Creating and using Persistent Levels

1. Introduction

The Ninja Slayer Unreal project employs a technique where we a singular stage consists a main level (Persistent) and four sub levels (Environment , Light , Level Design and System).

The sub levels are streamed into the main level, which is how a stage is created. Start by reading [REDACTED] to familiarize yourself with the concept.

Example image from the Cloudbuster Level Window

2. Setting up a persistent Level

Ensure that you follow [REDACTED] to set up the framework for your new level.

3. Persistent Level Content

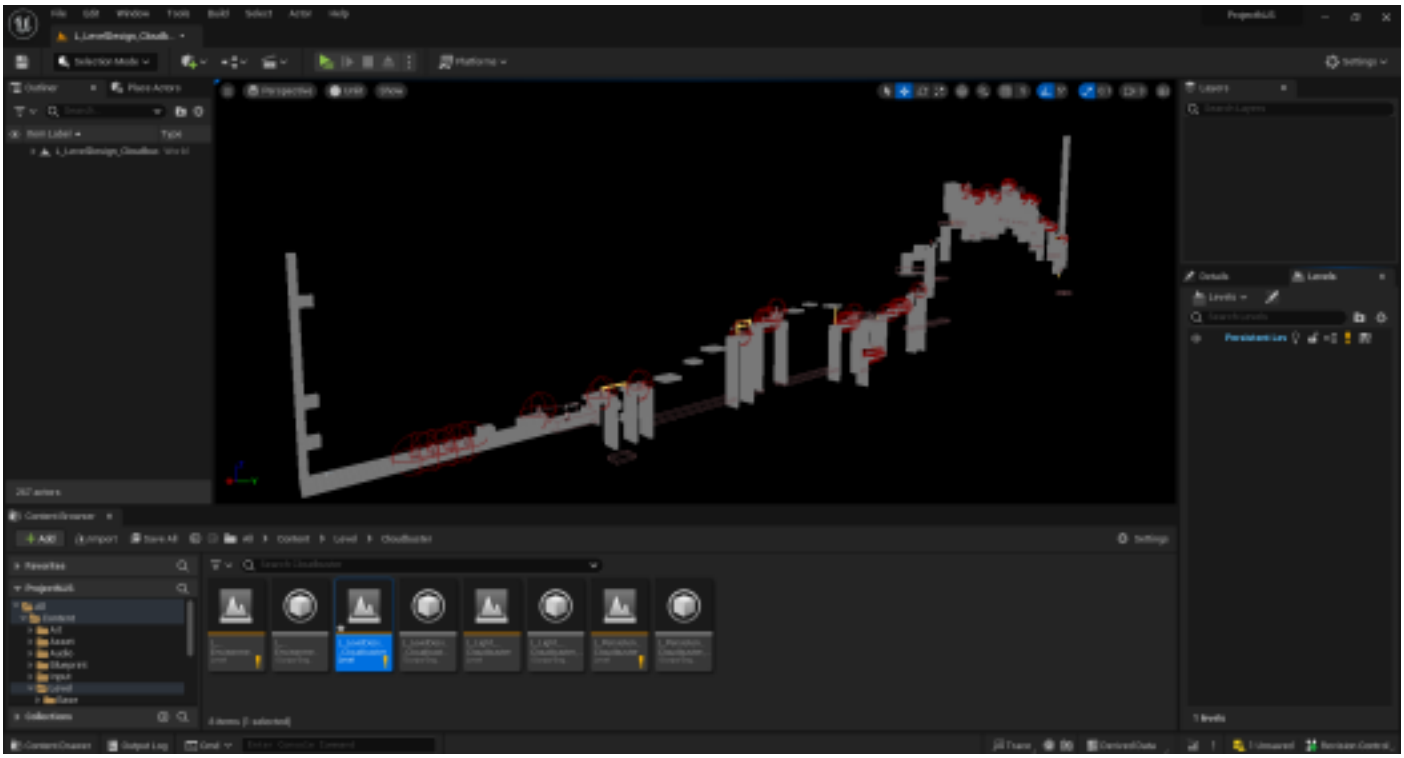
Although the [REDACTED] to persistent levels gave a brief summary, here is a breakdown of the content for each level:

- L_Persistent

  Absolutely no assets should be placed directly in L_Persistent , you should just ensure you have set up level streaming properly here.
- L_LevelDeisgn

  This level is used for placing level geometry , gimmicks and enemies .

  For detailed instructions please read [REDACTED] thoroughly.



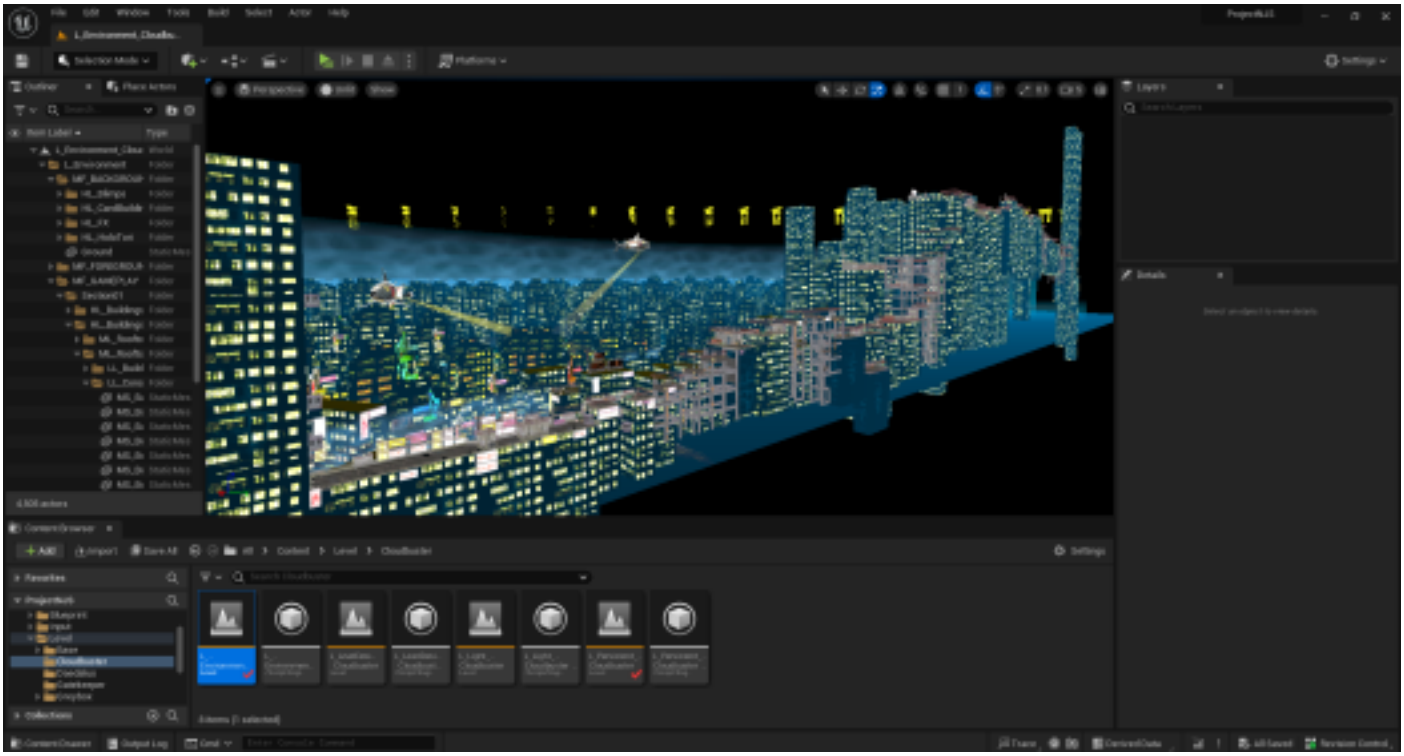
Example image of L_LevelDesign_Cloudbuster

Once you have approval, you can move onto the stage's L_Environment level.

- L_Environment

  This level is used for placing art assets and lights .

  For detailed instructions please read [REDACTED] thoroughly.



Example image of L_LEnvironment_Cloudbuster

- L_Light

  This level is mainly used for placing lights.

  For detailed instructions please read [REDACTED] thoroughly.

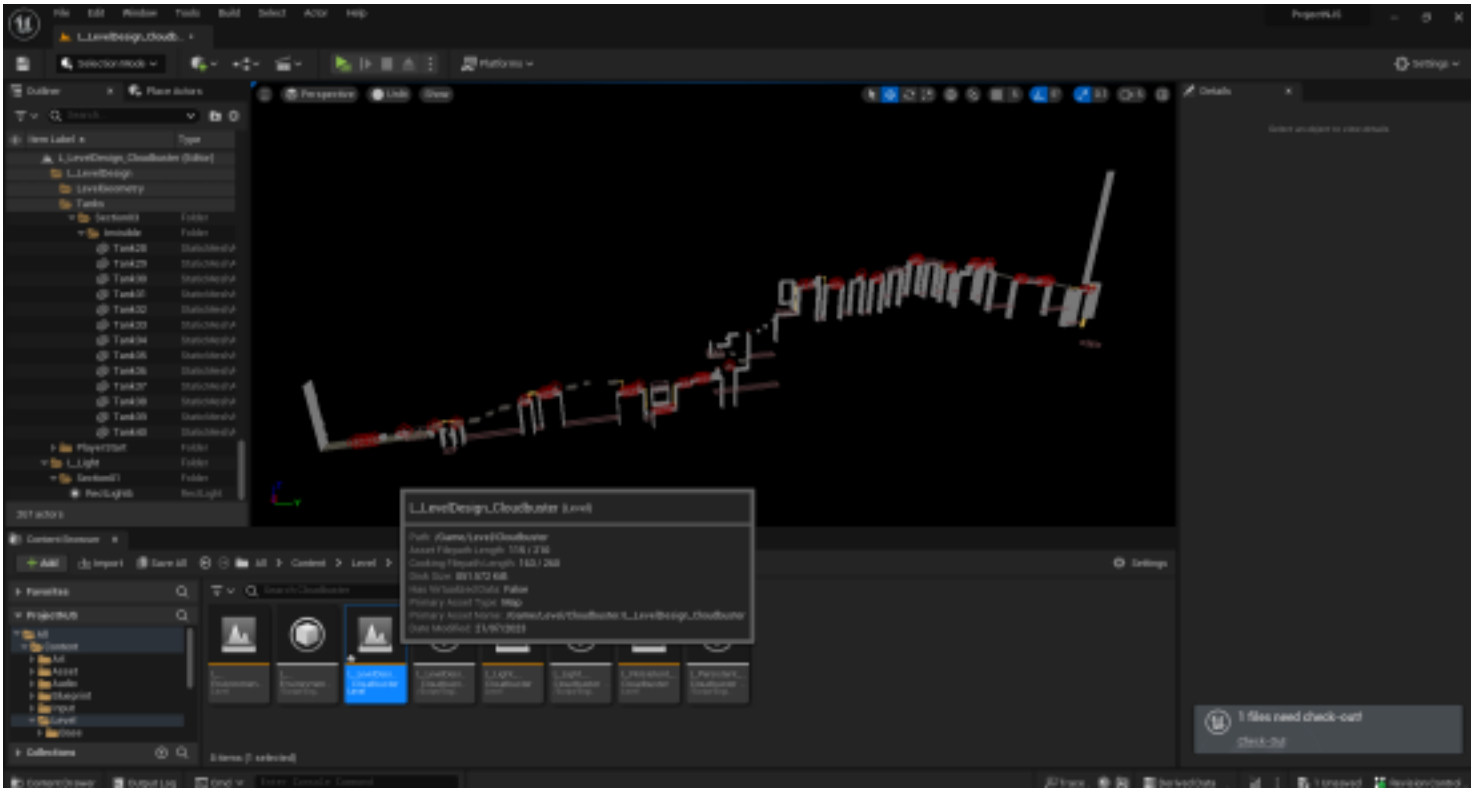
- L_SystemDebug

  Absolutely no assets should be placed directly in L_SystemDebug , you should just ensure you have it included in your stage.

4. Working between persistent levels

You have two options when it comes to editing levels.

Option 1 - Working Directly in in your sub level



Example image of L_LevelDesign_Cloudbuster being selected in the UE5 Viewport

For this option, just double click the sublevel you wish to edit. For this example, I have chosen `L_LevelDeisgn`.

This will directly load into `L_LevelDeisgn` and from here you can start placing assets.

This has the merit that you can easily ensure you are working out of `L_LevelDeisgn` and gives you an easy overview of the stage without any additional clutter.

This has the demerit that your level will be missing art assets, lighting or the ability to play the level as these features come from `L_Light`, `L_Environment` or `L_SystemDebug`.

> This means you can not play or test your level from `L_LevelDeisgn`, instead you need to load the persistent level to playtest.
>
> Be vigilant, if you test your level and find things you want to change, you would need to load back into `L_LevelDeisgn` first!

