
PMC Docs

Michael Pollind

Jun 25, 2020

TUTORIALS

1	Attractions	1
1.1	Flat Ride	1
1.2	Train	7
2	Walls	13
2.1	Fence	13
2.2	Wall	14
3	Path Attachments	17
3.1	Bench	17
3.2	Lamp	19
3.3	Sign	19
3.4	Trashbin	21
3.5	TV	21
4	Parkitect Asset Editor	25
4.1	Light	25
4.2	Materials	29
4.3	Custom Color	32
5	Indices and tables	33

ATTRACTIONS

1.1 Flat Ride

Rides that run through a certain animation cycle.

1.1.1 Color Settings

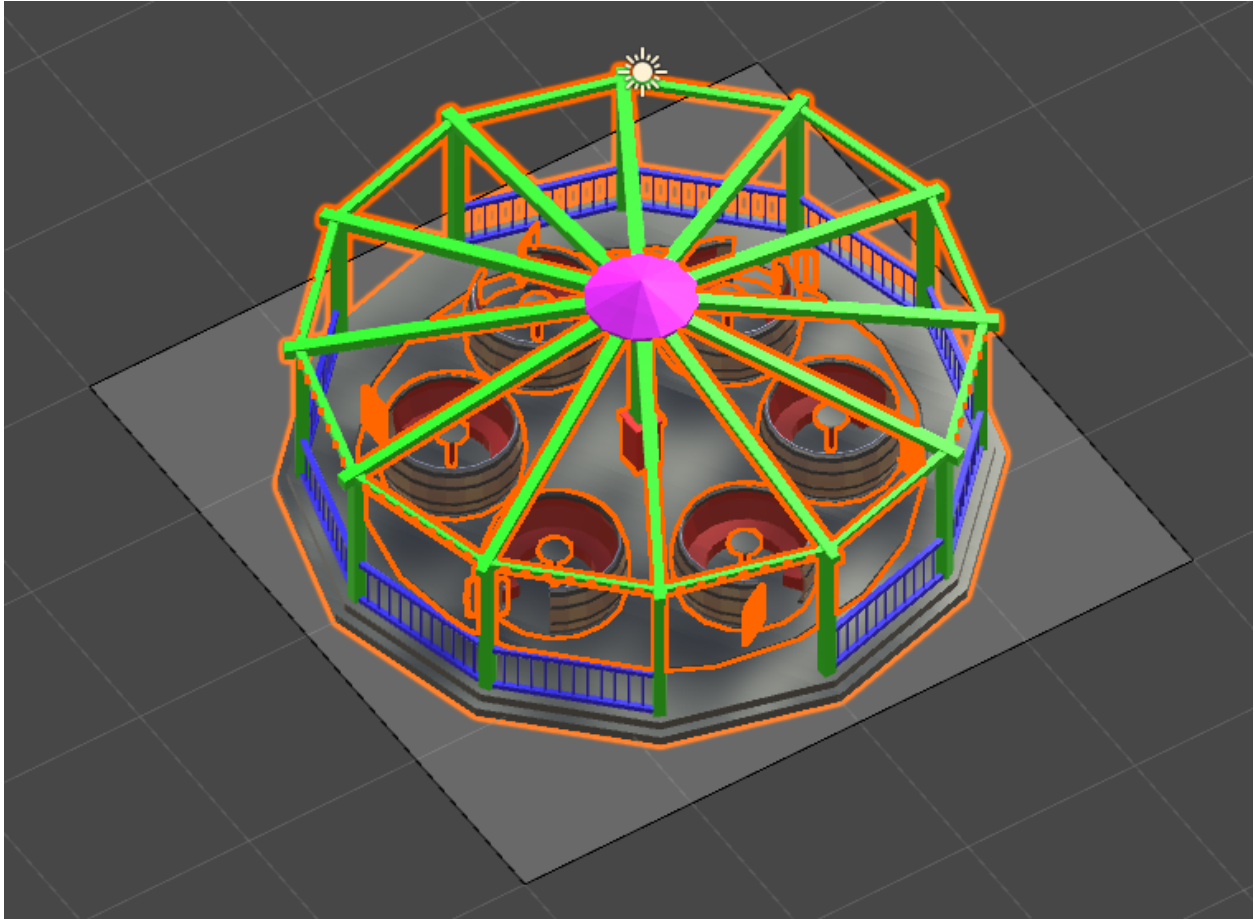
Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

1.1.2 Rating

Pick **Excitement**, **Intensity** and **Nausea** values for your ride. They are pretty much arbitrary - check comparable rides that are already part of the game to get an idea for what these values should roughly be.

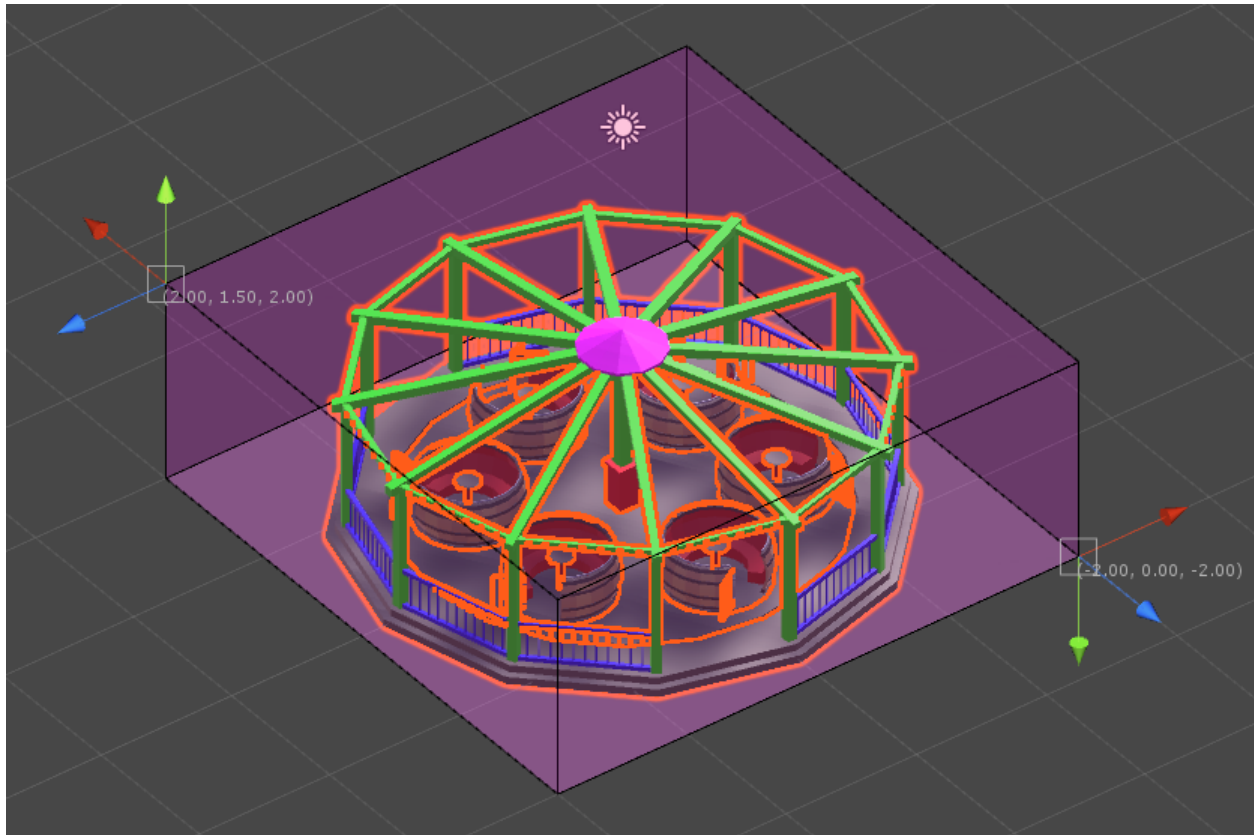
1.1.3 Ride Footprint

The size of the rides platform. Make sure it fully encloses the ride while still leaving enough space for guests to walk around it. Keep in mind that the ride entrance and exit can be placed on any tile so guests need to be able to navigate from the entrance to any seat and from any seat to the exit, no matter where the entrance and exit are placed.



