo scene, use:

1) Arrow keys i.e. UP,DOWN,LEFT and RIGHT for navigation

2) Mouse for look-over

Scripts

1. CameraFacingBillboard.cs
2. CollisionDestroy.cs
3. defaultAudio.cs
4. lightBlink.cs
5. lightsControl.cs
6. TrafficLightControl.cs

These scripts will be discussed in detail

CameraFacingBillboard.cs

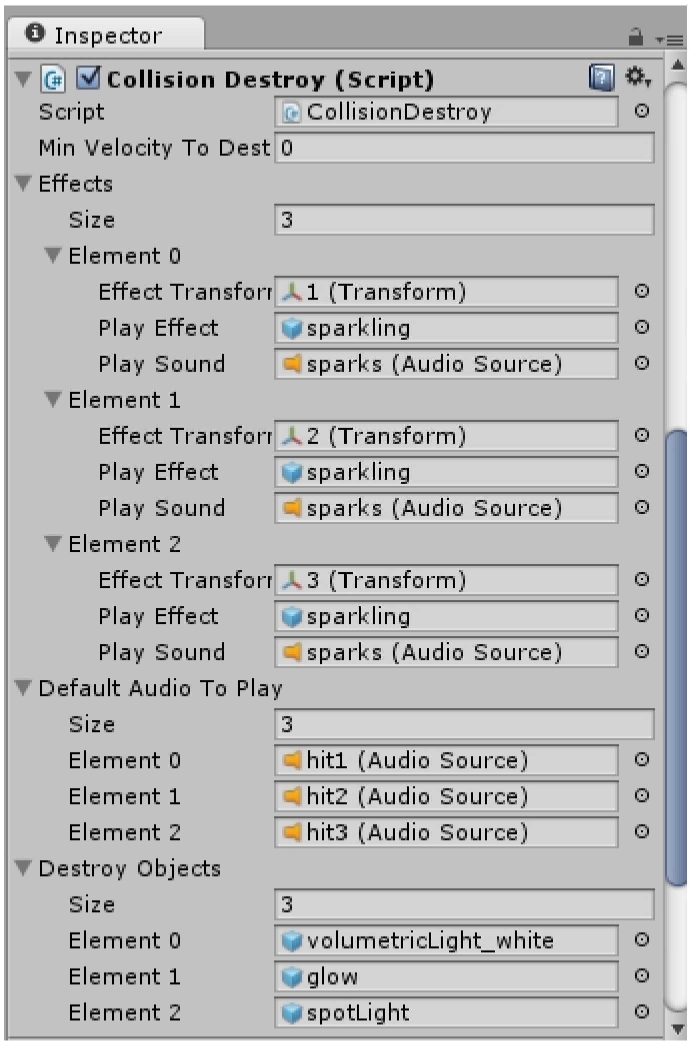
Purpose: To let the volumetric light mesh or any other quad mesh to face the camera rotation in order to give 3-D effect.

Usage: Drag and drop into the Game Object or Quad Mesh.

Important note: The script will search automatically for the main camera so please make sure that the active camera in the scene is tagged as “MainCamera”, or else error will occur.

CollisionDestroy.cs

Usage: This is the main script responsible for the dynamic behaviour of the objects. Its parameters are explained as follows:



Inspector

Collision Destroy (Script)

Script	CollisionDestroy
Min Velocity To Dest	0
Effects	
Size	3
Element 0	
Effect Transforr	1 (Transform)
Play Effect	sparkling
Play Sound	sparks (Audio Source)
Element 1	
Effect Transforr	2 (Transform)
Play Effect	sparkling
Play Sound	sparks (Audio Source)
Element 2	
Effect Transforr	3 (Transform)
Play Effect	sparkling
Play Sound	sparks (Audio Source)
Default Audio To Play	
Size	3
Element 0	hit1 (Audio Source)
Element 1	hit2 (Audio Source)
Element 2	hit3 (Audio Source)
Destroy Objects	
Size	3
Element 0	volumetricLight_white
Element 1	glow
Element 2	spotLight

Parameters:

1) **Min Velocity to Destroy:** At which minimum velocity should the destruction of object occur? It can be 0. If say it is set to 10, then the object colliding with the prefab will destruct the prefab if its velocity is greater than or equal to 10.

2) **Effects:** To specify special effects or particle system such as sparks. It is a class and always has a corresponding:
    1. **Effect Transform:** At which position in the local space the effect or the particle system should play?
    2. **Play Effect:** Which particle system prefab or game object to play?
    3. **Play Sound:** Play a sound effect corresponding to that prefab.