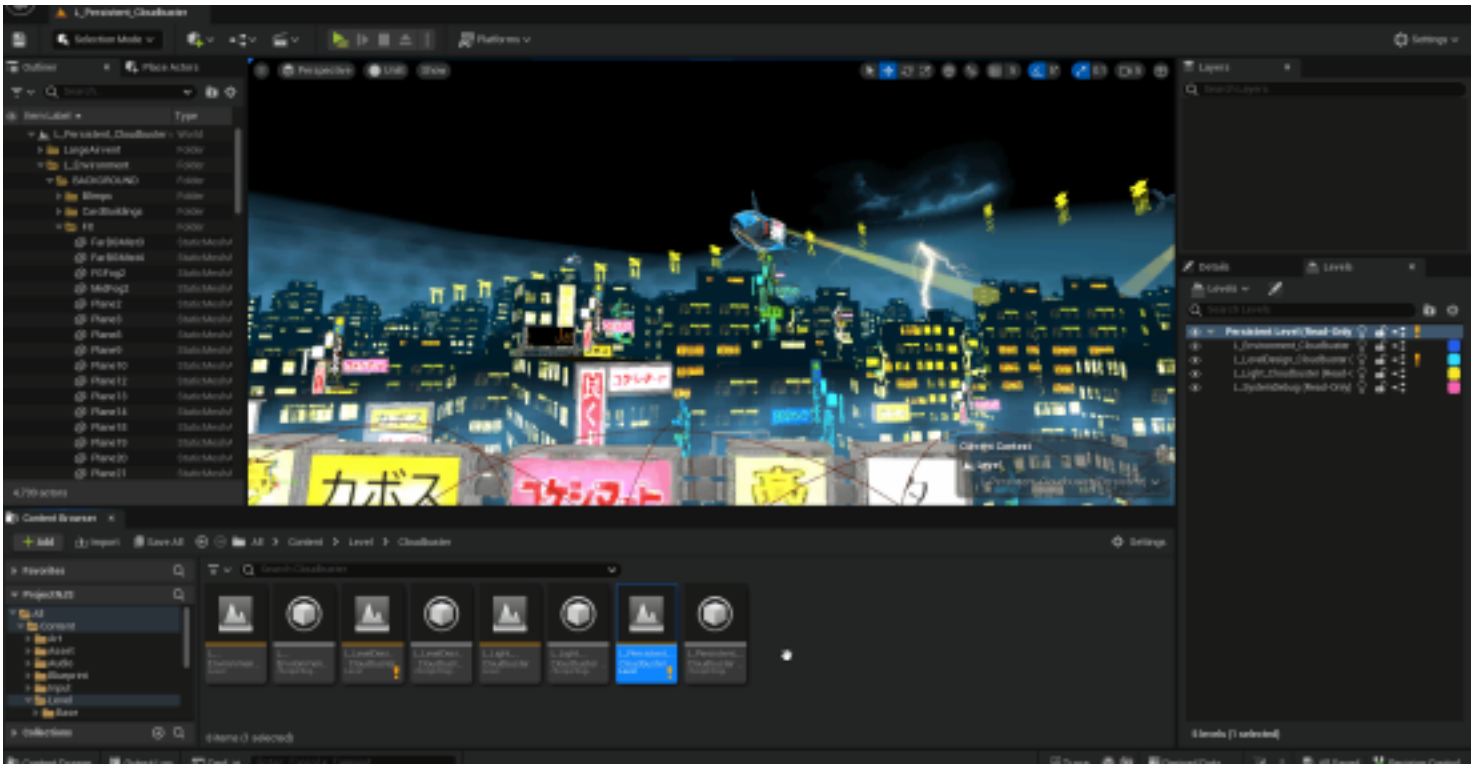
Option 2 - Working out of `L_Persistent`

For this option, start by opening the persistent level for your stage, and make sure the level window is

open too.

From the level window, double click on the level you want to work on which in this case is `L_LevelDeisgn`

> Make sure you are double clicking `L_LevelDeisgn` in the Level Window and NOT the content browser



Example of selecting L_LevelDesign from the Level Window while having L_Persistent still open

This will leave the `L_Persistent` stage open in the scene view but make it so that all changes are saved in `L_LevelDeisgn` .

The merit of this is that you can edit `L_LevelDeisgn` while seeing the lighting and being able to hop in and out of gameplay for testing. You can easily move assets between the sub levels and see what level each asset is located in via the outliner.

The demerit is that you can accidentally make changes to the `L_Persistent` stage instead of the intended sub level by accident if you do not remember to select the sub level you want to edit in the level window.

> Always make sure to work in the right level and don't edit `L_Persistent`
> You can see what level you are editing at all times and also see what level your selected asset is in via

a small window in the bottom right hand corner of `L_Persistent` .

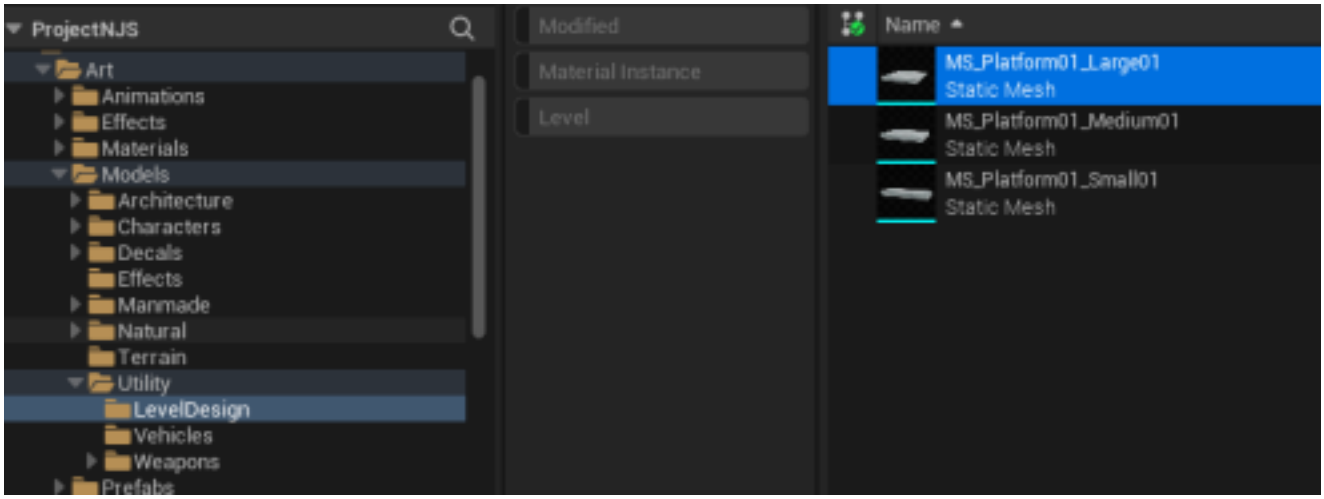
Example of the window at the bottom right hand corner, use it to check what level you are editing. In this case the loaded level is L_LevelDesign_Cloudbuster while the selected object is in L_Environment_Cloudbuster

latforms

The level geometry of a stage should be kept as consistent as possible between levels, to ensure a consistent style and feel between the levels and to maximize effectiveness with the player character.

When creating platforms you should always try to use the template platform models!