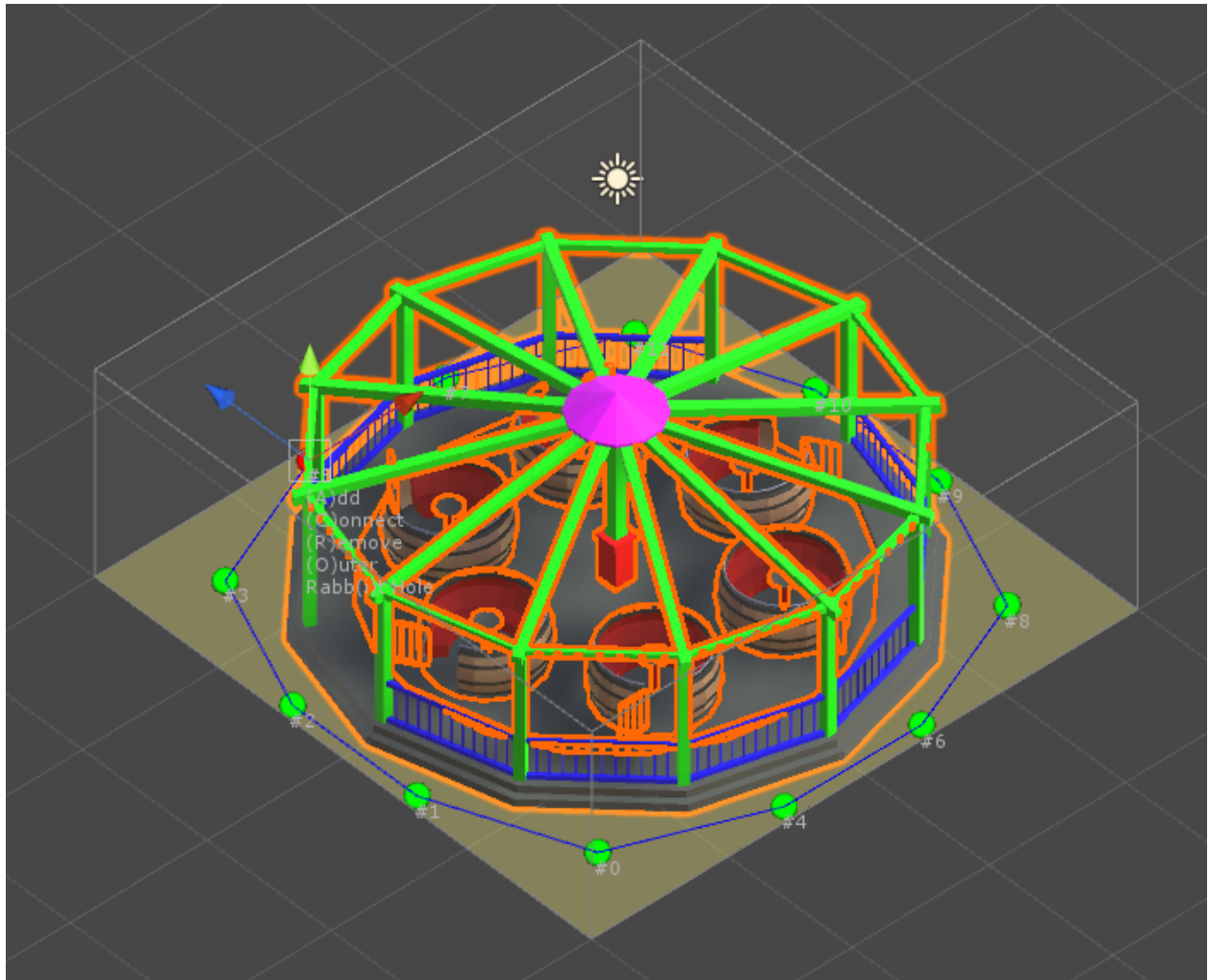
1.1.4 Collisions

Add one or more bounding boxes to your ride to define its clearance. No other ride, path or building is allowed to intersect with these bounds. Hit the “Add bounding box” button, then select the newly created bounding box from the list to edit it. While editing a bounding box you can hold the “S” key to more easily snap it to tile bounds. Make sure there is at least a 1 unit tall box that fully encloses the ride platform. Add more boxes as needed to block every area the ride touches when going through its animation cycle.

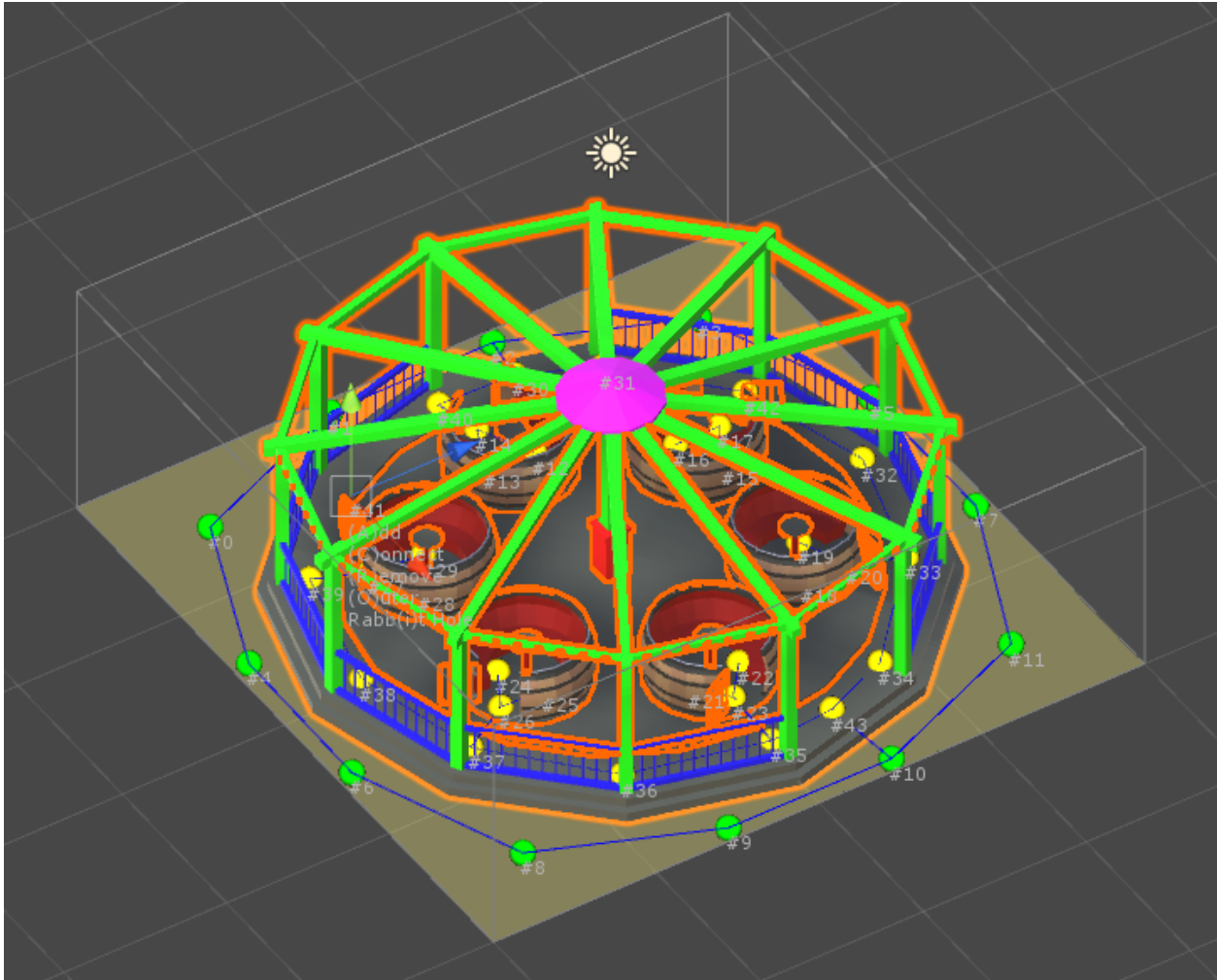


1.1.5 Waypoints

Click the “Enable editing waypoints” button to switch to the waypoints editor. A good start is to then hit the “Generate outer grid” button which creates green waypoints on the outer tiles of the ride platform. Select a waypoint, then press the “C” key to switch to the connection editing mode. Click on adjacent waypoints to create/remove connections. These connections are invisible lines that guests will walk on when navigating your ride. It should look something like this:

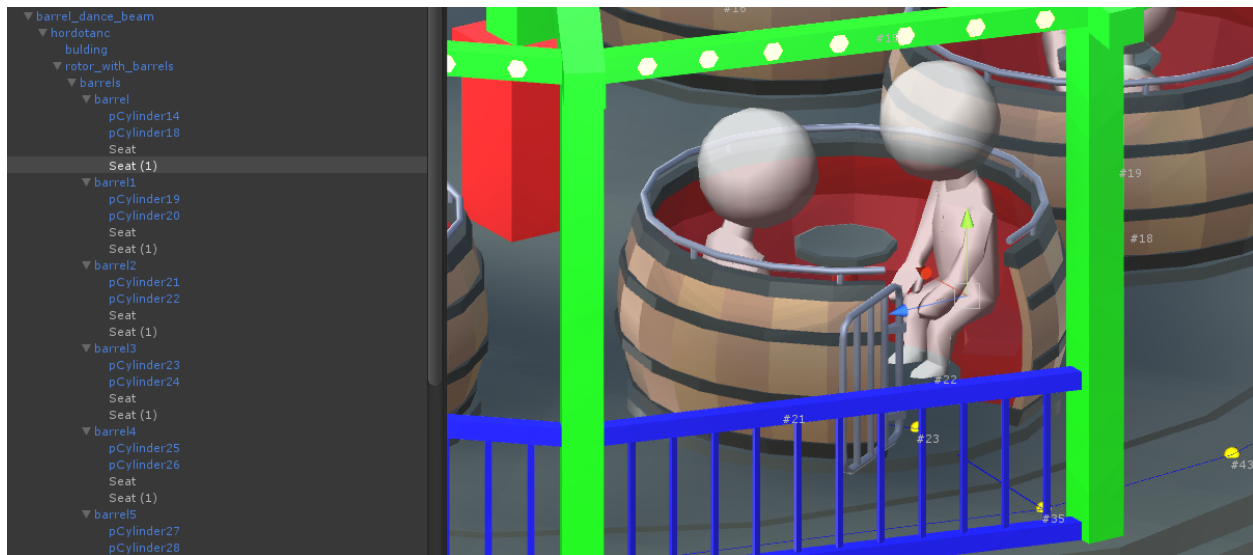


Next, hit the “A” key to create more waypoints near the seats and wherever needed, then connect them to the outer points:



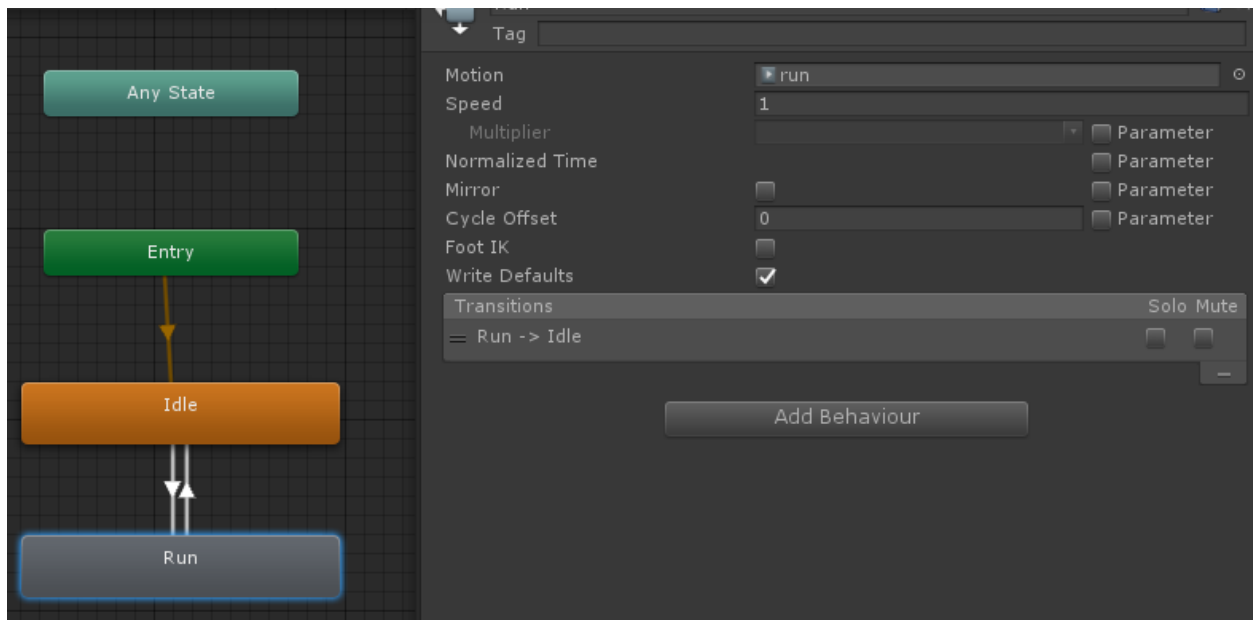
1.1.6 Seats

It's probably easiest to create seat markers in your modeling tool - simply create empty transforms named "Seat". Alternatively you can also add them in Unity.

