

Worlds

- [World Tutorial series](#)
- [Basic Upload and Modelling - Worlds](#)
- [Materials for Worlds](#)
- [Lighting](#)
- [Prefabs](#)
- [Physics](#)
- [Level Design](#)
- [Particles](#)
- [Post Processing](#)
- [Audio](#)
- [Udon](#)
- [Final Things \(Version control, optimizations\)](#)
- [World Work in Blender \(Advanced Users\)](#)

World Tutorial series

Have you ever wanted to make your own world, but never knew where to start? I am currently making a step-by-step tutorial series about how to make your own room in VRChat for the absolute beginner!

<https://www.youtube.com/embed/uLi52YrrDmY>

[The entire playlist](#)

Basic Upload and Modelling - Worlds

The easiest and most comprehensive way to learn world creation is by watching the tutorial series linked at the top of this guide:

<https://youtube.com/playlist?list=PLPdWkxUSZ65Fp6ICr\>

<https://www.youtube.com/embed/uLi52YrrDmY>

I recommend using VRWorldToolkit: a Unity Editor extension for making VRChat worlds[github.com]. It will help you optimize your world.

EasyQuestSwitch[github.com] helps to switch a world (or avatar) from PC to Oculus Quest: allows to switch properties of all specified assets (like the shader of a material) to a specified PC or Quest version with one button.

Probuilder

Great for making/prototyping levels and level geometry. Effectively a simple 3D modeler inside Unity.

<https://www.youtube.com/embed/ofzw6hKhI3k>

<https://www.youtube.com/embed/YtzIXCKr8Wo>

<https://www.youtube.com/embed/bigj13SU1rs>

https://www.youtube.com/embed/d3_2h4cN4cY

<https://www.youtube.com/embed/lmLG4nC9tm0>

Tips:

Create empty Gameobjects (GameObject>Create Empty OR Ctrl+Shift+N OR right click Hierarchy>Create Empty) and use them as folders to organize objects! BEFORE MAKING ANYTHING AS A CHILD OF THE GAMEOBJECT, RESET THE TRANSFORM!!! DO NOT MOVE THE EMPTY GAMEOBJECTS THEMSELVES, MOVE ALL OF THE THINGS INSIDE OF THEM!!! (If you don't this will screw up progrid!!!)

Z-Fighting, also called stitching, occurs when multiple 2D planes are in the exact same location (e.g. a picture on a wall). Avoid this by moving the object that is supposed to be overlapping the other a miniscule amount forward.

Polybrush

For simple sculpting inside Unity. Great for making natural landscapes.

<https://www.youtube.com/embed/JQyntL-Z5bM>

[docs.google.com]

Asset hunting

Tip: only download free assets you think you'll actually use; you can always come back to get them later. You can find plenty of free and paid assets on the Unity Asset store. I'll list some additional free asset sites below:

[Kenney Free Assets](#)

Textures

Check the dimension metrics for the materials that you download, if they are available. This will inform you how much you need to change the tiling setting on them to get them to scale properly. They will often be in deviations of a meter (e.g. 1m x 1m).

When hunting for textures, prefer lossless .PNGs rather than lossy .JPGs. Unity compresses the textures as mentioned above; using lossless images will lead to higher quality textures and smaller texture sizes in Unity.

[(Completely optional) If you want to losslessly compress the .PNGs on your computer you can use a program like PNGGauntlet. Unity does not store textures as common image formats like PNG or JPEG because of their relatively slow decompression time.]

Texture Search Engine[textures.one]

Free PBR texture sites:

- <https://ambientcg.com/>
- <https://texturehaven.com/>
- <https://freepbr.com/>
- <https://3dtextures.me/>
- https://www.poliigon.com/search?type=all&refine_by=assets-free
- <http://textures.com/>
- <http://gametextures.com/freebies>

If a non-color texture appears colored in Unity (like a slight green), bring it into a photo editing program like GIMP and export it as a grayscale image (8bpc GRAY in GIMP, GRAYA if it has an alpha/transparency channel).

[Substance Player](#) is a free program where you can easily create custom, high-quality textures using .sabar files. You can get these files for free at [AmbientCG.com](https://www.ambientcg.com).

<https://www.youtube.com/embed/Wx9vmYwQeBg>

Models

Hunting for models is far more difficult than for textures. Ensure that:

The file format is .obj or .fbx, preferably the latter

The poly count is low (or you'll have to decimate it yourself)

I would recommend checking the Unity asset store first. If you can't find it there then Google `"*object name* 3D model"`.