They can be found at:
/Art/Models/Utility/LevelDesign/

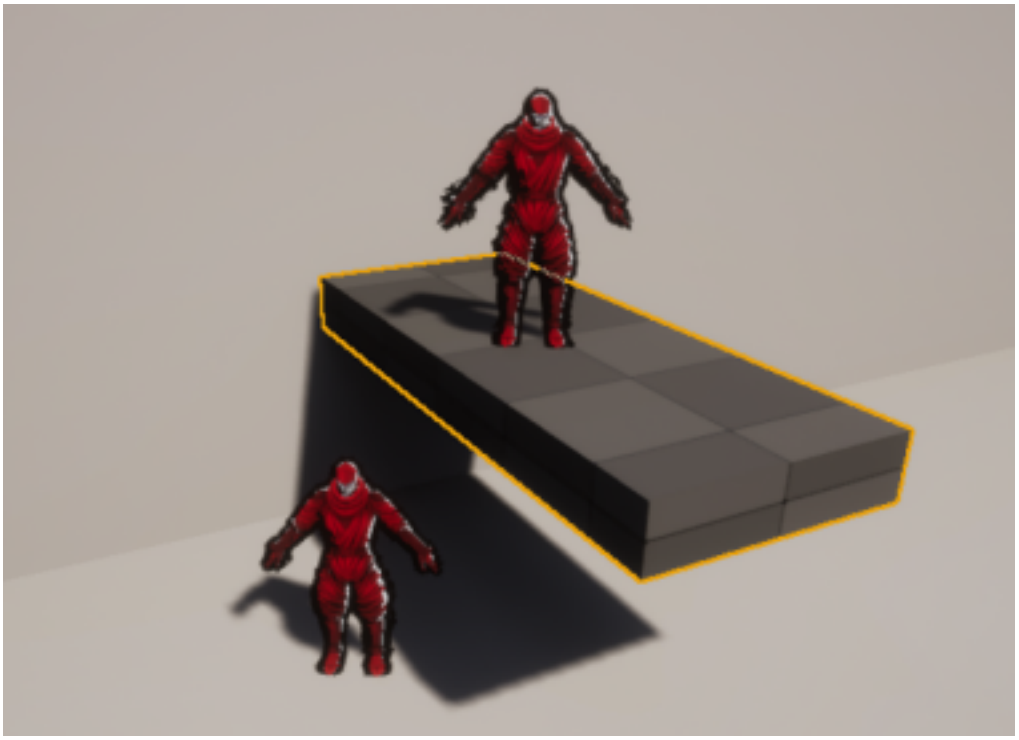


They come in **3 different sizes:**

Small

MS_Platform01_Small01

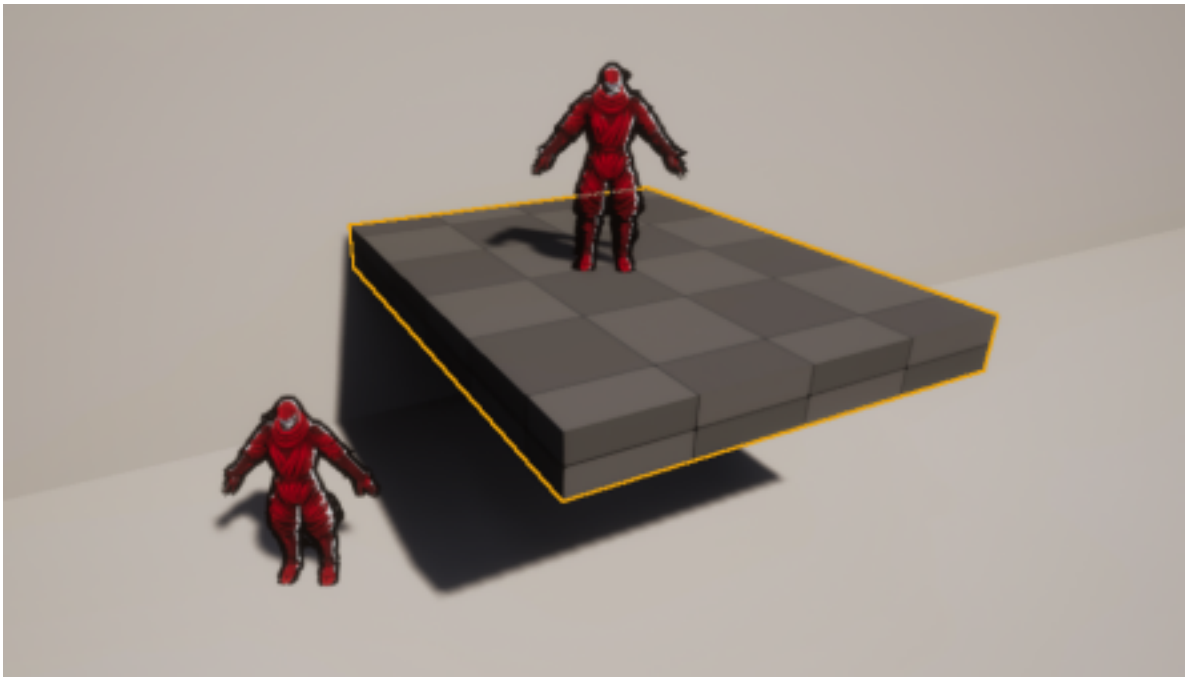
2 meters long



Medium

MS_Platform01_Medium01

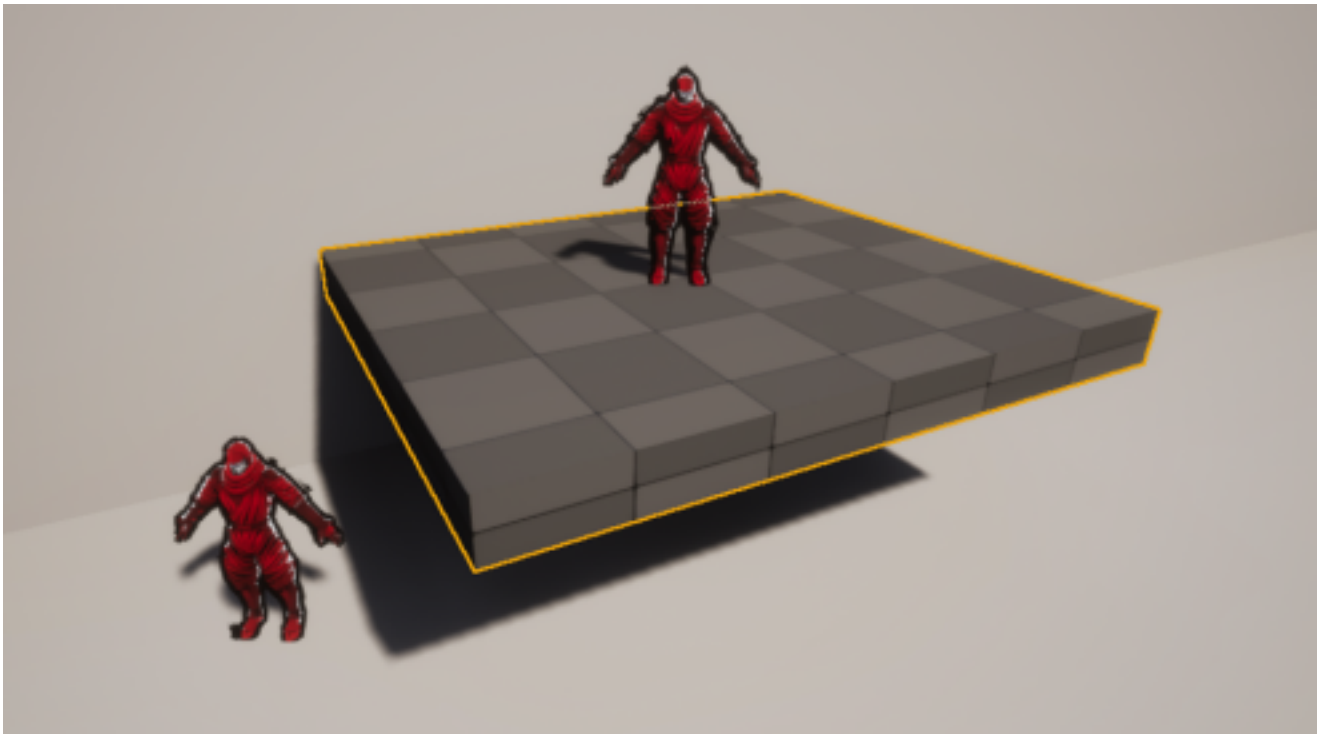
4 meters long



Large

MS_Platform01_Large01

6 meters long



Platform Scaling

In general, we want to **minimize scaling platforms.**

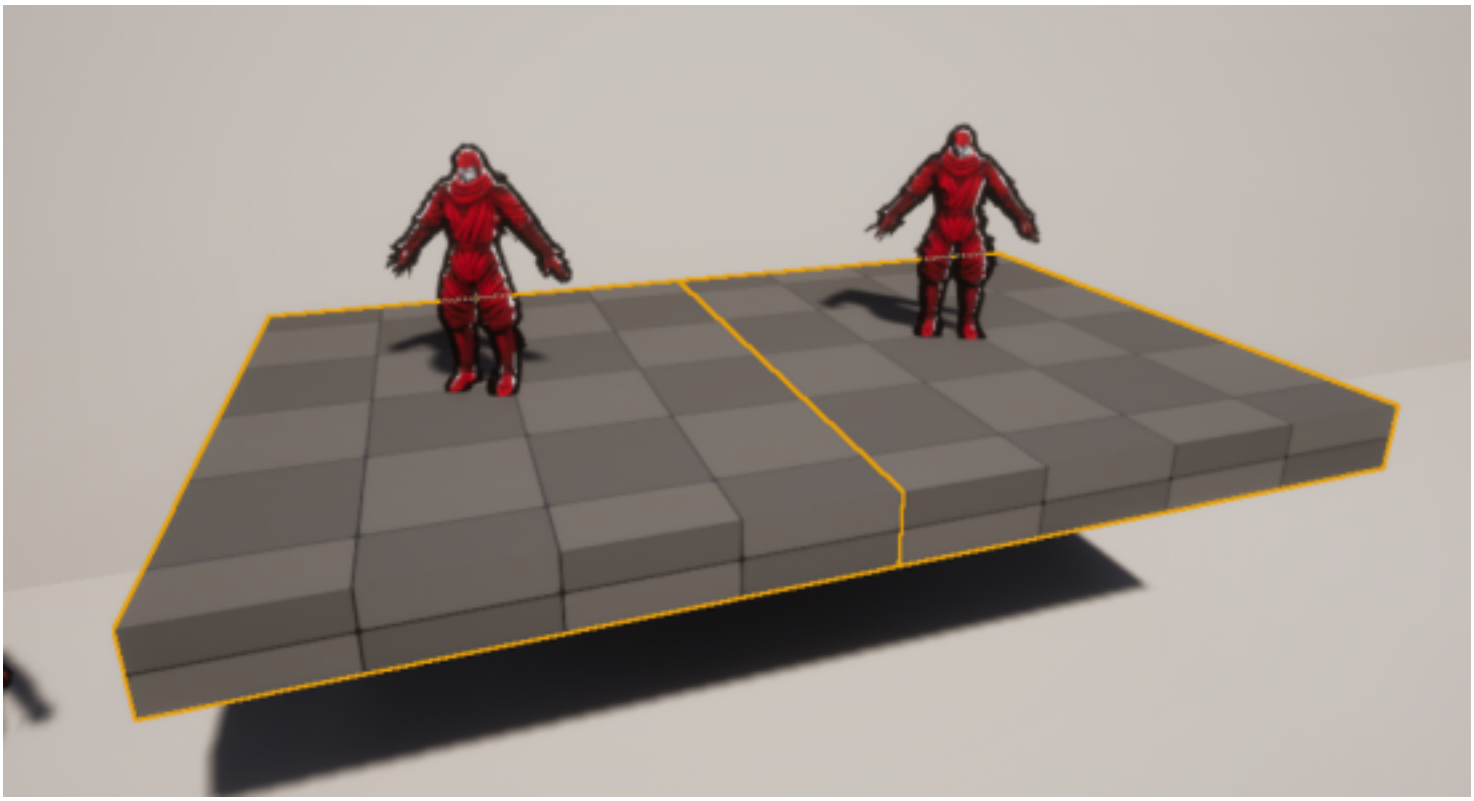
This means you should try to use the platforms at their original scale.

For example:

If you have placed a medium platform and feel that it is too small, try to use the large platform instead

of scaling the medium platform.

Also, feel free to place multiple platforms side by side to create larger platforms.



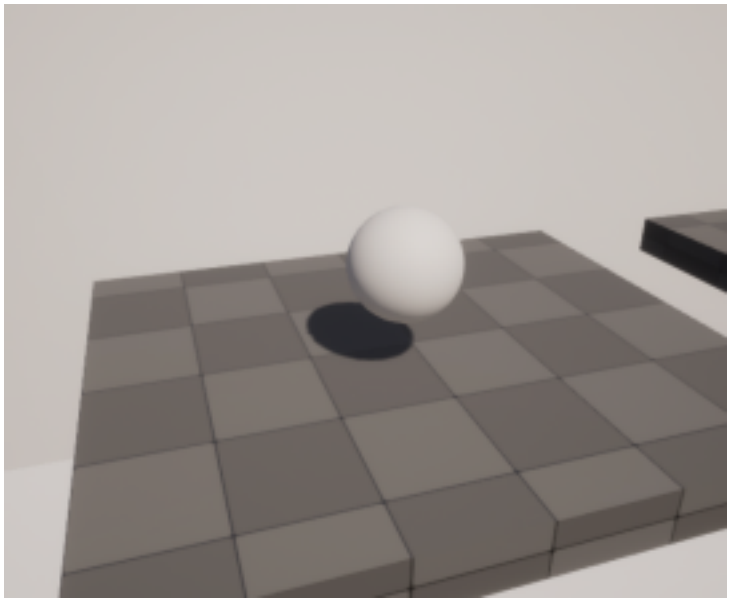
If required, you are able to scale platforms but it is ***highly advised not to scale them more than 150% and only scale them on the Y axis***

150% would be 1.5 on the Y axis

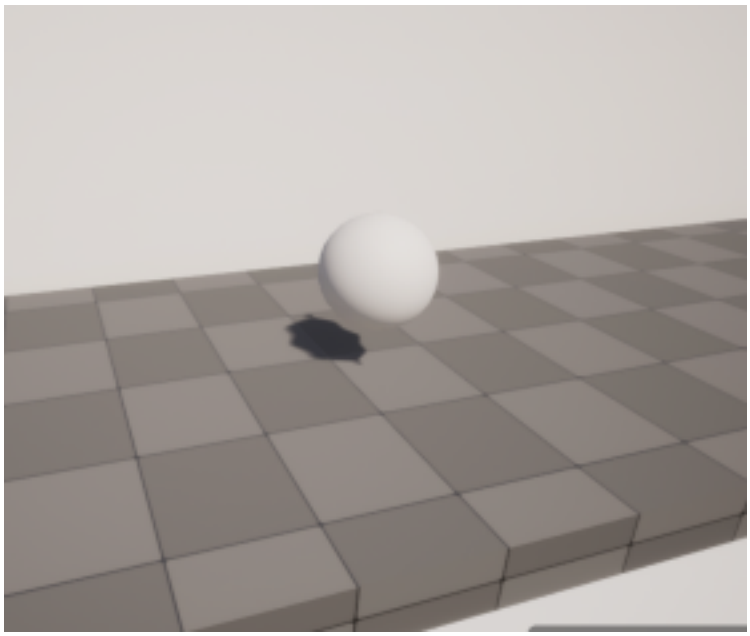
Why is this important?

Because these platforms will become the basis of our geometry, they have been created specifically to have correctly scaled lightmap UVs.

If a platform is scaled too large then the shadow maps will become **ugly, distorted and blocky.**



Platform at its default scale



Platform that has been scaled too far

Sections

To help with the creation of clean and manageable levels, levels should be broken into

sections. One section should be roughly be between 120 and 150 meters.

Sections don't need to be exactly 120 ~150 meteres, somewhat shorter or longer is fine, as long as they are broken down in a way that uses common sense.

For how to order sections into folders, please check Private

Example of the sections broken up in the Cloudbuster stage

World Origin

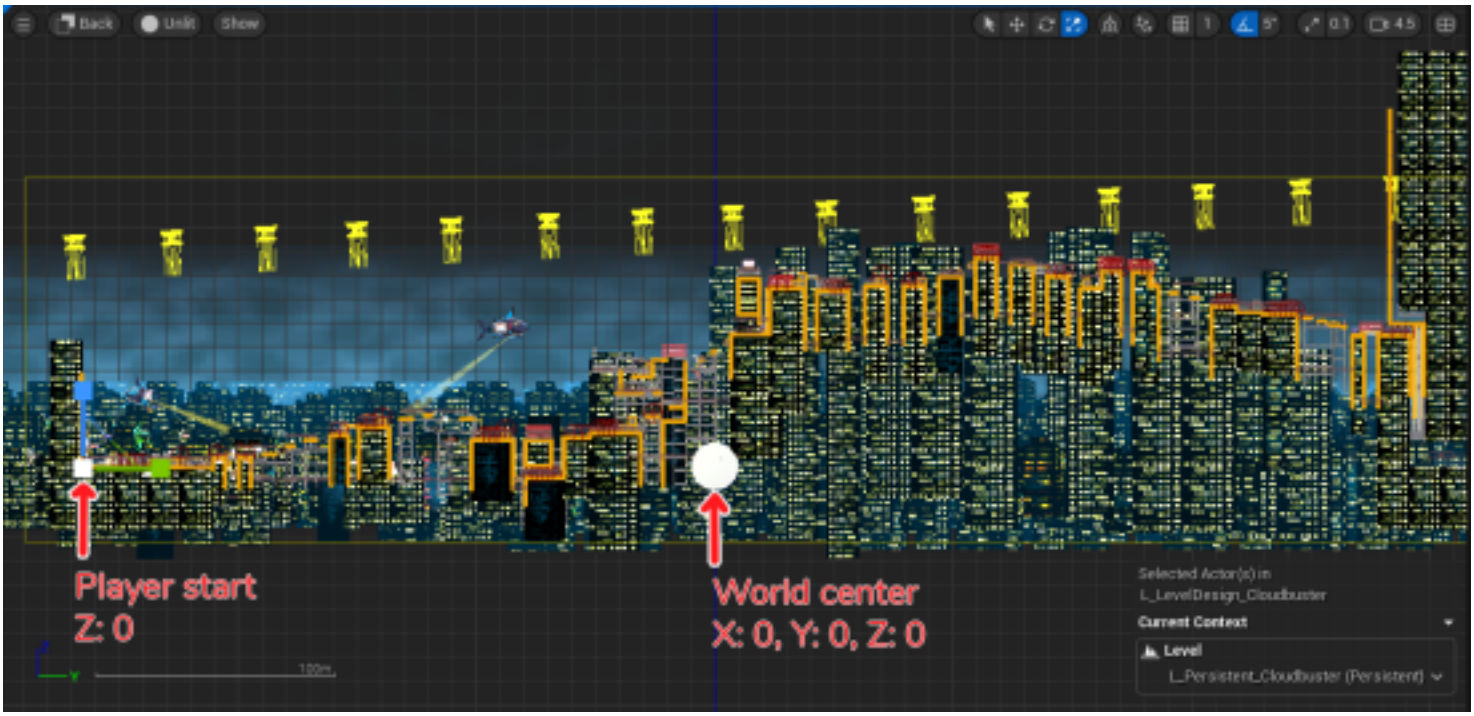
A consistent rule across levels is that the center of the level should be located as close to the world origin (i.e. X: 0, Y: 0, Z: 0) as possible and the ground level should be at Z: 0.

Having the center of the level as X: 0, Y: 0, Z: 0 ensures for a cleaner stage and minimizes problems such as .

floating-point rounding errors

The easiest way to achieve this is to firstly finish your greybox and then when it is done reposition the greybox to ensure the center of the level is at the world center.

A ground level of Z: 0 helps to keep consistency across levels and also minimizes errors. If you stage has changes in verticality, that is no problem, just ensure that the area where the player starts is at Z: 0.



Example showing how the world center and player start adhere to these rules

Depth

Because NJS is a 2.5D platforming game, it is ***crucial*** that the player and enemies and level geometry are all placed at the correct Depth of X: 0.
The programmers have implemented a feature by which the player and enemies will always adhere to a

depth of X: 0.

It is up to level designers to ensure that their levels adhere to this principal too.

- All geometry with which the player should interact (i.e. platforms, gimmicks etc.) needs a depth of X: 0.
- All geometry with which the player should not interact (i.e. art assets) should not be placed at a depth of X: 0.

Example showing assets placed at a depth of X: 0

Creating L_Environment

Introduction

This is the second level to be created and requires a finished greybox in L_LevelDesign

The approach is to build up art assets around your existing greybox, before disabling your greybox once the art is done.

For working building up your L_Environment level, it is highly recommended you work out off L_Persistent as described via Option 2 in Section 4 of Private

For this level it is also recommended that you work in the following passes:

- **First pass** - Block out the Gameplay area of the level with art assets before disabling the greybox
- **Second pass** - Expand on the blockout in Gameplay , include Foreground and Background assets
- **Detail pass** - Apply the details to Gameplay , Foreground and Background •
- **Lighting pass** - Place lights throughout the scene where required.