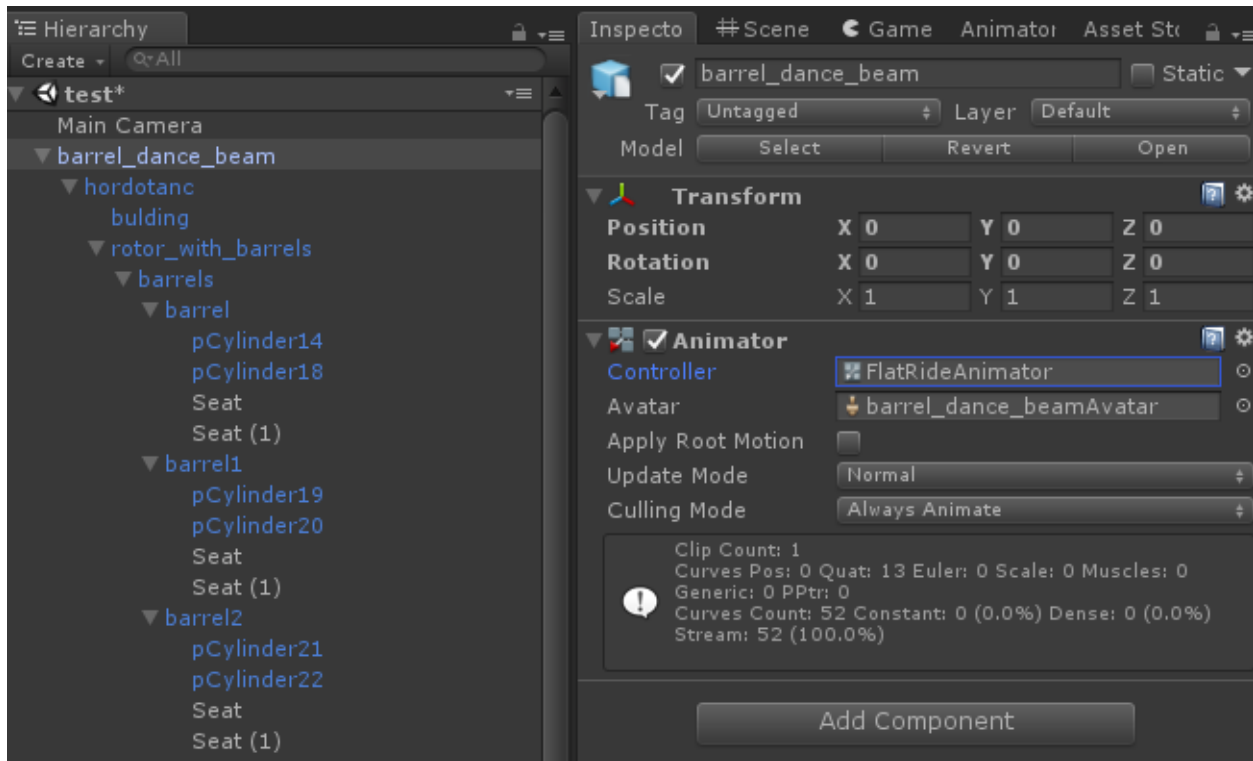


1.1.7 Animation

Double-click the animation controller located at Assets/Resources/Flat Rides/FlatRideAnimator.controller. Select the “Run” state and assign your animation clip to the “Motion” property. Most of the default rides in the game have an animation duration of around one minute which you should try to match for balance reasons.



Select your ride in the Hierarchy, then in the Inspector assign the FlatRideAnimator to the “Controller” property of the Animator component.



1.2 Train

Custom trains for most of the tracked rides in the game. The Resources/Reference Objects/Tracks folder contains reference track segments for most coaster types that you can use for scale reference, especially for determining where the car wheels should go. Note that the collision bounds, tunnel sizes and slope/curve radius of tracked rides depend on the ride type, not on the type of trains used by that ride, so generally it would be a good idea to keep your custom trains roughly within the same dimensions as the default trains used by that ride (otherwise your trains could clip through collision bounds/tunnels etc.).

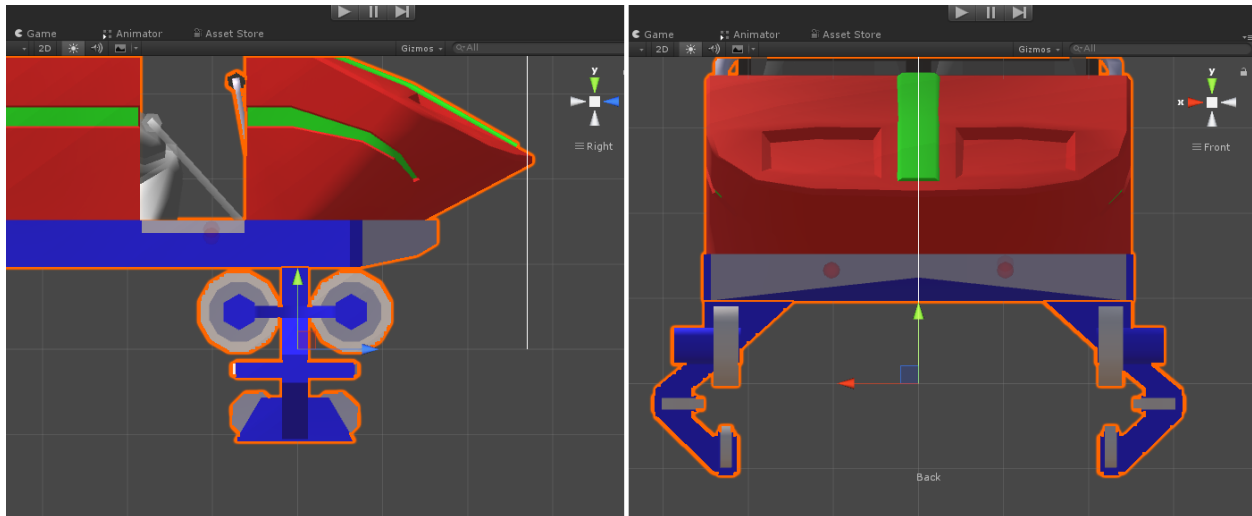
1.2.1 Color Settings

Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

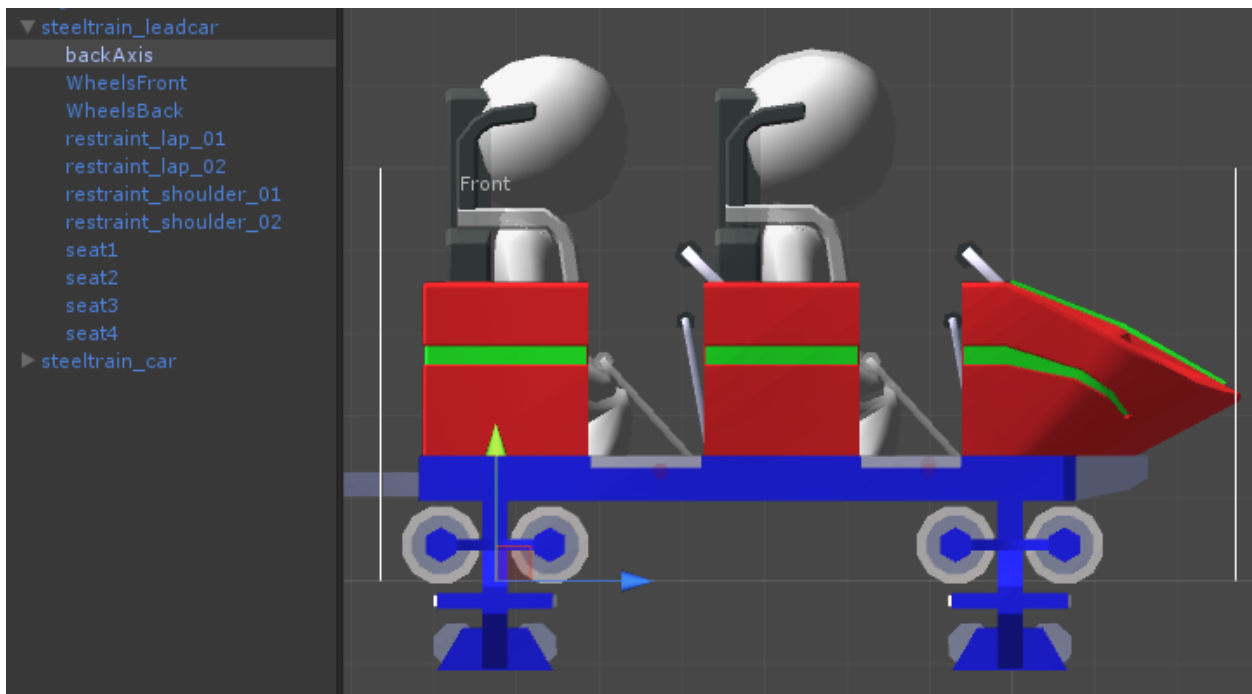
1.2.2 Car Model

You can use a different car model for the front of the train than for the remaining cars.