[Free 3D Model Sites & Resources](#)[devga.me]

Decimation (lowering the polygon count of a mesh) can be done manually by removing/merging vertices/faces or automatically.

To do it automatically in Blender, you can select quick decimation in CATS or use the decimate modifier on a mesh.

[The decimate modifier is also great for creating LODs if you want to go the extra mile.]

There are also programs meant specifically for decimation and retopology, like the free Instant Meshes.

DO NOT import .blend files directly into Unity! Export them as .fbx, then import!

You will also need a skybox. You can get some from the Unity Asset Store, or I also like

[TowelCloud's skyboxes](#)[gitlab.com].

Materials for Worlds

<https://www.youtube.com/embed/M7IXt4d8R4>

For an ideal introduction I would recommend watching my video on the topic of materials in my Create Your First VRChat World series or reading its [respective script](#).

Using a community created PBR shader (like [Mochie's](#)) will increase visual quality, and also offer many additional features including baked specular lighting! This will allow specular highlights to appear with baked lights, which is not possible when using the standard shader.

[Height \(AKA displacement\) map](#)

This texture insets or extrudes different parts of the material.

In Unity you should typically set this pretty low or else looking at the material at an angle will ruin the effect, which is especially noticeable in VR.

These maps are more computationally expensive than all the others (Oculus' documentation advises not to use them at all because of this). They can look great in certain circumstances but if not necessary then they should be avoided.

“Originally posted by **docs.unity3d.com**:



“From left to right in the above image: 1. A rocky wall material with albedo assigned, but no normalmap or heightmap. 2. The normal assigned. Lighting is modified on the surface, but rocks do not occlude each other. 3. The final effect with normalmap and heightmap assigned. The rocks appear to protrude out from the surface, and nearer rocks seem to occlude rocks behind them.”

Displacement maps are calculated using parallax mapping in Unity. This is a comparatively low-performance real-time approximation but also the not incredibly accurate, and the illusion breaks at intense angles. Using shaders like Mochie's standard allow you to control how many "steps" the parallax mapping has.

Secondary Maps (Detail Maps) & Detail Mask

Used for extra detail or imperfections, such as cracks or scratches in a surface or the tiny pores of skin. Originally posted by **docs.unity3d.com**:



This character has a skin texture map, but no detail texture yet. We will add skin pores as a detail texture.



The end result, the character now has subtle skin pore detail across her skin, at a much higher resolution than the base Albedo or Normal map layer would have allowed.



Detail textures can have a subtle but striking effect on the way light hits a surface. This is the same character in a different lighting context. You can find many [separate surface imperfection maps](#).

Thickness map

Used for subsurface scattering (SSS), to simulate when light passes through a material. A great example of when to use it is for skin, leaves or cloth.

You can generate this map by [inverting the normals and baking ambient occlusion](#) in Blender.

You will need a custom shader to apply it, like Silent's [photorealistic SSS](#) shader.

You may also want to use custom shaders for particular effects, like a [water shader](#) or a [faked volumetric lighting shader](#) to make a light source really pop.

Lighting

<https://www.youtube.com/embed/1pNMo2UQY1o>

Types of lighting:

- Real-time- calculates lighting every frame
- Baked- pre-calculates lighting on the editor's computer and generates light maps (textures that contain lighting and shading) which are overlaid on those objects
- Mixed- combination of baked for static objects and real-time for dynamic objects

Types of lights:

- Directional - Essentially the sun. There should be no more than 1 per scene. [Its rays are infinite, so its position does not matter but its rotation does.]
- Point - projects light outward in all directions from the center. Simulates lights like lightbulbs.
- Spot - projects light outward from the center in a cone. Simulates lights like spotlights.
- Area - projects light outward from a 2D rectangle or circle. Simulates lights like office ceiling lights. These lights in particular can cast much softer shadows, depending on its size. There is no real time version in Unity by default; these lights must be baked.

Real-time lights in VRChat will usually either look terrible, tank your framerate, or both.

Lighting Dynamic Objects

So how do you light dynamic objects? Light probes. These probes pre-calculate how an object would be lit at that probes location. Many of these probes are placed in a group, and when dynamic objects move through this group they blend the probes and are lit accordingly.