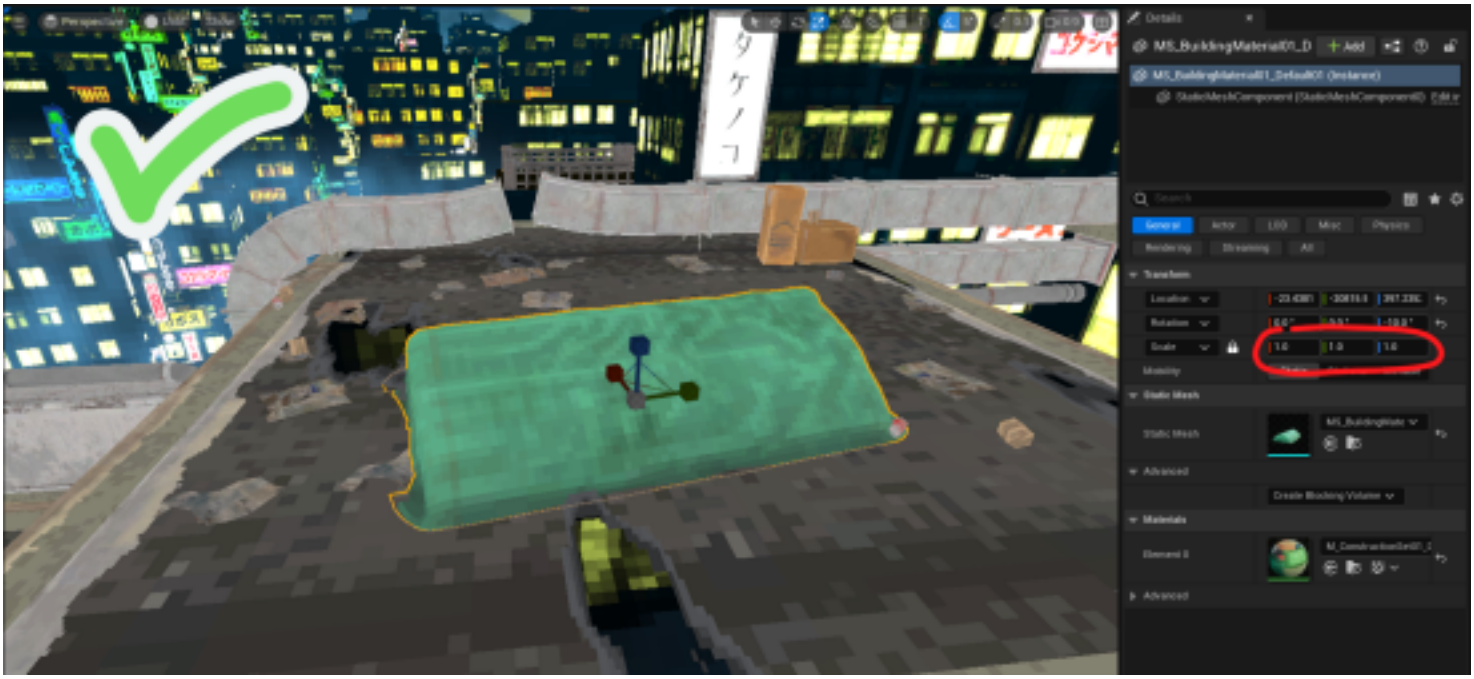
For more details on Gameplay , Foreground and Background be sure to read Private

Before the passes are broken down in more detail, it is important to cover some prerequisites of placing art assets.

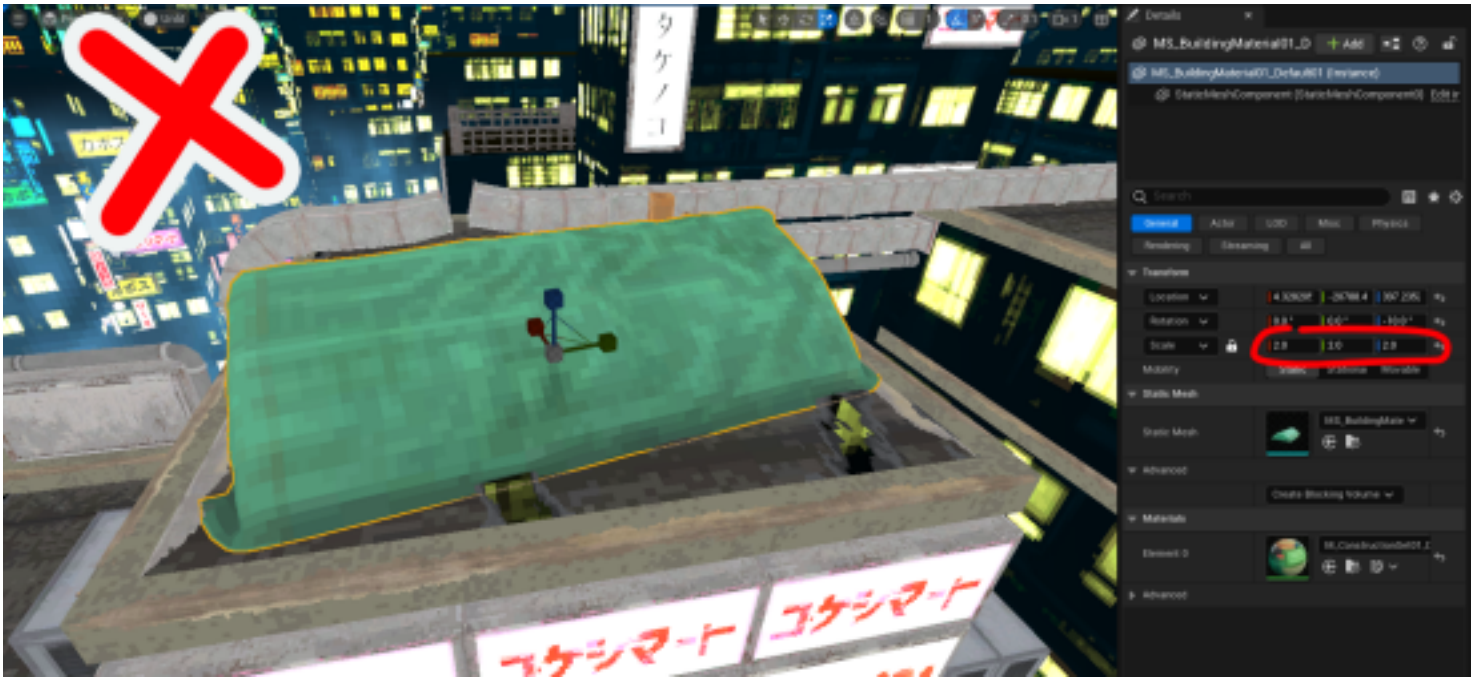
Scale

When working with art assets, it will be inevitable that things need to be scaled up or down in size. However as a rule of thumb, assets should not be scaled up or down more than **10%** of their original size

When scaling assets, it is important to ensure that textile resolution stays consistent across the level and that no assets look stretched or otherwise out of place.



Example of an art asset at a correct scale (0.90 ~ 1.10)

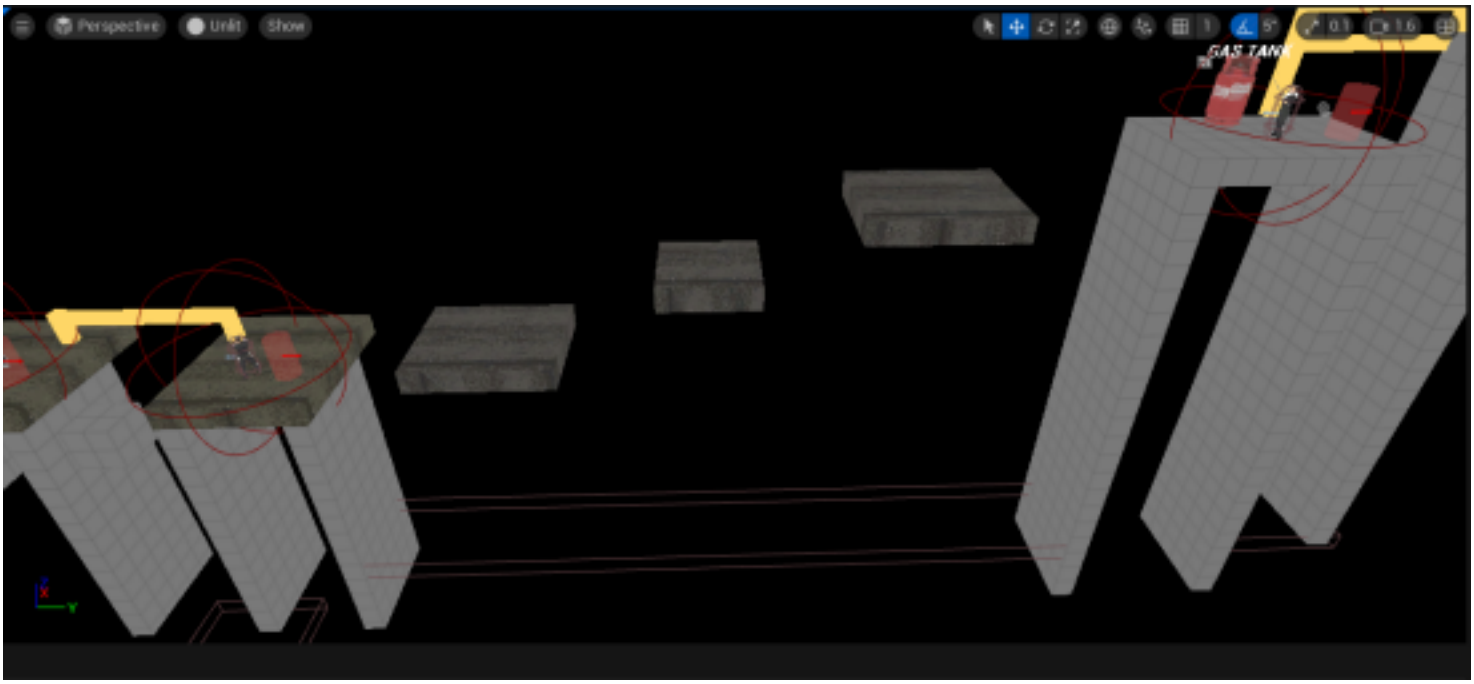


Example of an art asset an incorrect scale

Story and Justification

It is important to justify your greybox with art assets to create a cohesive story within your stage. To give a better understanding of what this means, I will use an example.

Below is the image of some floating platforms in Cloudbuster L_LevelDesign



Example of floating platforms in Cloudbuster L_LevelDesign

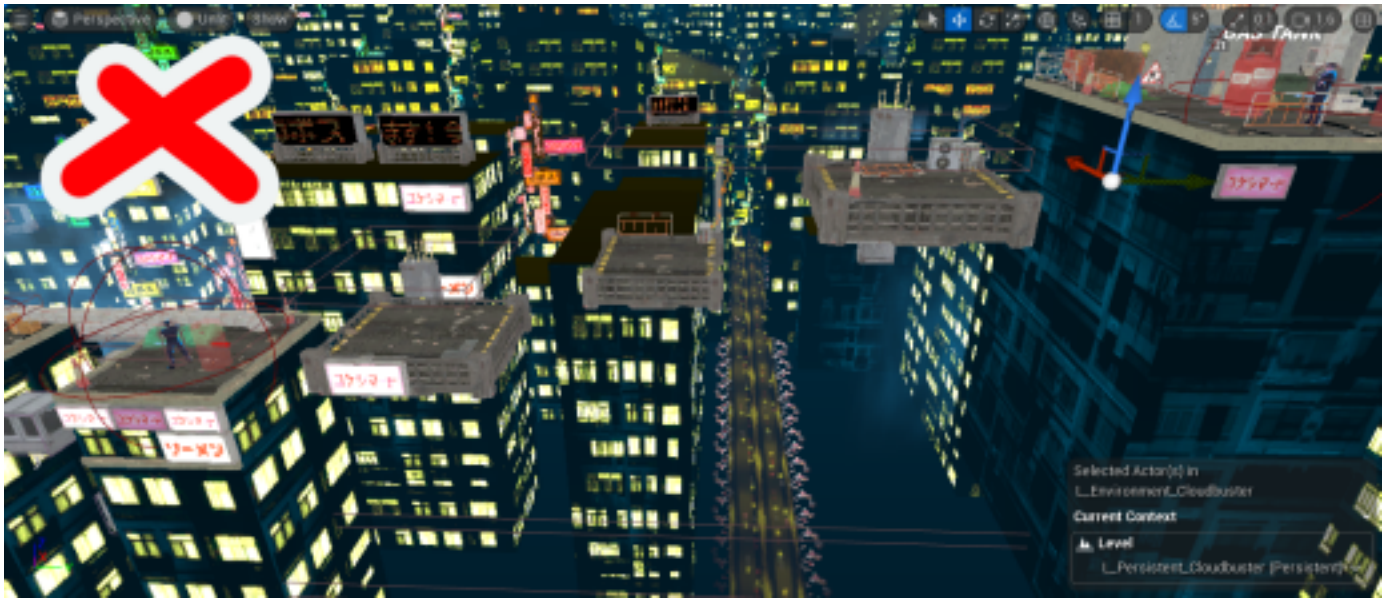
When creating art assets, we want to do two things:

- **Keep their original design intention.**

  If we make them buildings then they will have walls on their side from which the player can wall jump. This will change their original intention of being tricky platforms

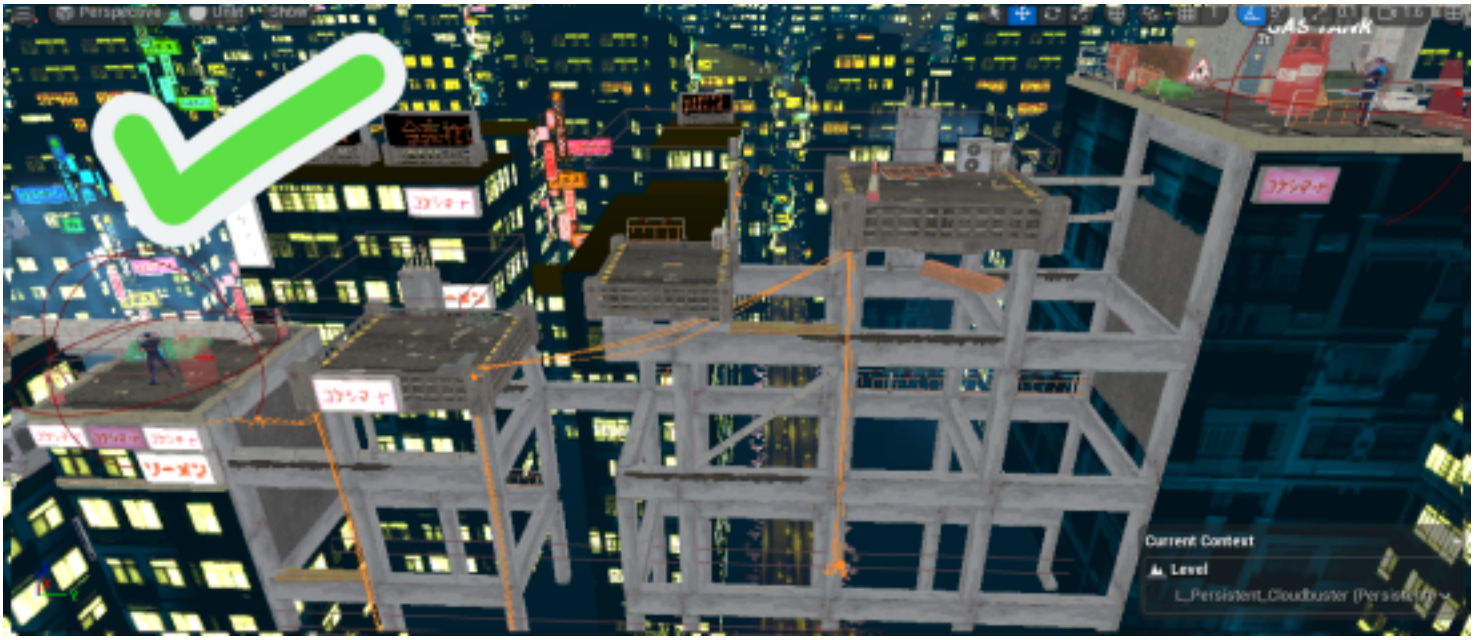
- **Make them look good.**

  You need to justify the visuals in a way that makes sense in the context of the level's story. If the platforms are just floating in the sky for no reason, it won't look good and makes little sense.