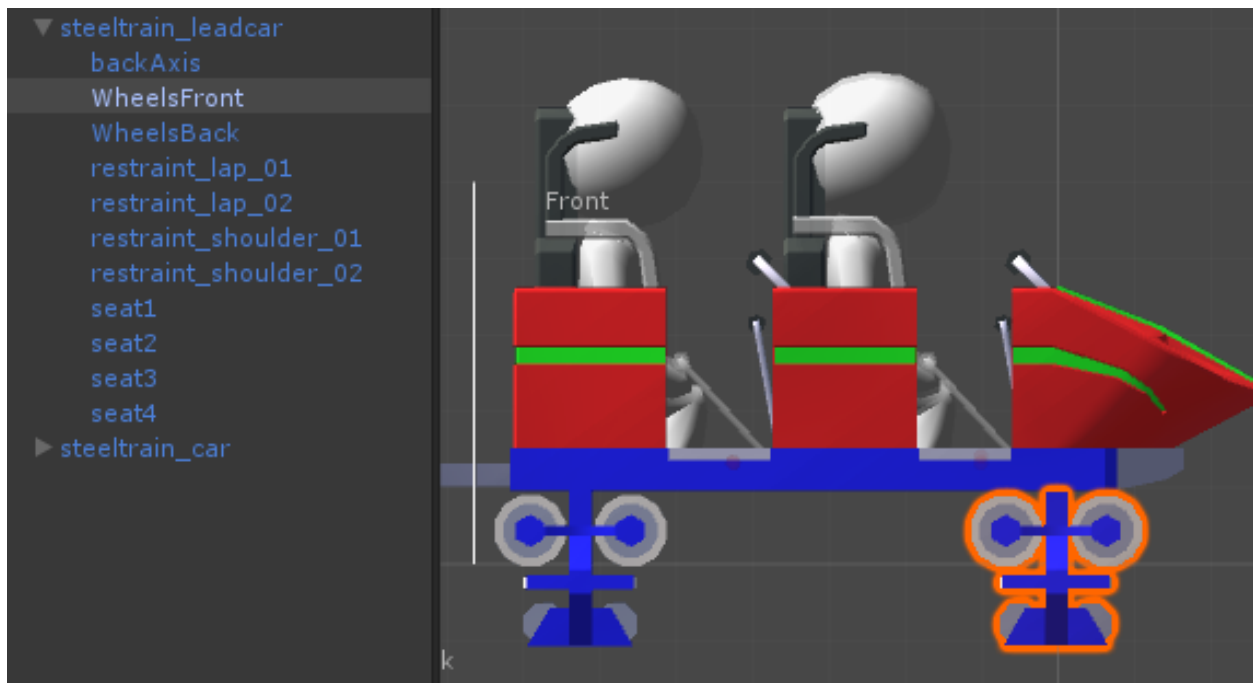
The pivot of your car should be centered between the front wheels and vertically aligned with the spot where the wheels will touch the surface of the track:



There needs to be an empty transform named “backAxis”, positioned in the same way as the cars pivot but in the location of the back wheels:

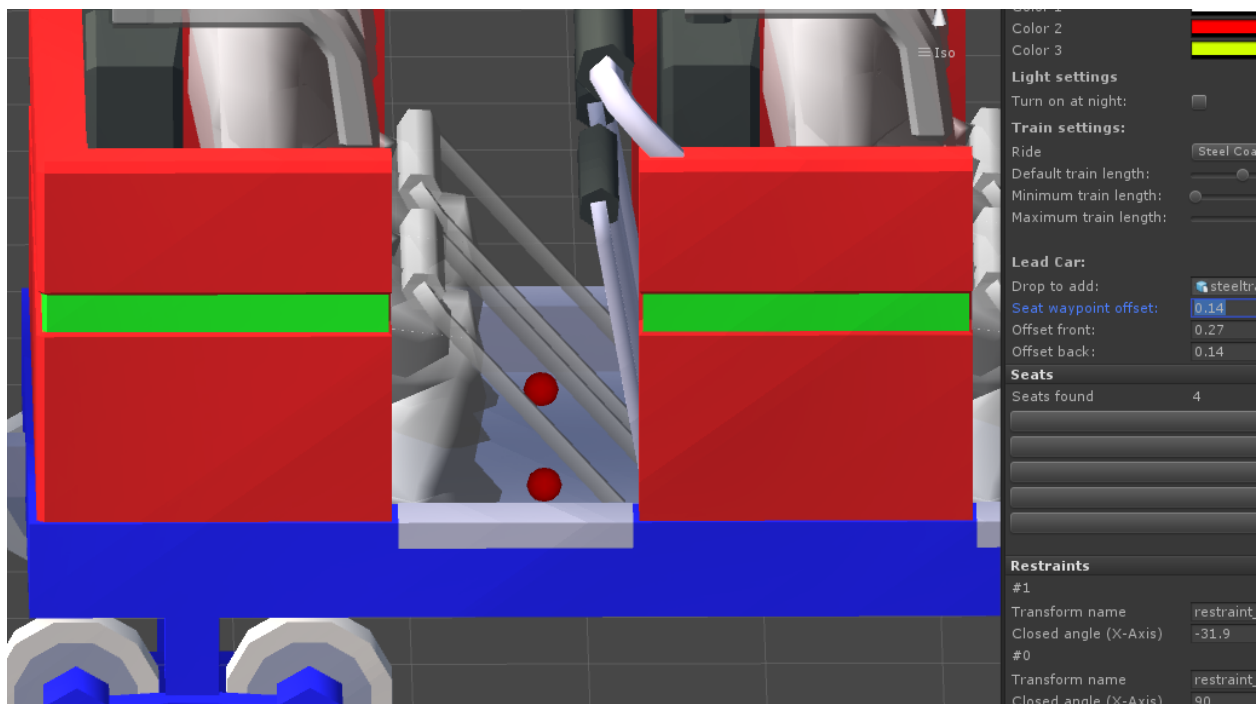


To make the wheels steer in curves split them into a separate part named “WheelsFront” or “WheelsBack” respectively. The wheels are not very clearly visible on some types of coasters or the curve radius of the track might be so wide that it’s not really noticeable whether the wheels are steering or not. In this case you shouldn’t split the wheels into a separate part.



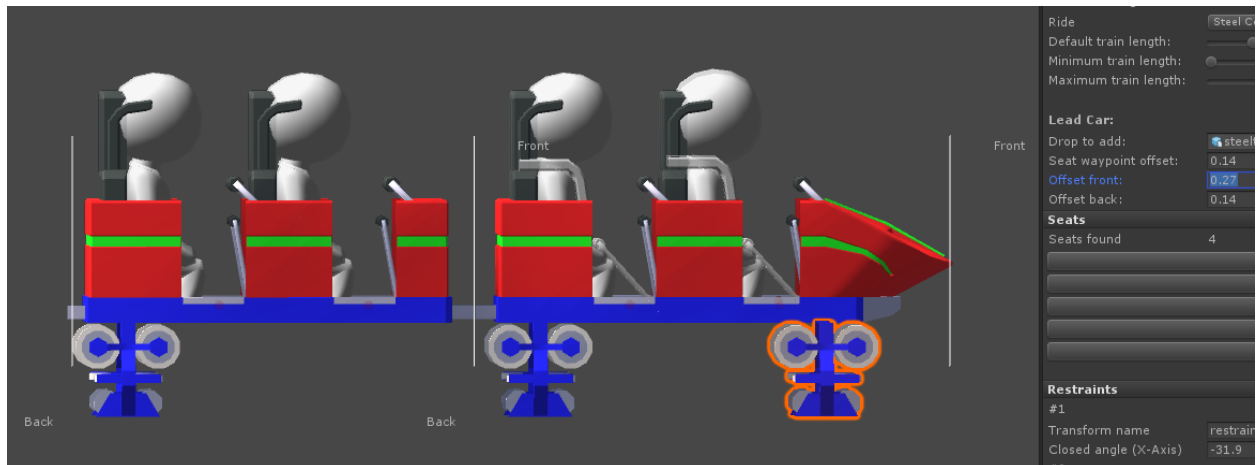
1.2.3 Car Settings

Seat waypoint offset: defines a point that is offset by the given distance from the seat locations of the train. It is shown as a red sphere in the asset preview. Guests will walk up to this spot when entering the car, so you should move this to a spot where they won't clip through the car model too badly when entering.



Offset front/Offset back: a distance relative to the pivot/backAxis marker of the car to determine where the car ends, used for collision detection with other trains and to determine where to position the next car in the train. Shown as

white lines with a “Front”/”Back” label in the preview. Move them roughly to the ends of the car model.

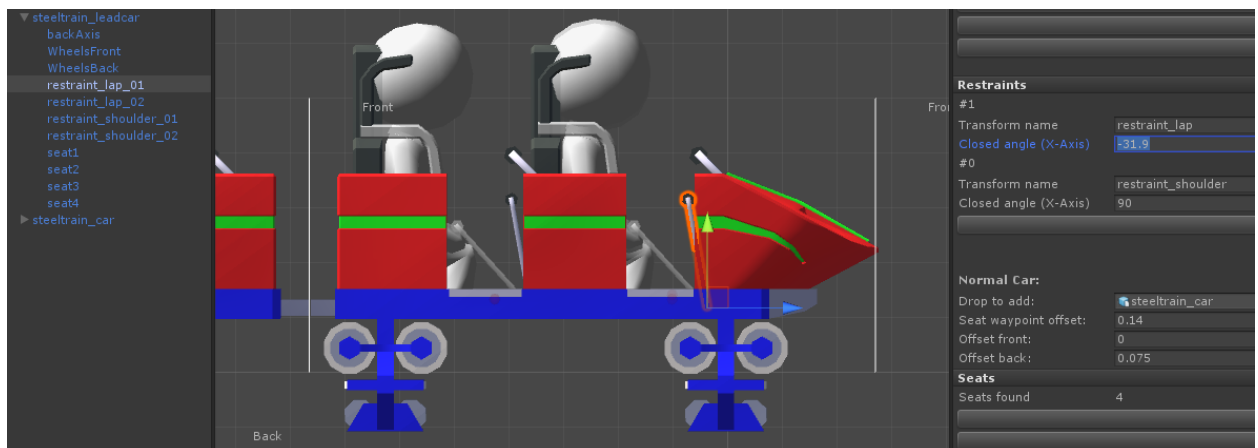


1.2.4 Seats

It’s probably easiest to create seat markers in your modeling tool - simply create empty transforms named “Seat”. Alternatively you can also add them in Unity.

1.2.5 Restraints

The restraints need to be opened by default in your car model and split into a separate part. The restraint pivots needs to be placed such that the restraint can rotate around its local X-Axis. Configure the angle at which the closed restraint should be positioned by adjusting the **Closed angle** property. You’ll see a preview of all affected restraints to assist you with determining the correct value for this property.



1.2.6 Path Finding

WALLS

2.1 Fence

Fences are like draggable walls. They have a 'flat' and optionally a 'post'. The flat will be the object itself. Therefore you should model your flat and post separately. Drag the flat object to the 'Drop to add' field to make it an asset.