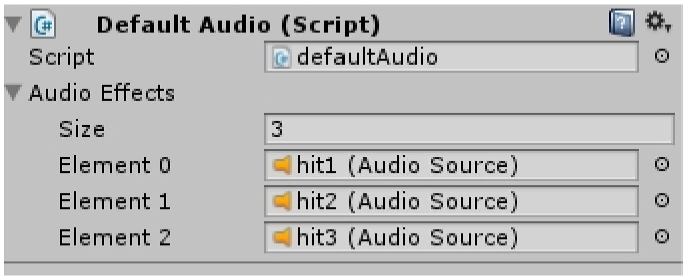
3) **Default Audio:** Its size must be greater than 0 which means there must be an audio effect to play. Specify the size of array of audio effects and one audio effect from that array of audio effects will randomly play.

4) **Destroy Objects:** Suppose you want to destroy some objects after the collision has occurred e.g. in Street Lights, the lens flares, halo, volumetric lights and spot lights should switch off. If you want such functionality, then you can specify which objects to destroy after the collision.

- Please note that this script must be attached to that game object (preferably root object) which contains the trigger collider

DefaultAudio.cs

Usage: This script is an extension of CollisionDestroy.cs. It is similar to the Default audio parameters of CollisionDestroy.cs. All Audio effects you specify, randomly one of them will always play on collision. Parameters are same as that of Default Audio to play of CollisionDestroy.cs



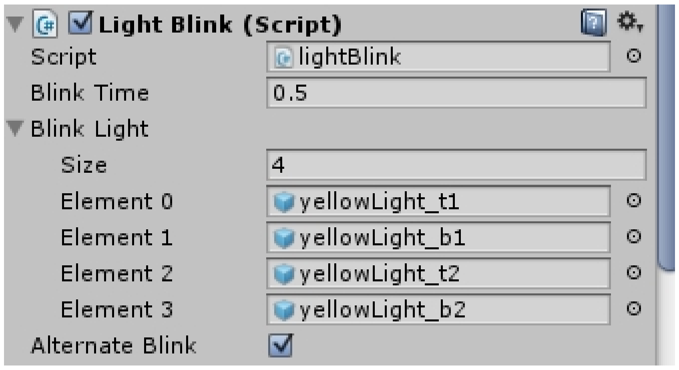
Default Audio (Script)

Script	defaultAudio
Audio Effects	
Size	3
Element 0	hit1 (Audio Source)
Element 1	hit2 (Audio Source)
Element 2	hit3 (Audio Source)

- Please note that this script must be attached to those child objects which contain a convex non-trigger collider.

LightBlink.cs

Purpose: To blink a game object (light) at a specified frequency. This script can be attached to root object.



Light Blink (Script)

Script	lightBlink
Blink Time	0.5
Blink Light	
Size	4
Element 0	yellowLight_t1
Element 1	yellowLight_b1
Element 2	yellowLight_t2
Element 3	yellowLight_b2
Alternate Blink	✓

Parameters:

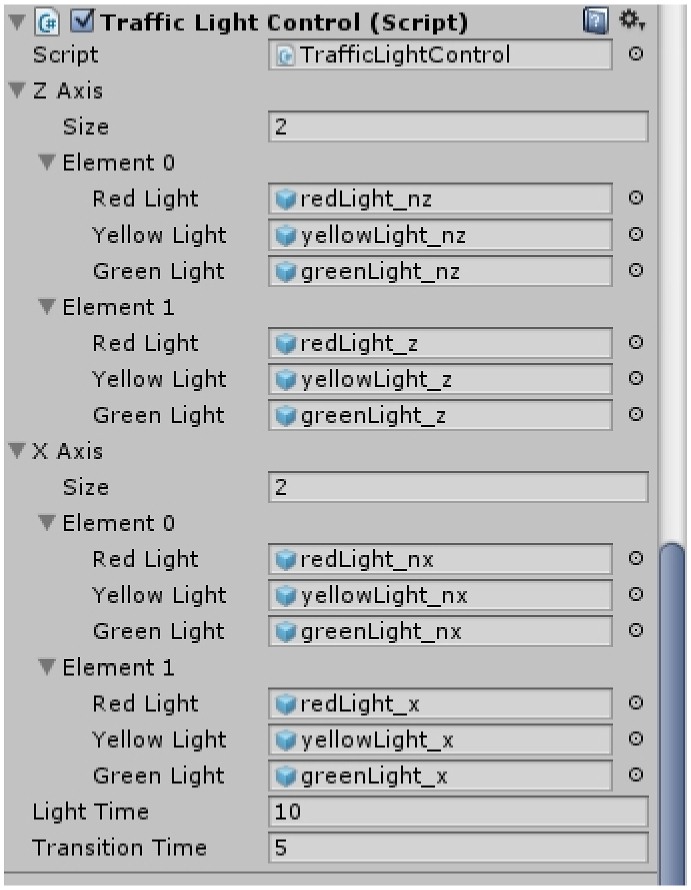
1) **Blink time:** Time interval between on and off operation of the light (game object).