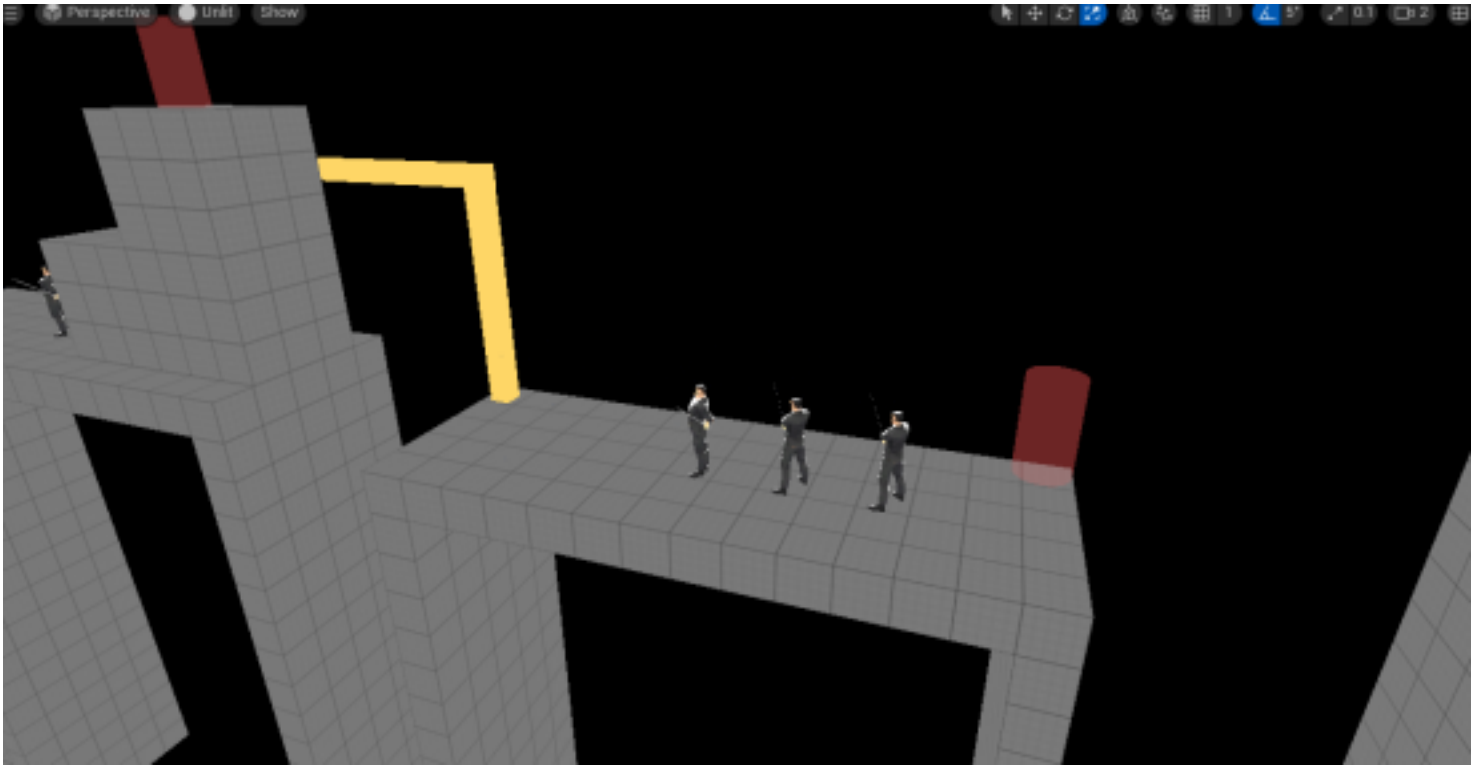
Example of poor justification, there is no reason for Neo-saitama to have random floating platforms

The solution in this case was to make them part of a construction site.
This gives a structure for the platforms to be attached to, while also allowing them to have nothing beneath them, all why justifying their existence in the story of Neo-saitama.



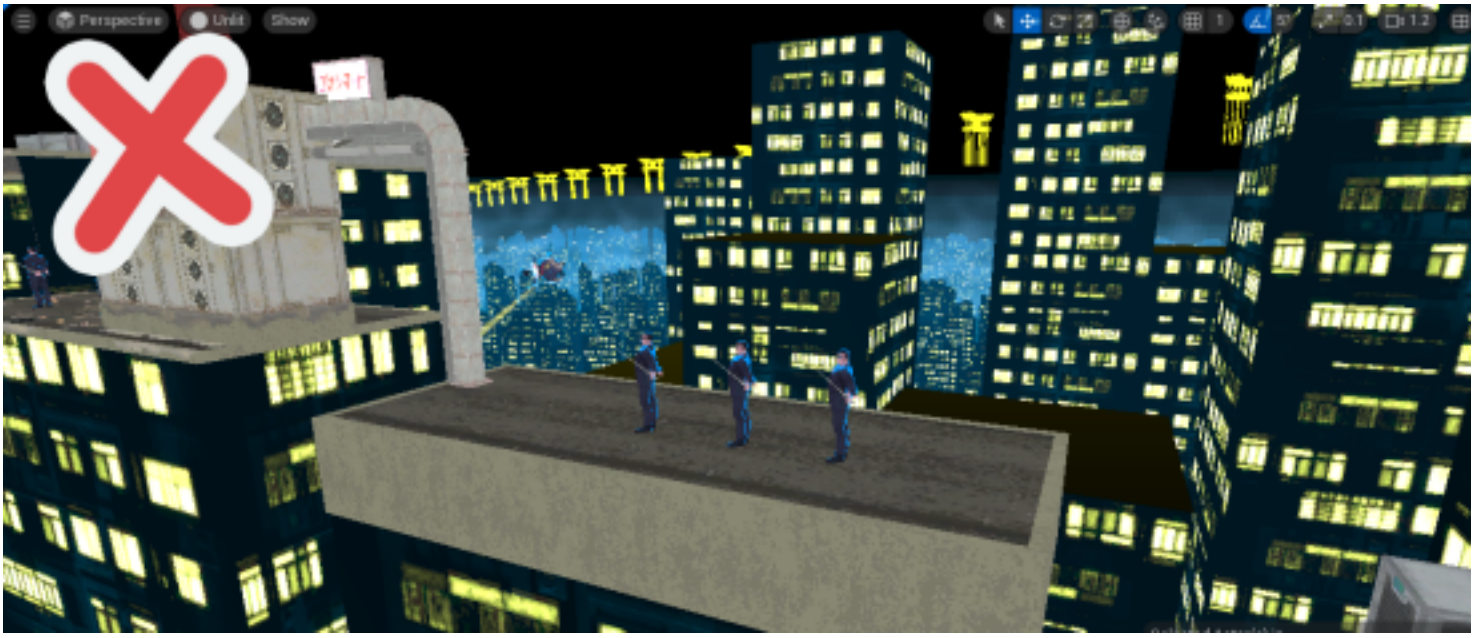
Example of good justification, the platforms have an appropriate reason to be there

Another example of justification and story is the following platform.
Here we can see three zako standing on a rooftop, ready to ambush the player. We need to think about how we can justify their existence in the stage and give them a story.



Example that shows 3 zako standing on a rooftop in the greybox

When doing the art pass, it would be possible to just justify the greybox by adding a building and calling it done. The platform is now justified in the world, but there is no story justification.
It leaves questions like how did the Zako get up there? Why is there just an empty rooftop like this in the city? etc.



Example that shows the zako standing on a rooftop, but with little story reason

The solution, in this case, was to extend the building out and make it look residential and add a loft with

an open door. This creates the story of the zako having burst out onto the roof via the loft, ready to ambush the player.

Example that how to build an interesting story for the zako

Now that **Scale** and **Story and Justification** are hopefully clear, it is time to discuss the four art passes required for creating the level.

First Pass -Blocking out Gameplay area

For this pass, make sure you are starting in L_Persistent and you have set up level streaming correct so you can can view L_LevelDesign while also being able to place assets in L_Environment .

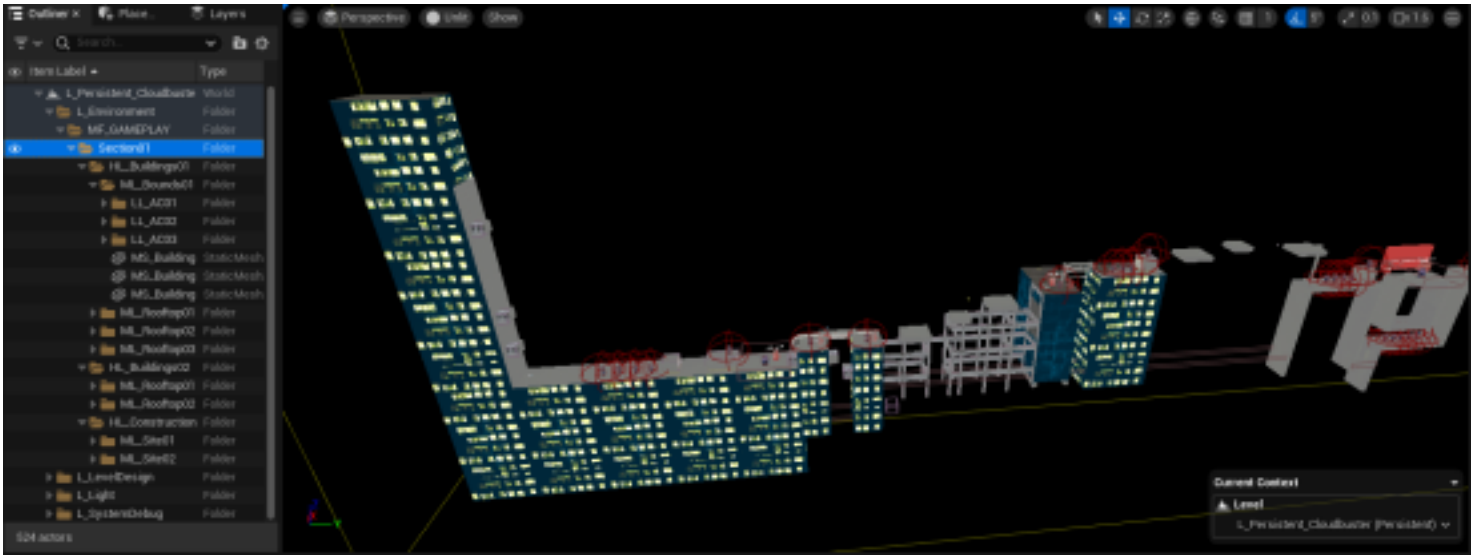
Your goal is to block out the art assets in the **Gameplay** area of the level, while matching the greybox. You should be aiming to just fill in the broader sections of the level without focusing on the fine details at this time

> For **Gameplay** area, ensure you have read Private (https://app.clickup.com/7518758/docs/75eh6-7203/75eh6-29747)

Example of how to start working on your first pass

Here is an example of a finished first pass of Section01 of the Cloudbuster stage.

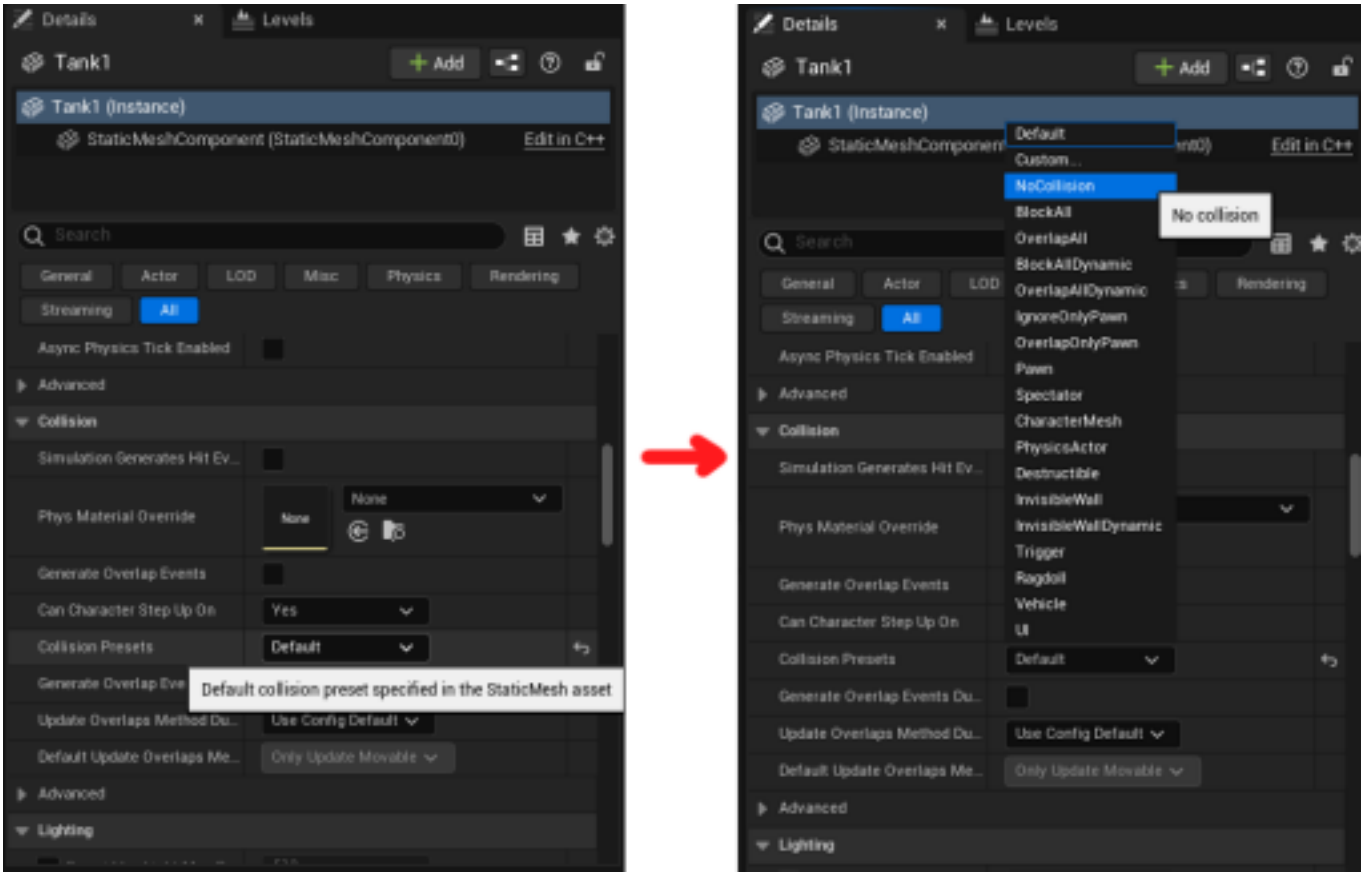
As you can see, art assets have been added closely to where the greybox is, turning it from just a greybox into the start of a full stage.