2.1.1 Color settings

Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

2.1.2 Fence Settings

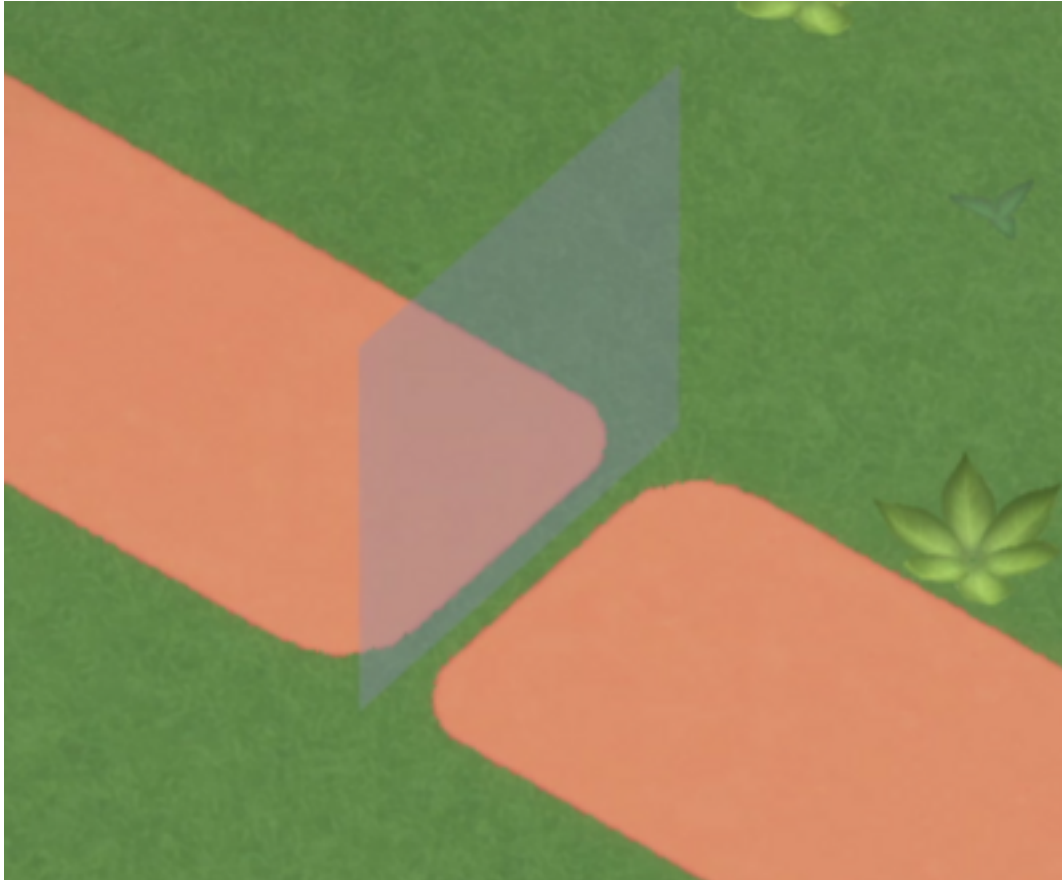
Post: if the fence should have a post you can drag the post game object from the scene to this field. This will make the post game object a child of the fence object called 'Post'.

Has mid posts: whether the fence has mid posts.

image of mid posts difference

2.2 Wall

Walls are an extension of deco objects with the only difference that they can block paths.



Because walls are an extension of deco they share the same [deco settings](deco).

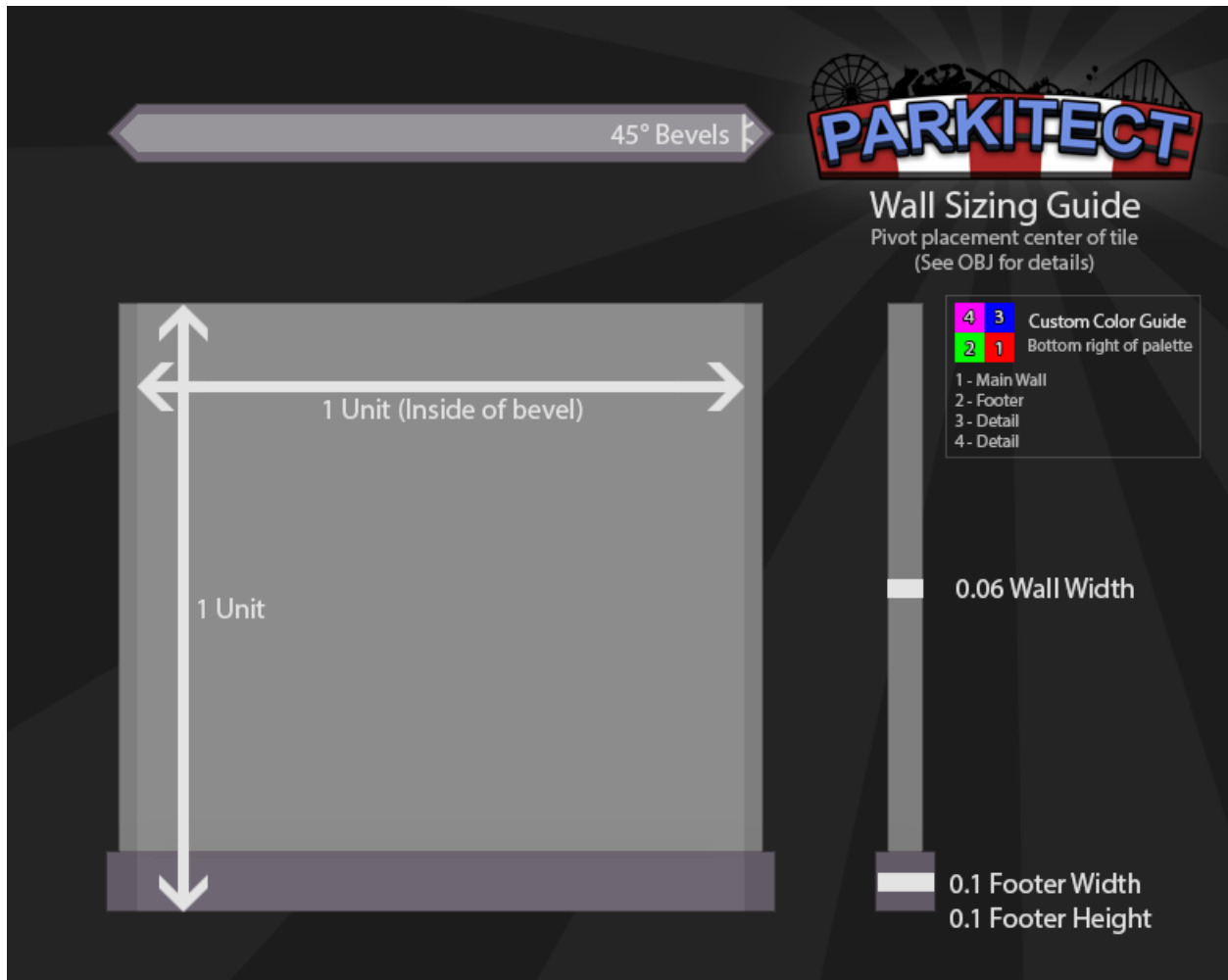
2.2.1 Wall settings

The block settings will toggle which sides of the object should block paths. If that side of the object is placed between two paths, they will not connect anymore. A white cross will appear on the sides of a wall that are blocked.

Height: the height of the wall. Usually it should either be 1, 0.5 or 0.25 to match the games default walls.

2.2.2 Wall Size

Garret made a fancy guide on how to make walls fit with the rest of Parkitect.



PATH ATTACHMENTS

3.1 Bench

Benches can be build on the side of paths. They have one or more seats where guests can sit on.

3.1.1 Color settings

Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

3.1.2 Bench settings

Has back rest: whether guests sitting on this object can put their arm onto the back of the bench. Use the example bench included in the Asset Editor package as guide for the height of the back rest.

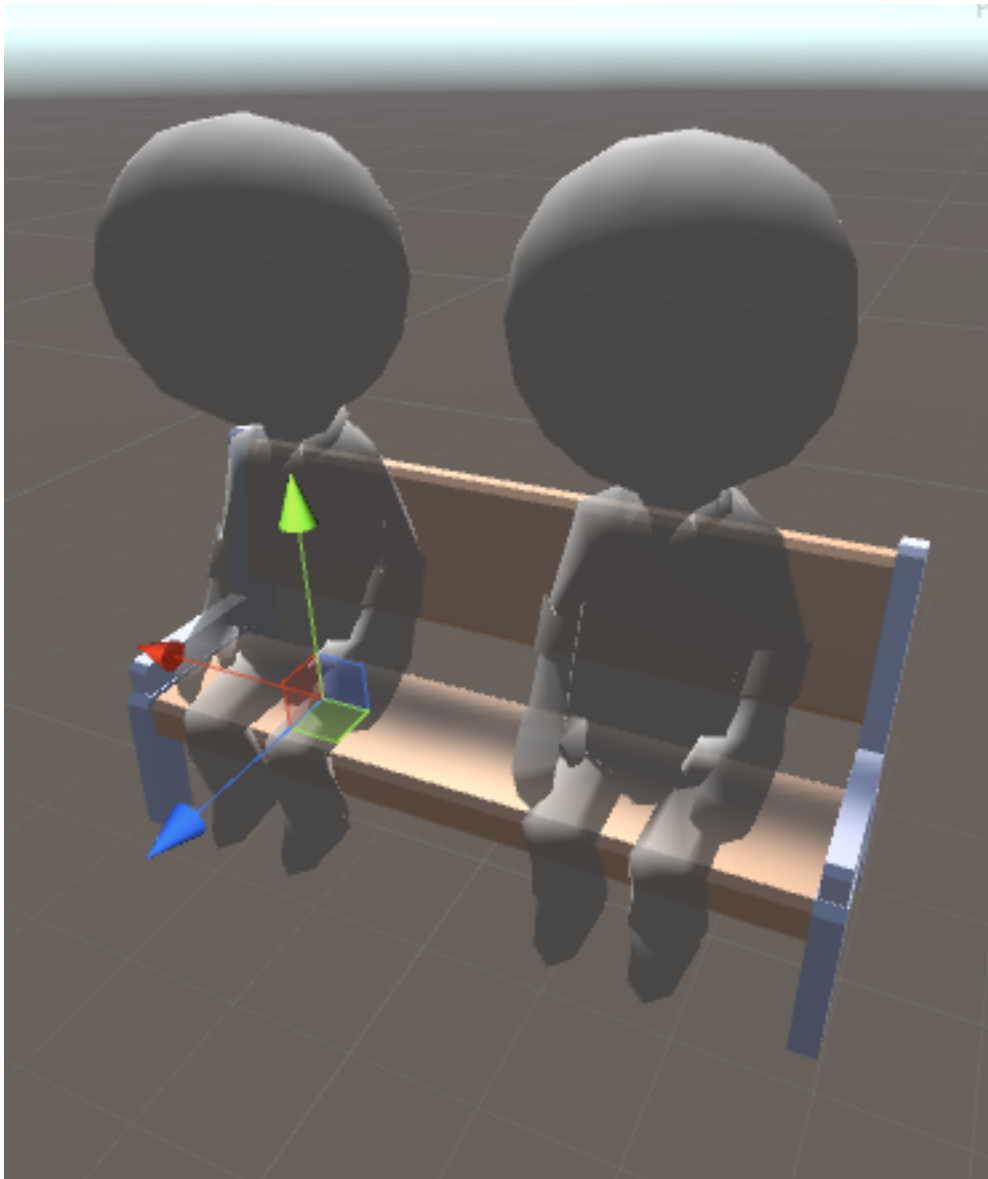


3.1.3 Adding Seat

```
▼ bench_modguide  
  Seat1  
  Seat2
```

Adding seats works by adding child game objects to the bench. Any game object in the hierarchy whose name starts with “Seat” will be marked as seat. Guests will sit directly on top of this object.