2) **Blink Light:** Array of those game objects which you want to blink

3) **Alternate blink:** This is a Boolean value. If set to true then those game objects at even index of array will blink first and then the odd indices ones. This means if there are 2 lights, then 1st will be on, 2nd will be off. Then 2nd will be on and 1st will be off and vice versa. See “FlashingLights” prefab.

TrafficLightControl.cs

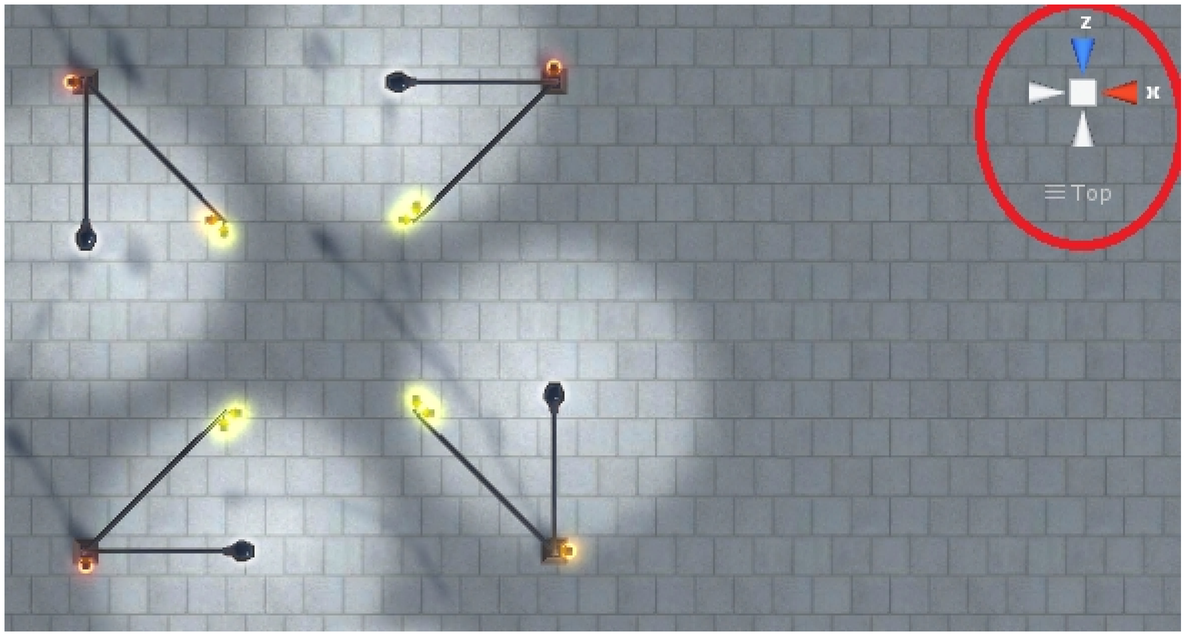
Purpose: To control the traffic lights behaviour.



Traffic Light Control (Script)

Script	TrafficLightControl
Z Axis	
Size	2
Element 0	
Red Light	redLight_nz
Yellow Light	yellowLight_nz
Green Light	greenLight_nz
Element 1	
Red Light	redLight_z
Yellow Light	yellowLight_z
Green Light	greenLight_z
X Axis	
Size	2
Element 0	
Red Light	redLight_nx
Yellow Light	yellowLight_nx
Green Light	greenLight_nx
Element 1	
Red Light	redLight_x
Yellow Light	yellowLight_x
Green Light	greenLight_x
Light Time	10
Transition Time	5