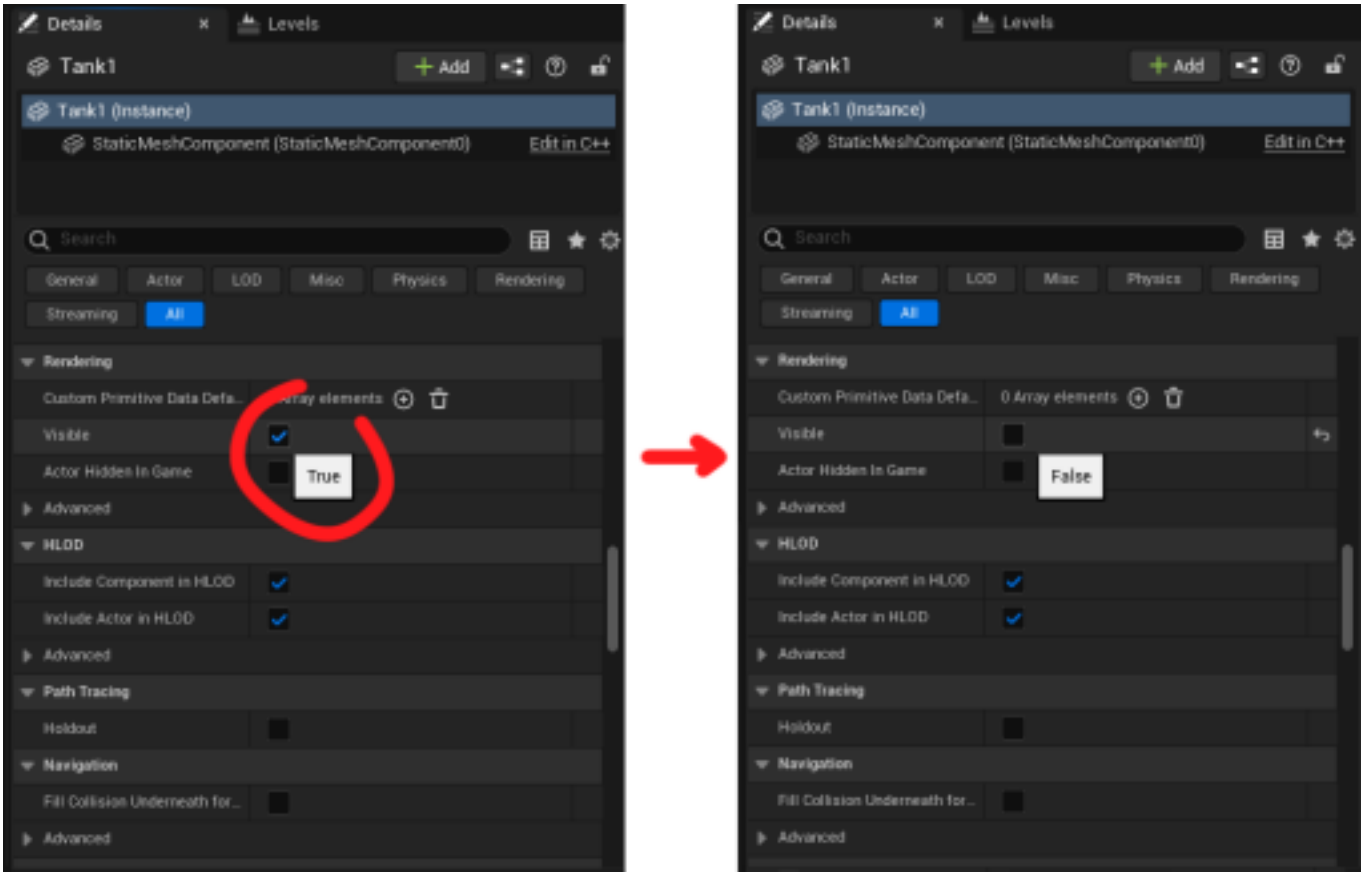
Example showing the first pass completed on the Cloudbuster stage

Once your first pass is done, it is time to disable the collision and visibility of the geometry in `L_LevelDesign`. Here is a demonstration on how to do so:

- **For abetter view of the details window, expand this text.**



Close up image showing the turning off of collision on an object.



Close up image showing the turning off of visibility on an object.

Do **NOT** delete the geometry in L_LevelDesign

Do **NOT** just turn off visibility and leave collision on

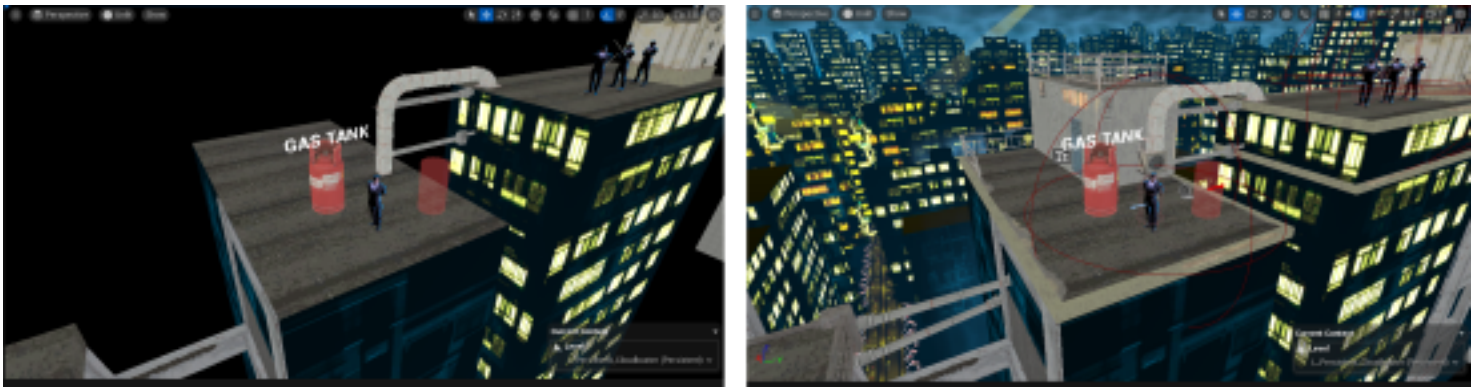
Do **NOT** forget to create the hidden folder in L_LevelDesign like described in Private (_https://app.clickup.com/7518758/docs/75eh6-7203/75eh6-29747

)

When this is done for all the sections of your stage, the first pass is **complete.**

Second Pass -Expand Gameplay , Foreground , Background

For this pass, L_LevelDesign should already be completely disabled and it is now about expanding the **Gameplay** area and building up **Foreground** and **Background** .



Example image showing First Pass (Left) and Second Pass (Right)

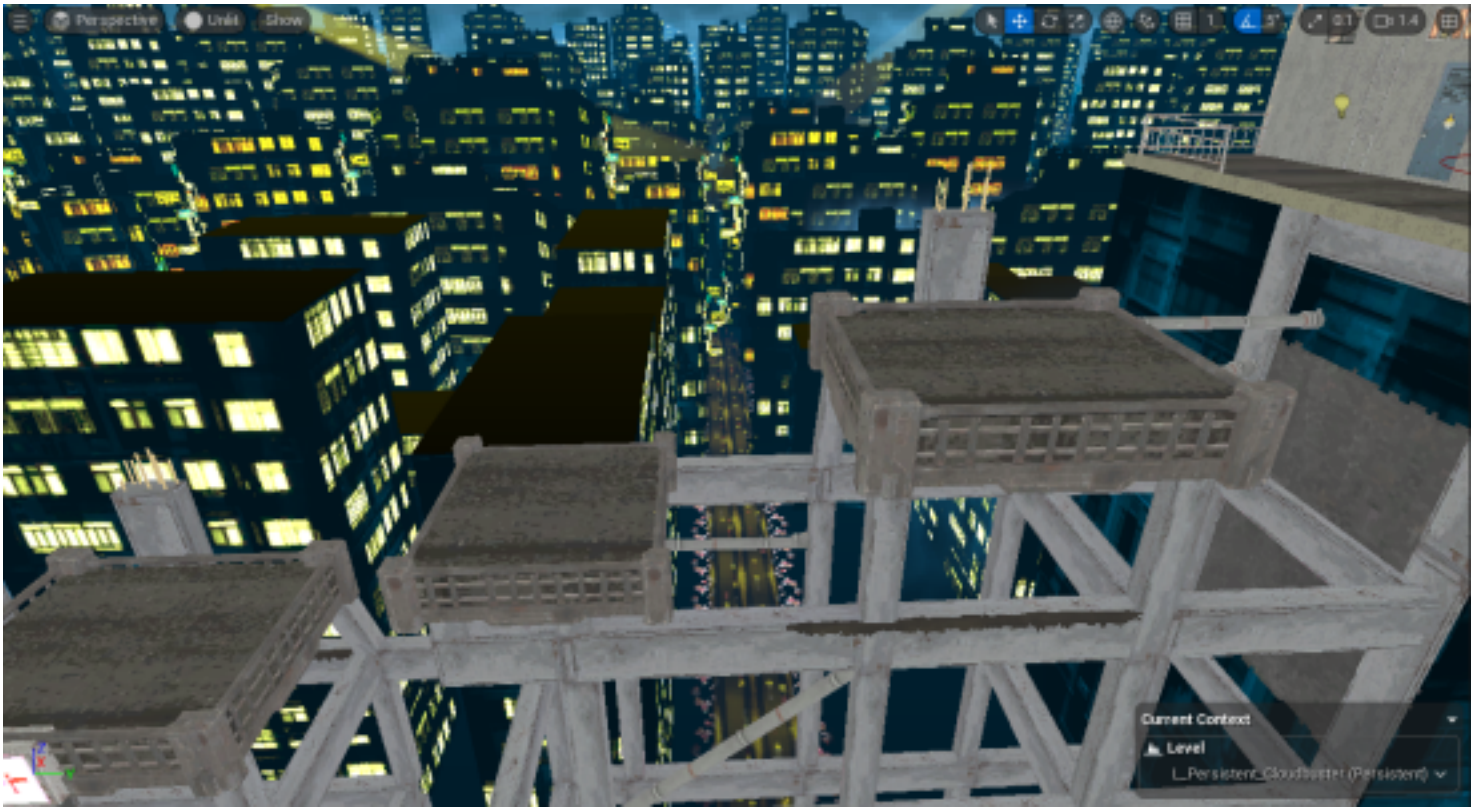
For **Gameplay** your focus should be on expanding the art assets you laid out in the first pass.

For Cloudbuster that meant adding more details to make the buildings recognizable as buildings. Trim was added to the edges, along with railings, a loft area, pipes etc.

Example of new assets painted in red

For **Foreground** , it is time to start connecting the story you have built in the **Gameplay** area with the rest of the level.

The construction site gave a very clear view of the foreground and background, so it was the ideal spot to break up the buildings in the back with a busy street.
The idea behind starting your **Foreground** and **Background** in the second pass is you want to build them up so they compliment the **Gameplay** area, so you need a rough **Gameplay** area first.



Example of a highway visible from the construction site

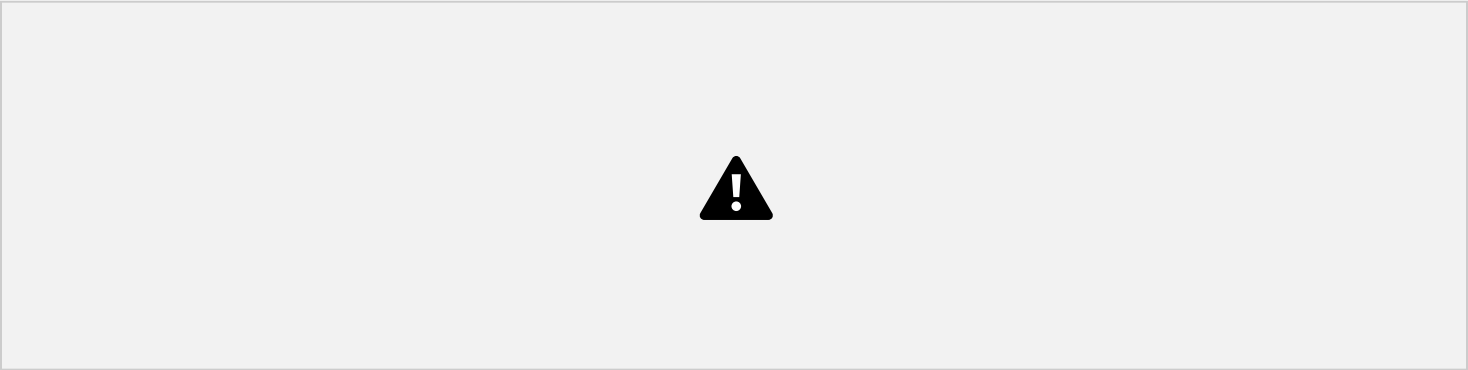
The **Background** requires less attention, as long as you just make something that fits the style of the stage so far.