When you’ve added a seat correctly there will be a rendered guest helper that will show where guests will sit.



3.2 Lamp

Lamps currently do nothing special, they're simply decoration that can be placed alongside paths. Check the [\[Lights\]\(Lights\)](#) page to see how to add lights to your lamp.

3.2.1 Color Settings

Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

3.2.2 Settings

Lamps have no unique settings. They only support [\[custom colors\]\(custom-colors\)](#).

3.3 Sign

Signs can be placed over or on the side of paths. They can also optionally show texts.

3.3.1 Color Settings

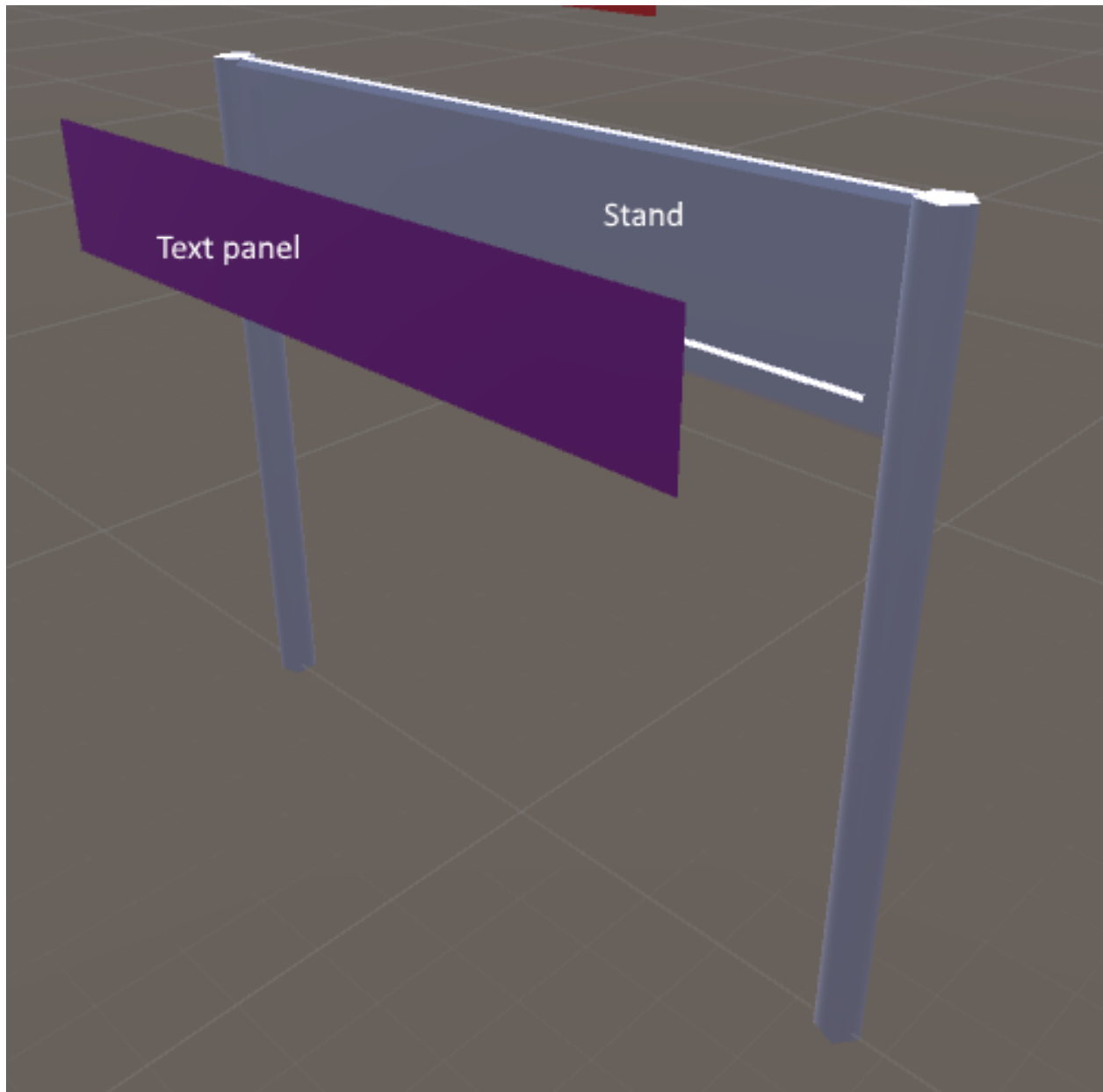
Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

3.3.2 Sign Settings

Text object: the child game object of the sign that should show the text. Parkitect will replace the material on this object with the text material.

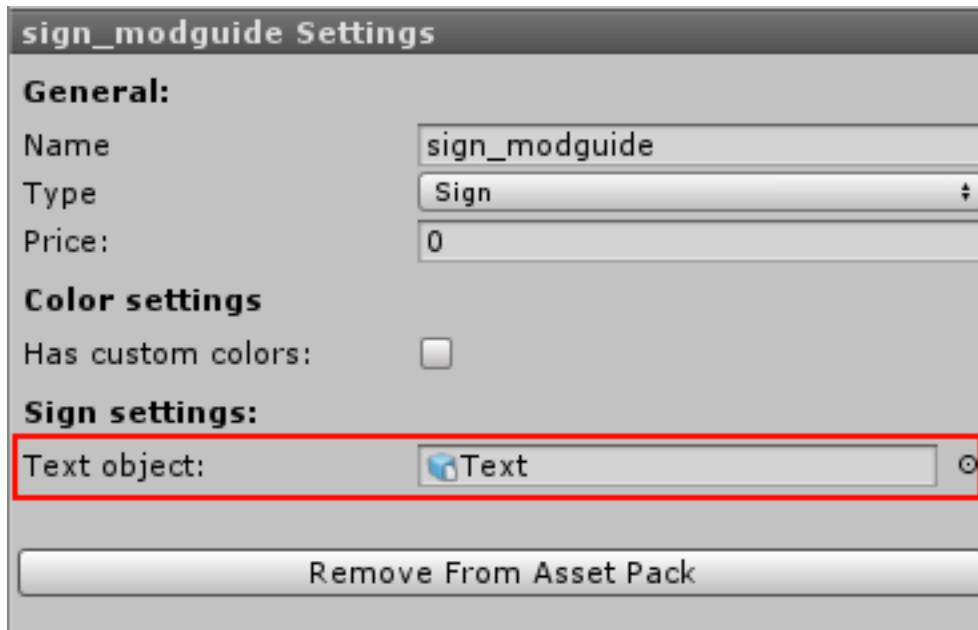
3.3.3 Setup Sign Object

Your Sign object should consist of two models. The stand and a child text panel model, see the image below.



The stand can be modeled and mapped normally following the [custom colors](custom-colors) guide. The text panel requires 1 material named *SignText*. If you name the material correctly, the game will replace the material with the real text shader that shows the scrolling text.

Make sure you've dragged the child text panel game object to the *Text object* setting of the Sign settings.



3.4 Trashbin

Trashbins can be placed on the side of paths.

3.4.1 Color Settings

Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

3.4.2 Settings

Trashbins have no unique settings. They only support [\[custom colors\]\(custom-colors\)](#).

3.5 TV

TV's are path attachments that are on the side of paths/queues.

3.5.1 Color Settings

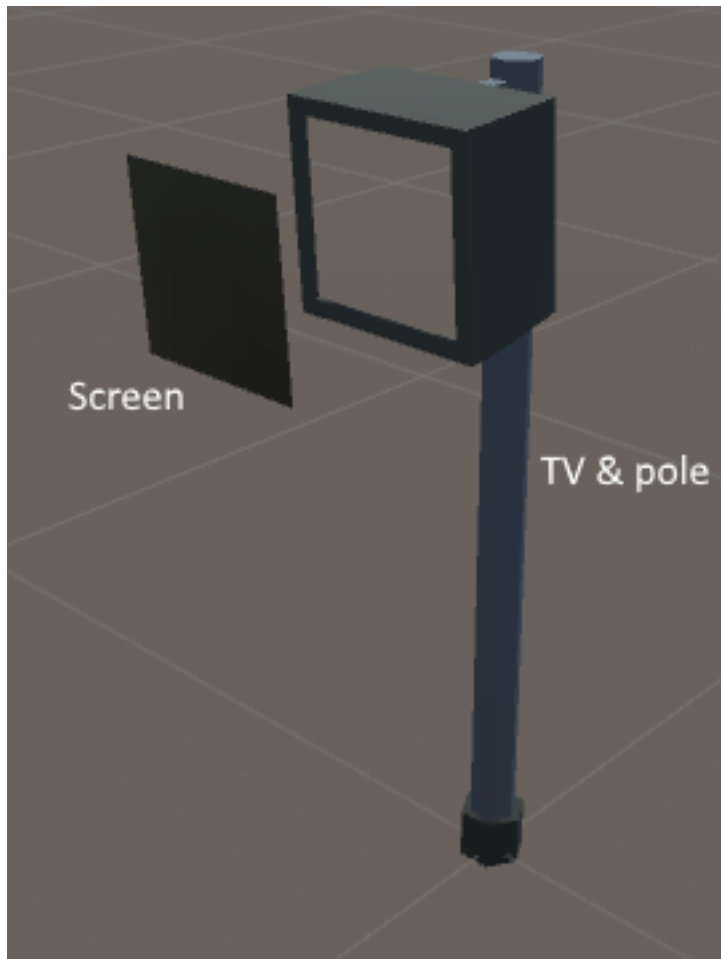
Every object support custom colors. Check [\[this\]\(Custom-colors\)](#) page for more info on custom color settings.

3.5.2 TV Settings

Screen object: the child game object of the TV that should show the image. Parkitect will replace the material on this object with the screen material.

3.5.3 Setup screen object

Your TV object should consist of two models. The pole with a TV encasing and a child glass panel model, see the image below.



The pole with TV case can be modeled and mapped normally following the [custom colors](custom-colors) guide. The screen requires 1 material named *TVImage*. If you name the material correctly, the game will replace the material with the real TV shader that shows images of your park.

Make sure you've dragged the child screen game object to the *Screen object* setting of the TV settings.

tv_modguide Settings

General:

Name: tv_modguide


Type: Tv

Price: 0

Color settings

Has custom colors: ☐

Tv settings:

Screen object:  Screen

Remove From Asset Pack

PARKITECT ASSET EDITOR

4.1 Light

Lights look nice, especially during night. You can put lights onto any object, but usually you'll use them on [Lamps](Lamp) or [Deco](deco). There are two types of light effects you can use.