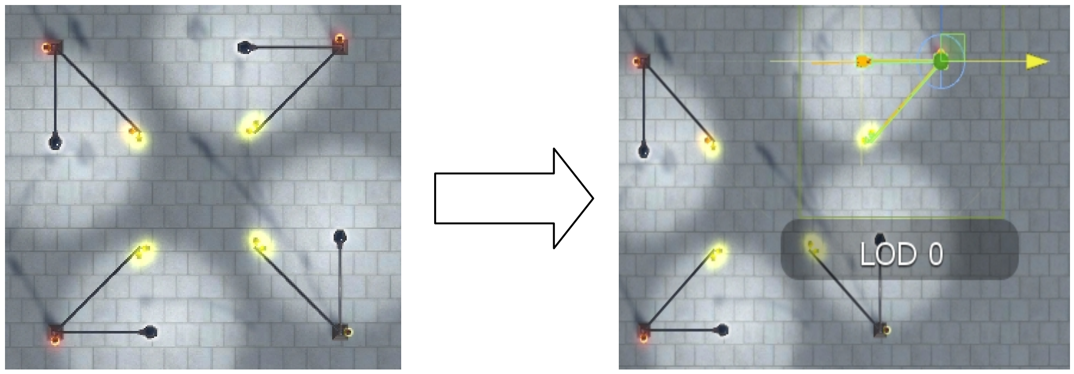
Usage: Observe the traffic lights from parallel projection top view.



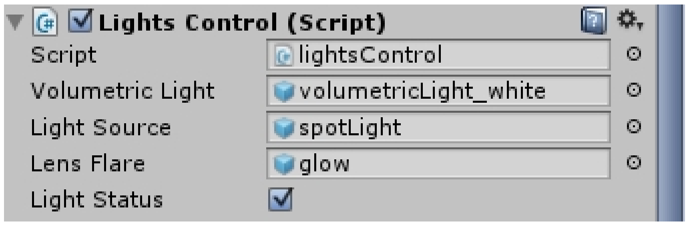
1) In **Z Axis** and **X Axis**, specify the length of array of light boxes which are facing the respective direction when observed from the top view
2) A light box is a class and always has a red light, yellow light and green light. (Refer script)
3) Put all red lights, yellow lights, and green lights facing positive X and negative X directions into their corresponding columns in **X axis** parameter.
4) Similarly put all red lights, yellow lights, and green lights facing positive Z and negative Z directions into their corresponding columns in **Z axis** parameter.
5) Steps 3 and 4 must be done carefully. They already have been set up in prefabs.
6) **Light time:** Time in seconds for which red light and green lights will stay.
7) **Transition time:** Time in seconds for which only yellow light will stay.

Traffic Light prefabs

- Traffic light prefabs that have light boxes facing all directions are marked as _proto. They can be used for prototyping propose e.g. you can delete the light boxes in particular directions as per your need
- For <u>**LEFT HAND TRAFFIC,**</u> traffic light prefabs are marked as **_LHT.**
- For <u>**RIGHT HAND TRAFFIC**</u>, traffic light prefabs are marked as **_RHT**
- IMPORTANT: In order to implement a crossing system of 4 traffic lights, do not just keep on rotating and either prefab. This is because in the script, lights for Z axis and X axis were specified, on rotating them will change their directions and synchronization too, so you have to reconfigure the traffic lights in the script, however readymade crossing systems prefabs are also given for both <u>**LEFT HAND TRAFFIC**</u>, <u>**RIGHT HAND TRAFFIC**</u> and <u>**PROTOYPING**</u>.
- Drag and drop them and position their children (individual traffic lights as per your need)



Purpose: Suppose for day scenes, you want to temporarily disable volumetric lights, spot lights or flares etc. or only for some street lights, you do not want the volumetric lights and spotlights, then you can use this script. Refer to the scripts code. The functions are public so they can be accessed publically in your own scripts to dynamically change light's status according to time of day.



Lights Control (Script)

Script	lightsControl
Volumetric Light	volumetricLight_white
Light Source	spotLight
Lens Flare	glow
Light Status	✓

Parameters:

1) **Volumetric light**: Which volumetric light to control?
2) **Light Source**: Which spot light or projector to control?
3) **Lens Flare**: Which glow effect, lens flares or halo to control?
4) **Light status**: It is a Boolean value. If set to true, then all the above game objects will be activated, else if set false, all of them will be deactivated.

THANKS FOR PURCHASING THE ASSET

For any queries or suggestions, please contact the publisher at sukhvisukh@gmail.com.

We may add more street components in the future upgrades according to your suggestions.

Please rate and review the asset on the asset store ☺

Thanks