Example of a Cloudbuster's Background after the first art pass

Once the second pass of your level is done, the art should be mostly completed across the whole of the stage, ready for the final detail pass.

Detail Pass - Add detail to the stage

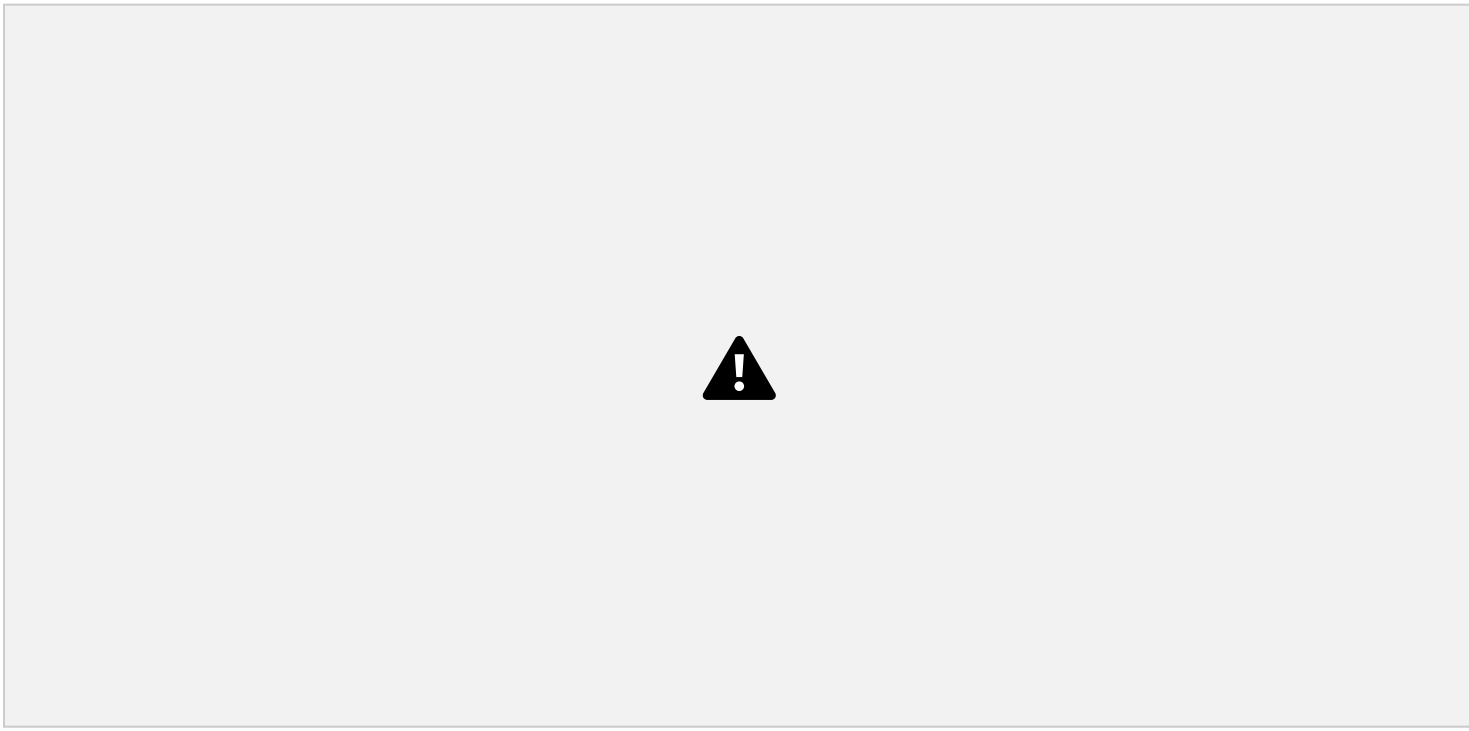
For this pass, is time to finish of placing art assets and polish off the visuals. The point of the detail pass is to really bring your environment to life and finish fleshing out the story and justification.



Example image showing Second Pass (Left) and Detail Pass (Right)

For **Gameplay** your focus should be on adding the finishing touches to make it all look good.

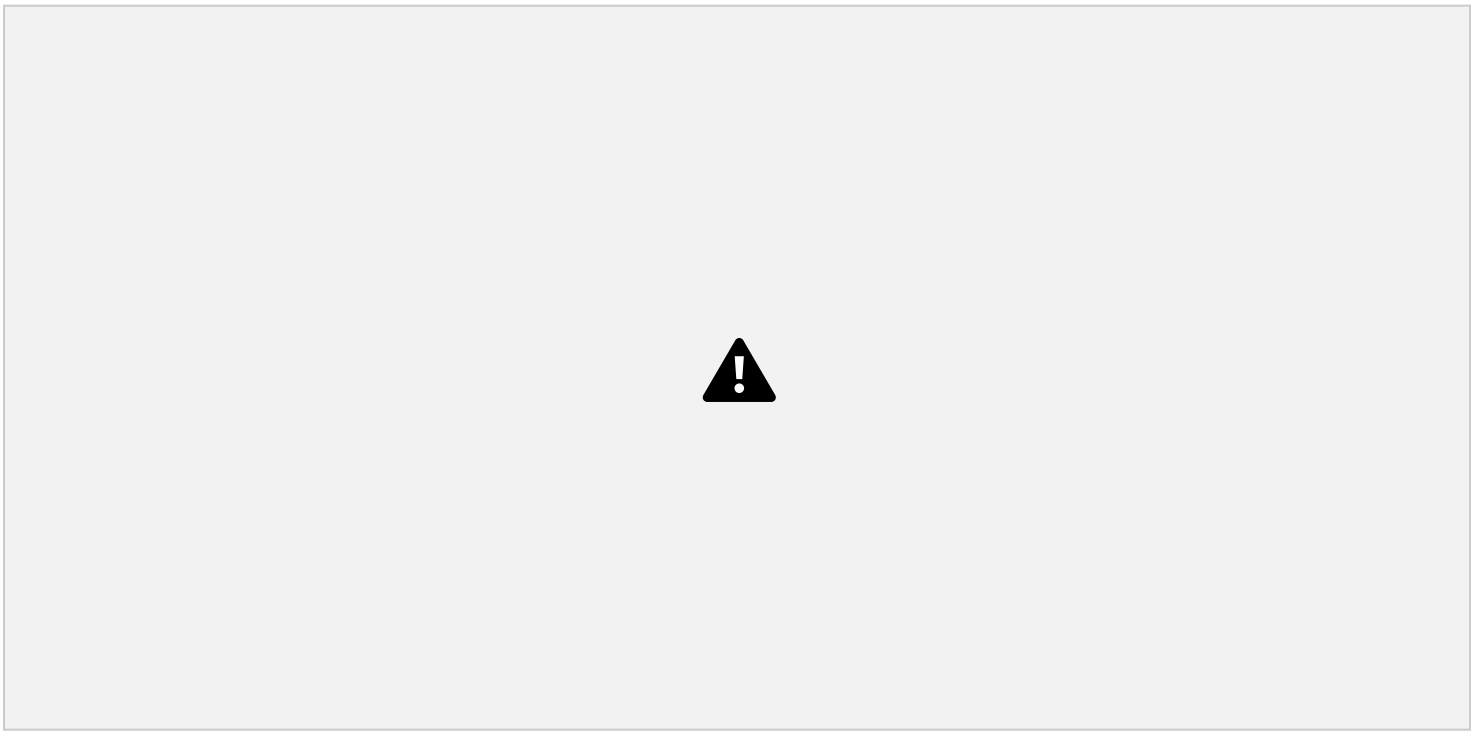
For Cloudbuster that meant building up props that gives the feeling of a dirty city, with certain parts under construction. It would be strange to have construction sites with no evidence of them reflected in
the surroundings.

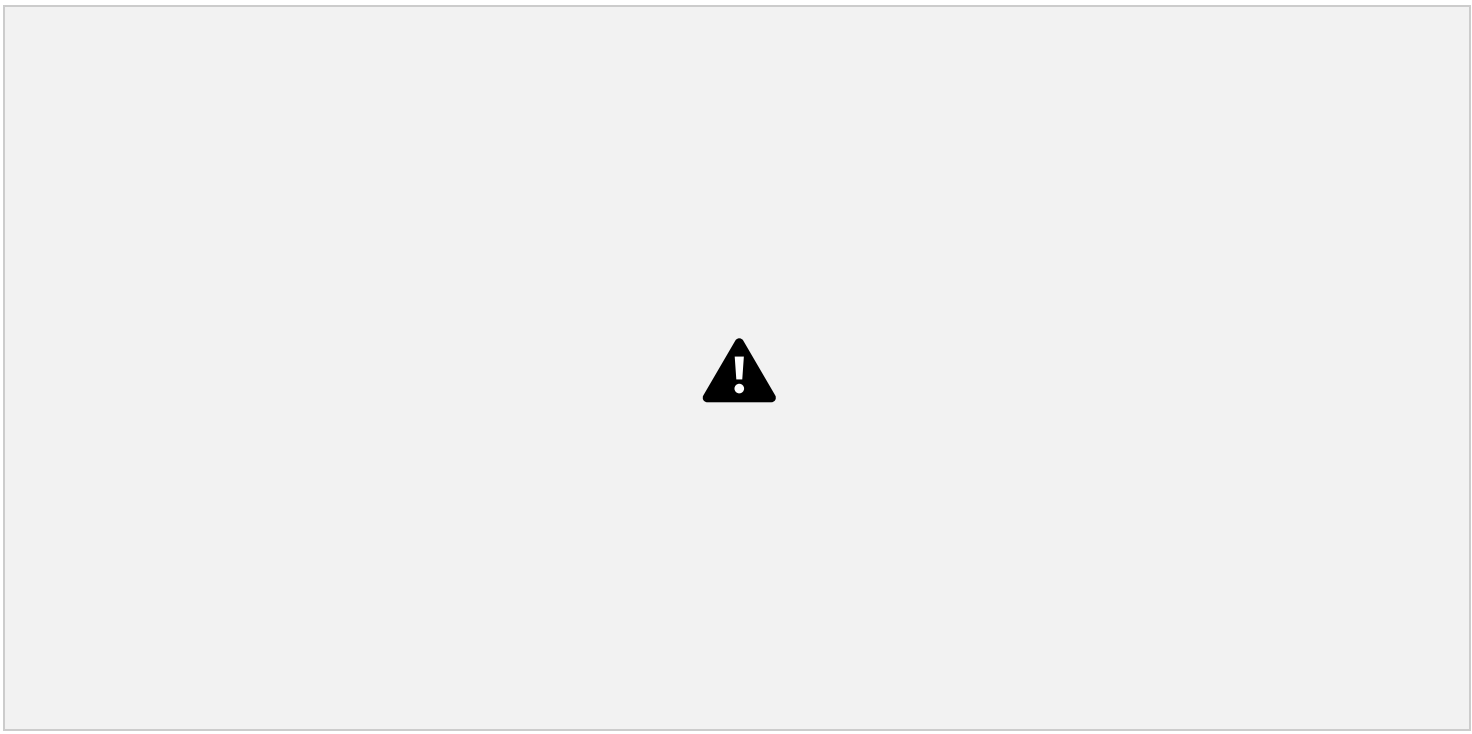


Example showing the new details highlighted in red

Details can also be more than just superficial, like in the example below.

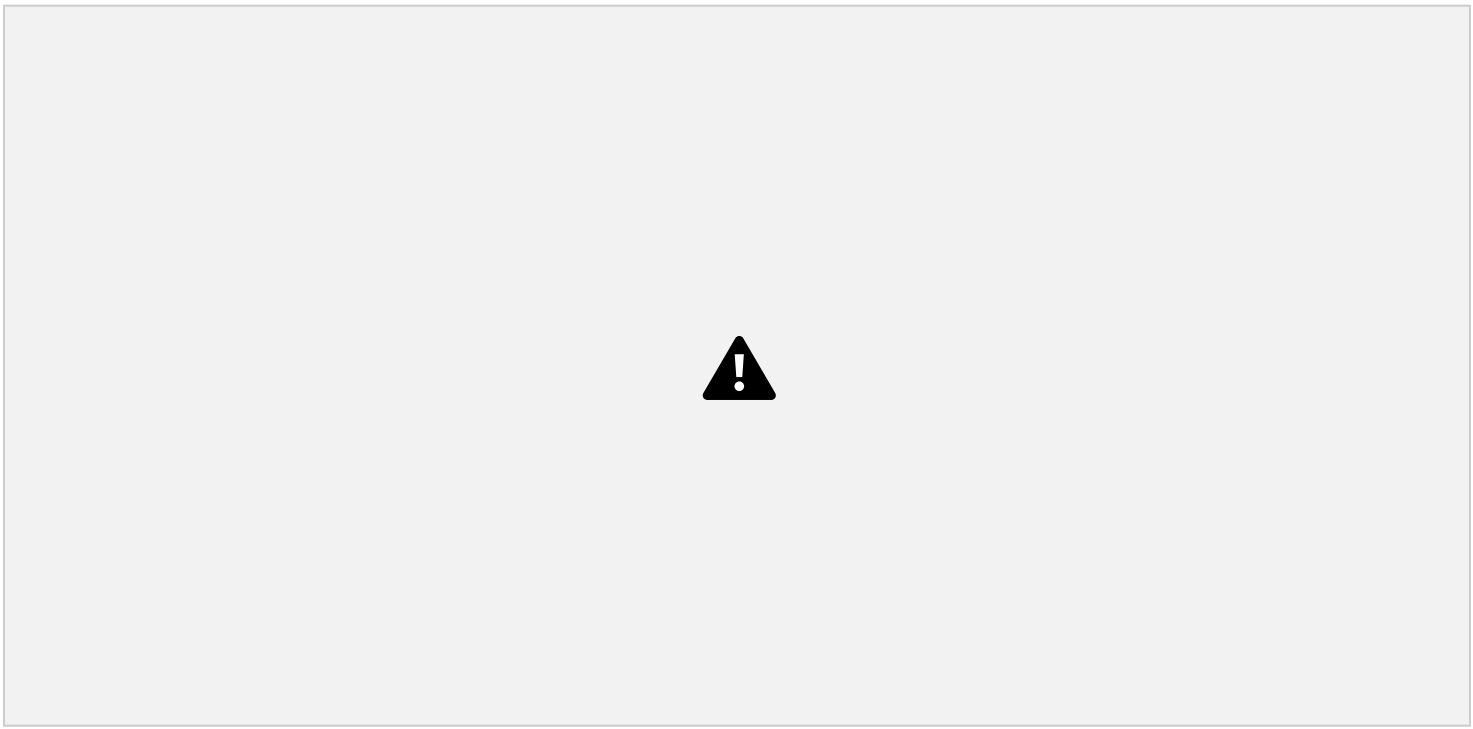
Here the details of orange wires have been added to the construction site. Not only do they add to the general aesthetic, but in fact they add as a visual guide to help signal the player where to keep going.