By default lights are always on, even during the day. For lights that should be on during the night and off during the day there are the following settings:

Turn on at night: whether this light only turns on during the night (on) or is always on (off).

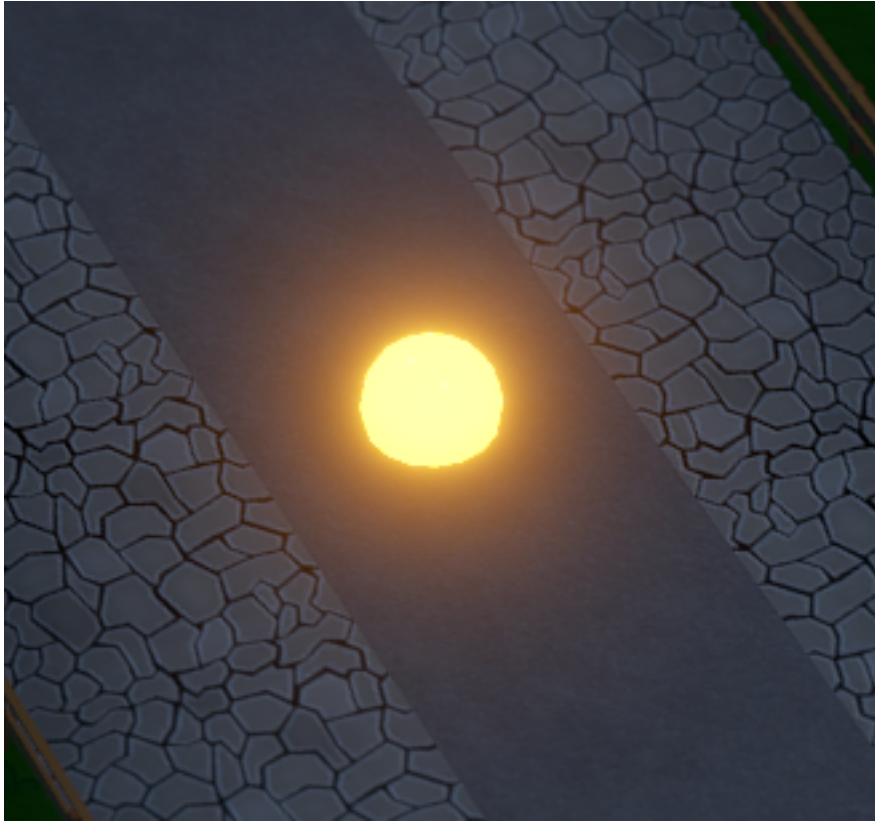
If your object uses [custom colors](custom-colors) these additional settings are available for actual lights (doesn't matter for glow effects since they use the custom colors from the material):

Use custom colors: whether the emitted light should be tinted using one of the custom colors (on) or not (off).

Custom color slot: pick which of the custom color slots should be used for tinting the emitted light

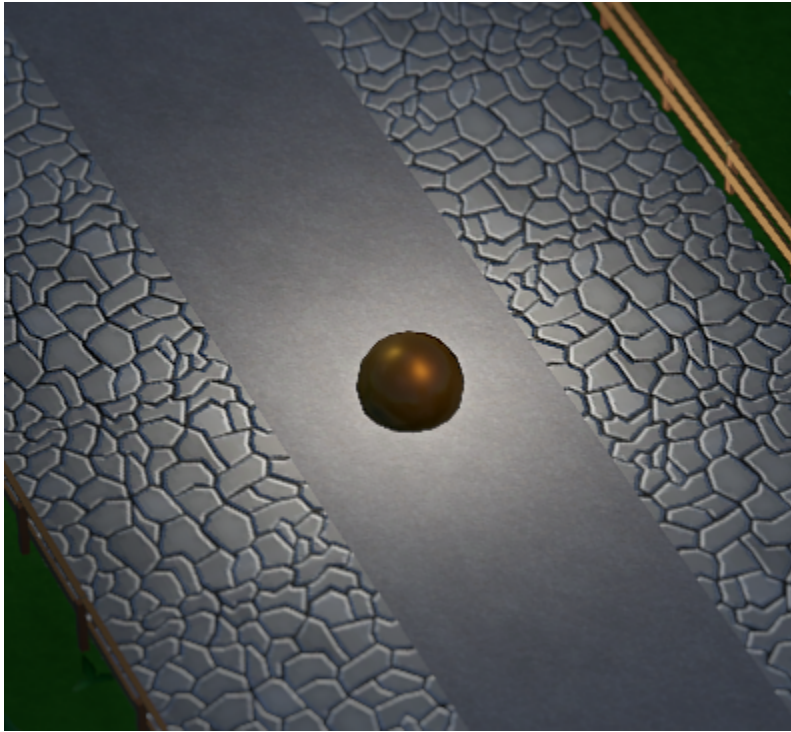
4.1.1 Glow Effect

You can give parts of your object a glow:



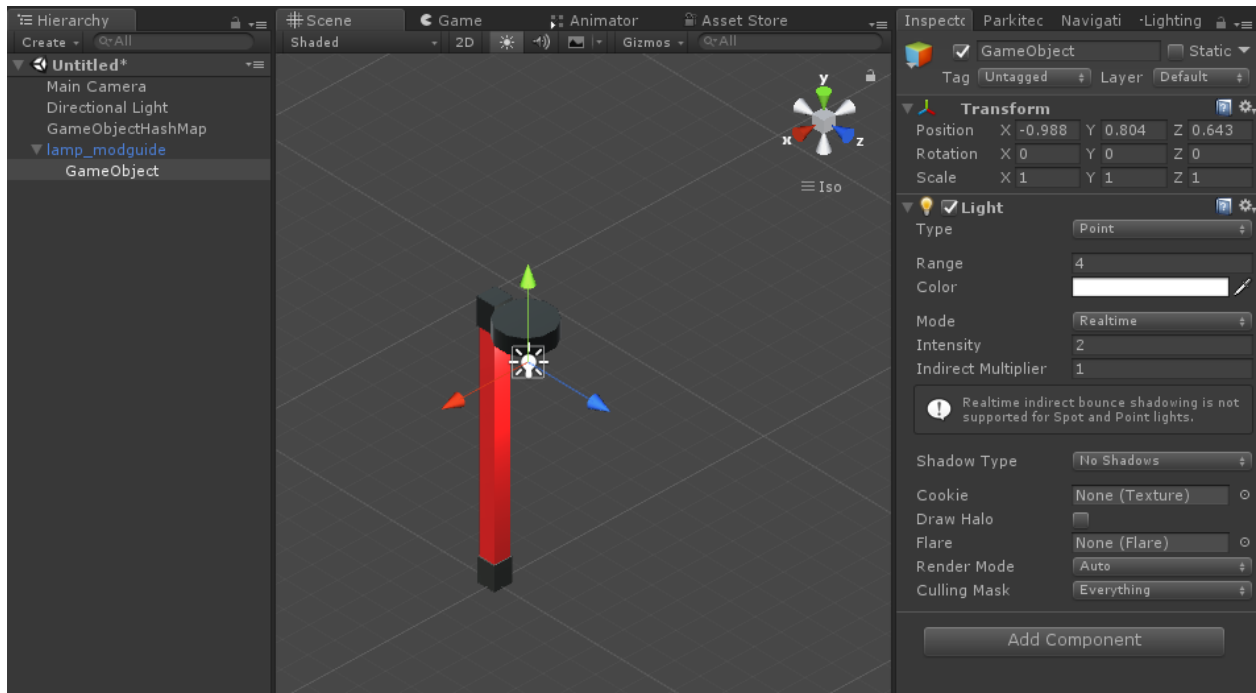
This is achieved by using the [CustomColorsDiffuseIllum/CustomColorsSpecularIllum material](Materials) on the parts of your object that should glow. This effect has **no performance costs**, so you can use it as much as you want.

4.1.2 Actual Light Effect



This effect **has performance costs**, so consider if your object actually needs this before using it, especially for small objects that players tend to spam all over the park. For example, for small light bulbs (such as the ones we got on the rides or string lights) it is enough to give them a glow effect, they don't actually have to light up the surroundings. Ideally also don't use more than one light source per object. If your object has multiple visible light sources you can probably approximate them with a single light component.

To use an actual light, add an empty `GameObject` to your custom object and position it where the light should originate from, then add a `Light` component to it.



The light type can either be “Point” or “Spot”. Leave **shadows disabled** as that can have massive performance costs. Some good default values are:

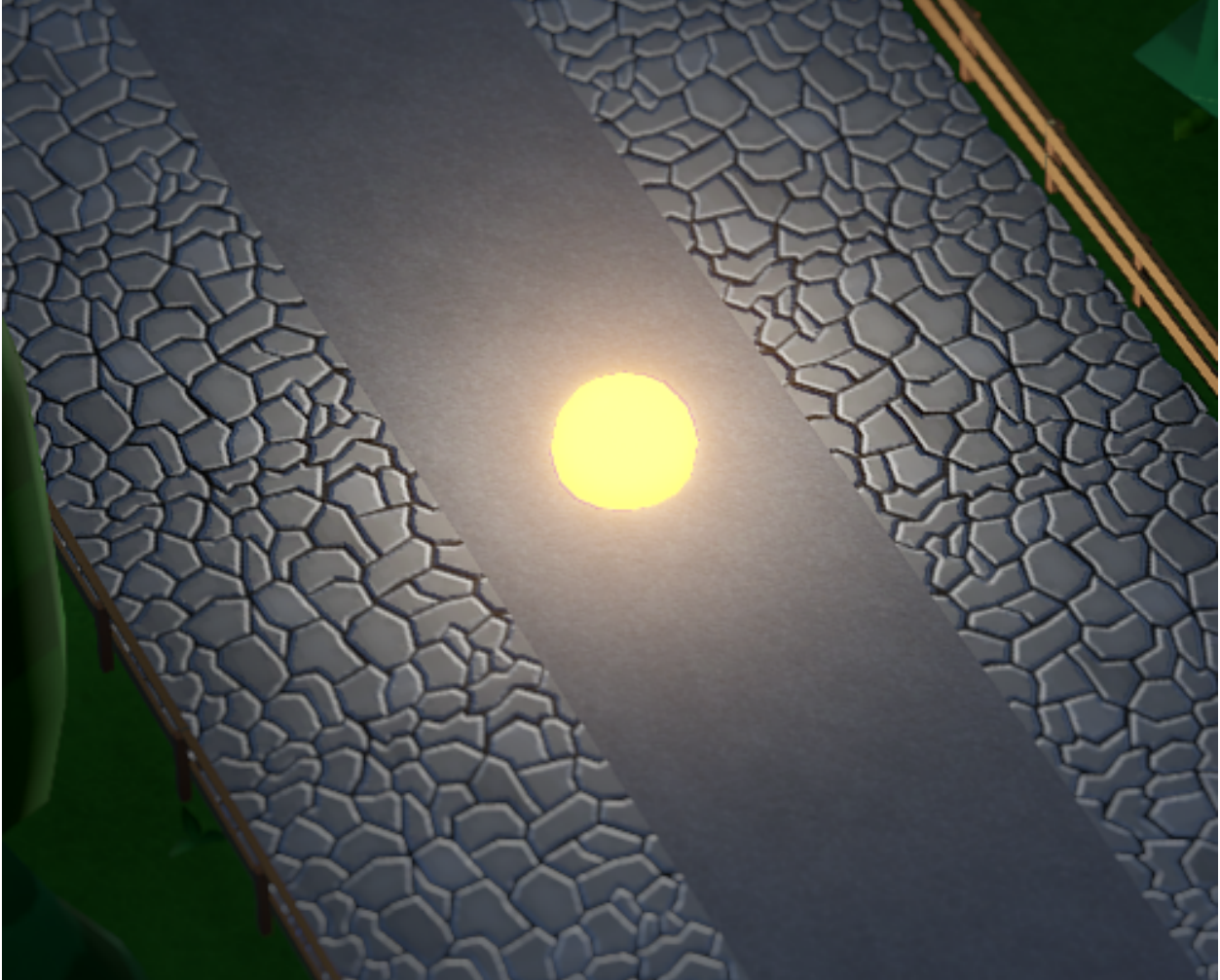
For point lights

- Range: 4
- Intensity: 2

For spot lights

- Range: 1.5
- Spot angle: 100
- Intensity: 2.5

You can also combine both effects of course:

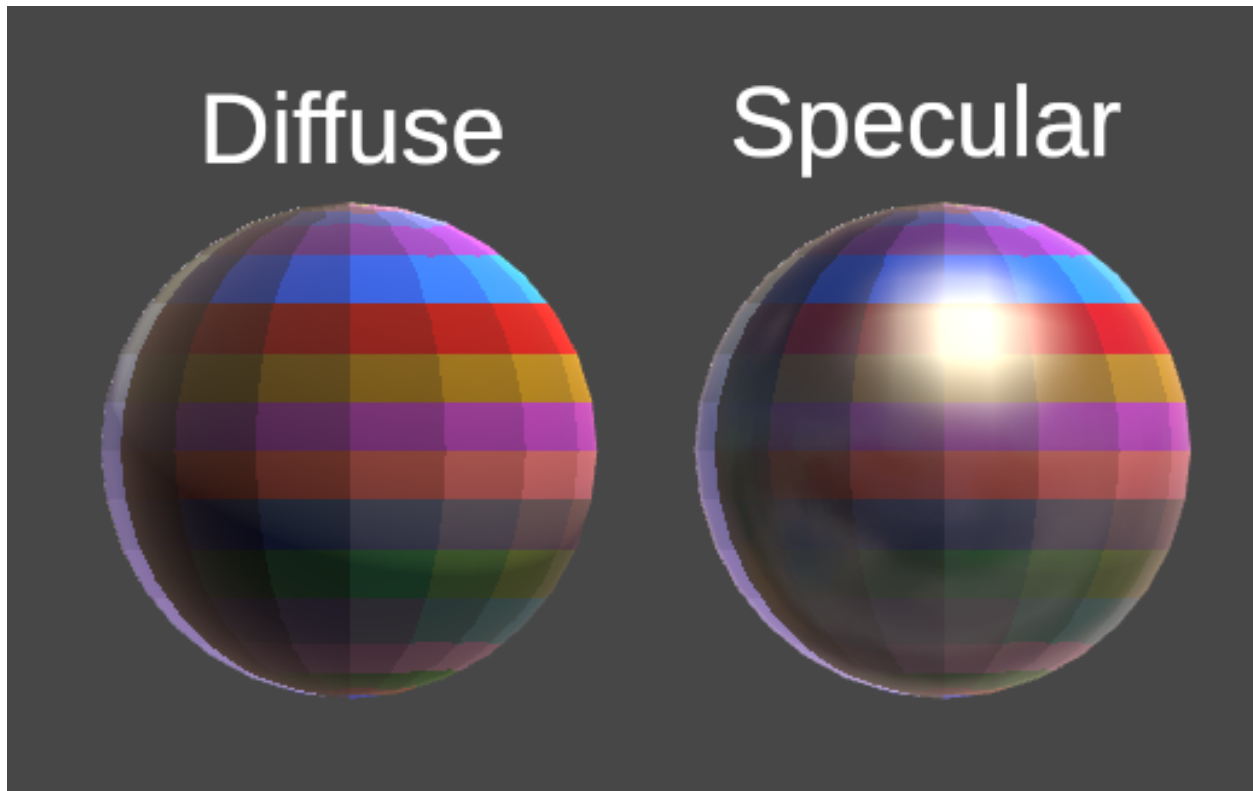


4.2 Materials

We've shipped some materials with the asset editor that you should use for your custom objects. These materials are special treated by the game. The materials are:

- Diffuse
- Specular
- CustomColorsDiffuse
- CustomColorsSpecular
- CustomColorsDiffuseTransparent
- CustomColorsSpecularTransparent
- CustomColorsDiffuseIllum
- CustomColorsSpecularIllum

As the name gives away there are two base materials: the Specular materials are used for metallic looking materials and the Diffuse ones for anything else. Then there are custom color variants that support having player-defined colors and the additional variants of those that support transparency and illuminated ones that [glow at night](Lights).



These materials all have the same texture, the 256 palette, and are used by most objects in Parkitect. Because of that the GPU can use one texture for almost all objects. **We also can guarantee that if you use these materials your assets will keep rendering properly in future game updates. If you roll your own materials they may (and probably will) stop working in the future and you have to update your asset pack. That's why we strongly encourage you to use our materials.**

4.2.1 Using the materials

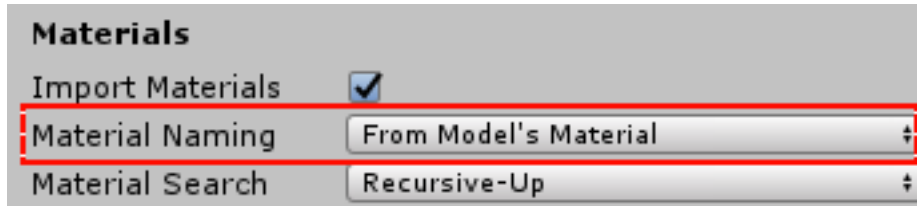
As you can see the 256 palette is a very simple texture with 256 colors and the largest part of Parkitect is build on these colors. To use them on your models you need to import the 256 palette texture in your 3d modeling program. The texture is shipped with the asset editor and is located at *Assets/Resources/Textures/256palette.tga*. The next step is to map the texture on your model, also called UV mapping.

> Tutorials on how to map uv's with your favorite 3d program can be easily found on the internet.

Next, map every face of the model to a color on the texture. The outlines of the face don't have to follow the outlines of a color. As long as every vertex is inside the square it will work correctly.

Make sure the material name of your model matches with one of our models. Otherwise it will not be replaced properly in-game, leading to broken models in the future. Use *Diffuse* or *Specular* for non recolorable objects and *CustomColorsDiffuse* or *CustomColorsSpecular* for recolorable objects.

Export the model as fbx, obj or another format that Unity can read and copy it to *Assets/Resources* in your asset project. Select the model and make sure the that "Material naming" has the value of "From model's material" in the inspector.

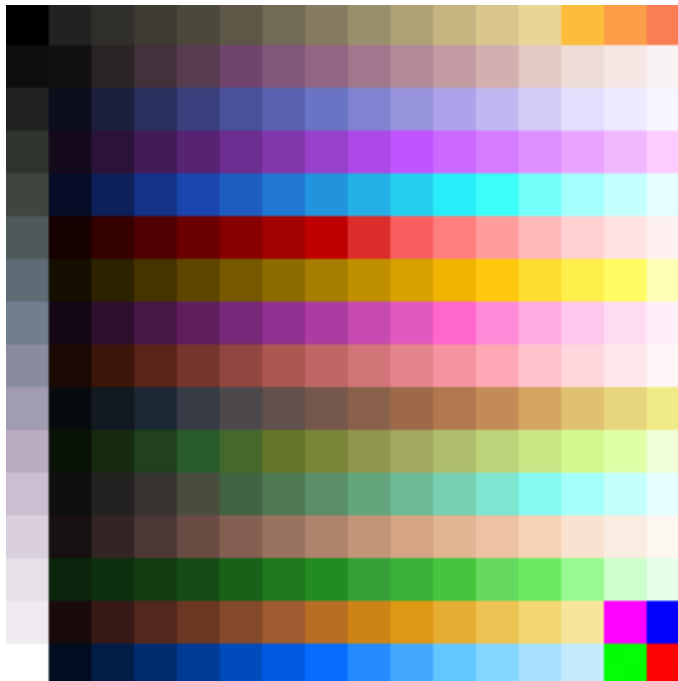


> For some model formats you need to check the scale factor of the model in the inspector. For example, fbx needs a factor of 100 to match 1 unit with 1 tile in Parkitect.

4.2.2 Custom coloring

In the bottom right of the 256 palette you see 4 special colors. Those are the colors that can be recolored in-game. The pixels are in the following order:

Color	Target
Indigo	Color 4
Blue	Color 3
Green	Color 2
Red	Color 1

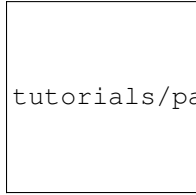


Your object can support a maximum of 4 custom colors. Map the faces of the objects that needs to be recolored to these 4 special colors. Don't skip a color with mapping. For example never use color 1, 3 and 4 but instead use color 1, 2 and 3.

When you've mapped the colors you also need to [enable recoloring](Custom-colors) on your object and set some default colors.

4.3 Custom Color

Some objects in Parkitect supports custom colors. You probably have used it before.



tutorials/parkitect_asset_editor/./img/custom_color_picker.png

The asset editor supports custom colors if you setup your model correctly.

> Check [this](#) page on how to setup your model to support custom colors.

When your model is setup correctly you can enable custom colors in the asset editor. 1. Check *Has custom colors* to enable support. 1. Select the amount of custom colors your model has in the *Color count* field.

After that you can pick the colors for your asset. Players can change the colors later but this will be the default value that will be selected in-game.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`