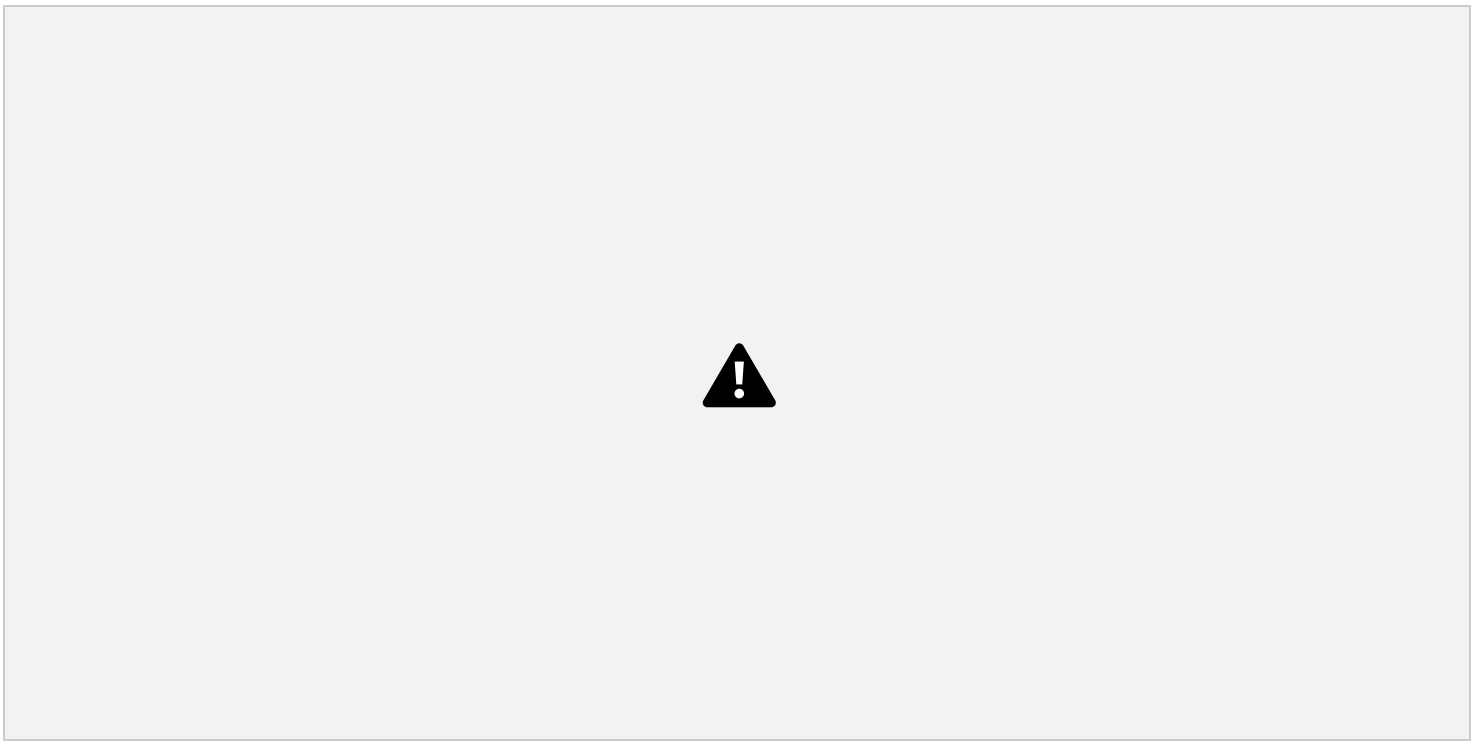




Example of how details can draw the player's eye and help guide their advance

It is important to fully flesh out the **Foreground** and **Background** to fully fit the stage at this point too.





So in summary, the detail pass should be used to expand the story of the world while providing helpful elements to the player too.

> Once the detail pass is done, it is important to check that no objects will interfere with the gameplay, i.e. debris in range of the player or zako

Example of checking nothing blocks the gameplay area, move or remove it if so

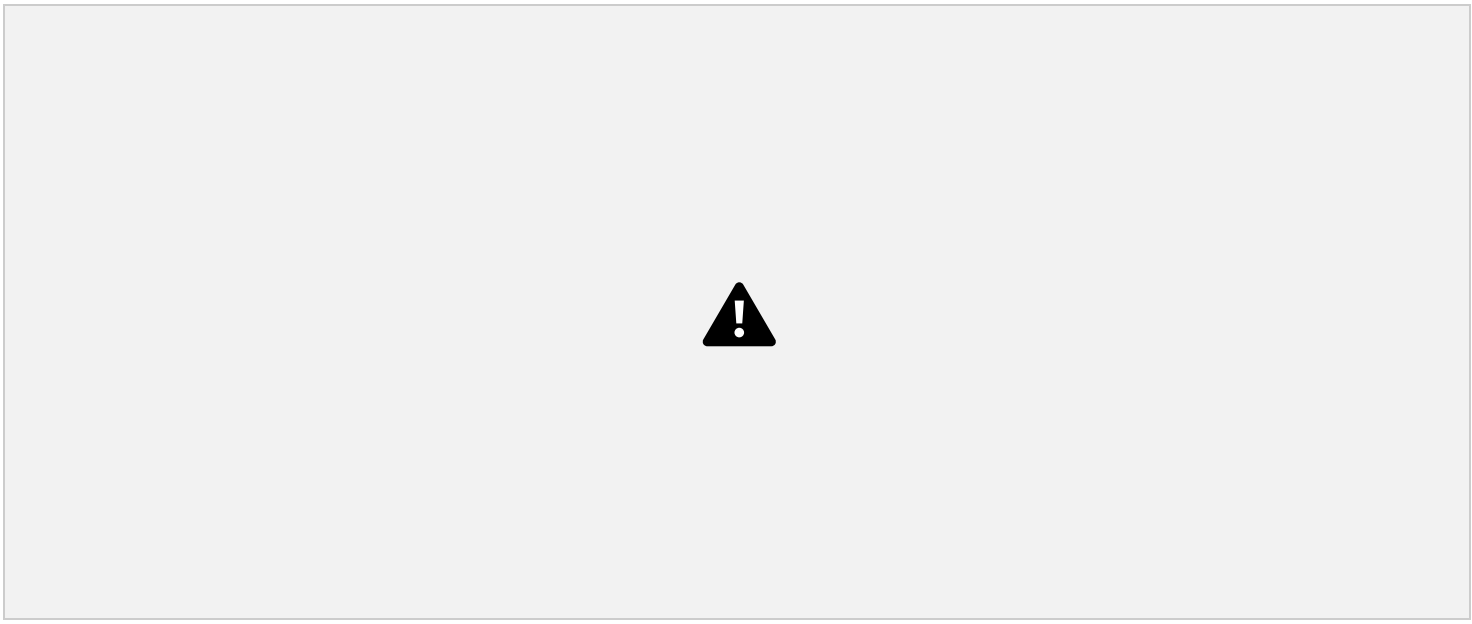
Lighting Pass - Adding the lights

The last thing to do is to add lights to your scene where necessary.

For general rules on what lights to place and where to place them, refer to Private

In regards to the concept behind placing lights, take an approach similar to detail pass. Make your environment look good while also using light to guide the player.

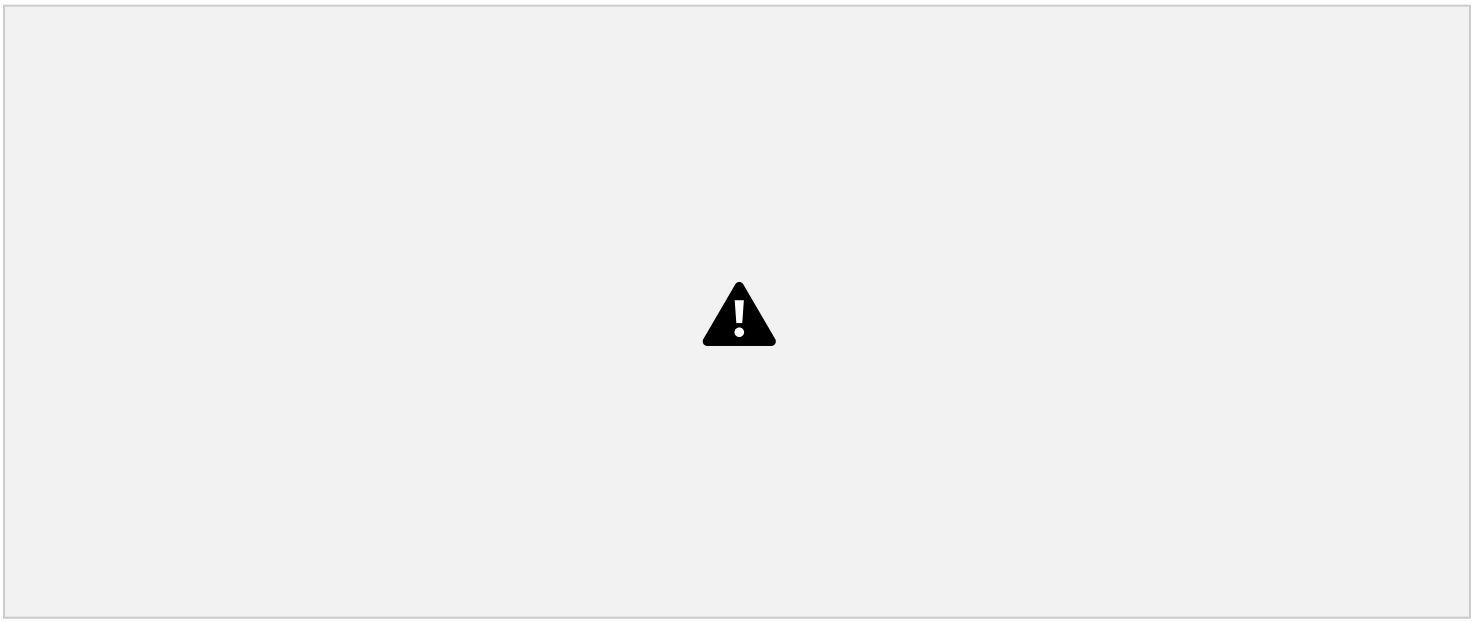
In the following example, the player must move from a platform at the bottom left of the screen to the top right, but the dark light makes it very hard to see what is going on and where the platforms are.



Example of a poorly lit area

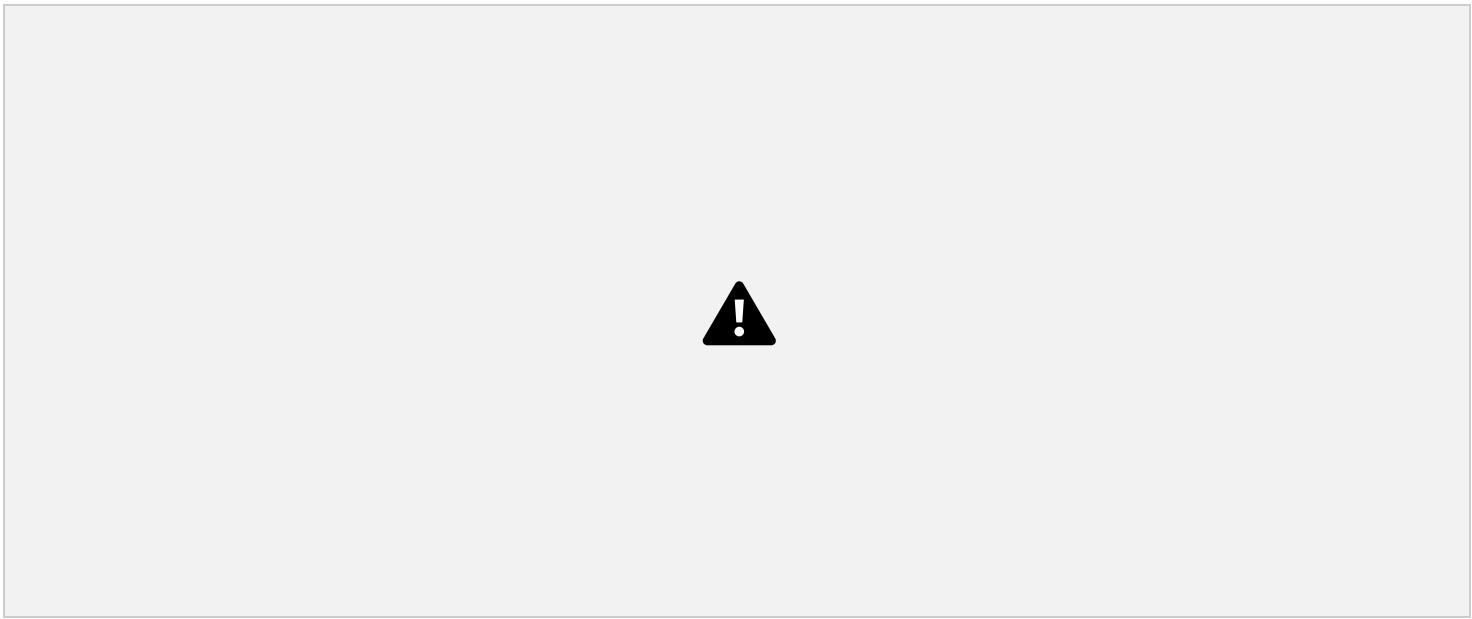
This is where you can use good lighting to help guide the player. Remember, players will usually always want to move towards light.

A good solution would be to use a light that matches the current lighting of the level to highlight the next platform the player needs to move to.



Example of good lighting practice to highlight the player's path

A bad solution would be to do something like light the pit or use a colour that does not match the scene. This could make the player think they actually need to jump down there, which can lead to anger and confusion.



Example of bad lighting practice that could confuse the player