Add a light probe group to your scene. Place probes around every point there is a lighting change. Spawn a new gameobject like a capsule (set its tag to EditorOnly so it doesn't upload) and drag it around to see how it takes on the lighting from the probes.

Reflection probes

Used for reflections. They act as a camera, taking 6 pictures from where they are placed (the 6 directions) and combining that data into a cubemap (a spherical image). Overlapping reflection probes blend.

Box projection is a feature that can create parallax, by projecting the cubemap to the bounds of the reflection probe. It is preferable for interiors, especially ones that are more rectangular shaped. Outdoor environments (or other open environments where parallax would look weird) can keep this option turned off.

Box projection can achieve "good enough" looking reflections for static objects without having to resort to other far more expensive reflection techniques.



Reflection Probe Box Projection [docs.unity3d.com]

DO NOT use screen space reflections in VR. They are incredibly taxing on performance!

Screen space reflections only reflect what is in screen space. So if there is a reflective surface (like water) with a reflected object, if the player looks down at the surface without the object in view then the reflection also disappears even when it shouldn't.

In addition, since they are "screen space" and VR renders two screens, sometimes there can be discrepancies between the screens.

Lighting settings

Go to Window>Rendering>Lighting Settings to bring up the window.

- If your scene is completely indoors with no windows, turn off the skybox lighting.
- If it is an outdoor scene, turn skybox lighting on and set the directional light source in the lighting settings.
- If it is an overcast scene, make sure your sky is set to a cloudy skybox. You may want to adjust the intensity of the environmental lighting.

Texels per unit

Texels are like pixels placed onto an object, and used to calculate the baked lighting. For many PC worlds I find ~30 texels per unit to be a sufficient compromise between quality and size.

Have your lights set to cast soft shadows.

To actually see how the lightmap texels will be projected onto your geometry, you can click the dropdown at the top of the Scene window (the one that is usually set to "Shaded") and set it to "Baked Lightmap".

To change the lightmap UVs of your mesh, you can do so in the model settings (click on the model in the Project window and change the Generate Lightmap UVs settings at the bottom of the Model tab).

While some objects with short shadows may be more accurately lit with hard shadows, soft shadows are more accurate across longer distances and look better at lower texel resolutions. Unity doesn't include adaptive shadows out of the box like PCSS (Percentage Closer Soft Shadows).

Next time you go in the sunlight, look at your shadow. Look at how the edges are sharp close to the origin (your feet) and get blurrier as you go up! That's what techniques like PCSS emulate.

Do not overuse texels! The more texels you have, the larger in file size your lightmap(s) will be!

<https://unity3d.com/how-to/progressive-lightmapper-optimization-tips>

There are some situations where you would want to have a higher texel count because of overlapping lightmapping UVs (common with probuilder objects). [UV overlap](#) [docs.unity3d.com]

In your level design, avoid putting detailed shadows on the floor or low on the walls if your lighting is baked; the non-static models walking on top of those shadows won't have those detailed shadows projected onto them.

Lightmap resolution

Controls the maximum resolution of the lightmap(s). Set it so that you only have one lightmap for optimal performance.

Bounces

Controls how many bounces a light ray will perform. [These take a long time to bake]

How many light bounces your scene will "need" depends on what type of scene it is, how many lights there are, how bright (or dim) it is, and how many shaded crevices need to be filled in with global illumination.

- Outdoor scenes can get away with less (~2).
- Indoor scenes may warrant more light bounces (3-5).

Bakery Lightmapper

[Link to Bakery on the Unity Asset Store](#)

A very powerful (paid) alternative to Unity's built-in progressive lightmapper. Very highly recommended after you've made a couple worlds and are serious about making more with even better lighting. Watch this video for details.

<https://www.youtube.com/embed/toMvetpwOHU>

Read [the manual](#) to learn how to use it. I also made a [video tutorial](#) on my Patreon that explains how to use it with VRChat.

You need to put a Bakery light component on all the lights in your scene that you want baked.

The [real time preview](#) is also very highly recommended as it allows you to preview your changes in real time (rather than having to wait minutes to see each change). This time savings not only speeds up your creation, it allows you to be more creative and experimental, and try things you otherwise wouldn't have.

Note that this add-on will only work with NVidia graphics cards.

Light Volumes

Preview_0.png

This is a package by RED_SIM that essentially acts as 3D lightmaps. It is extra work to set up but is feasible for small to medium sized scenes (or small sections of larger worlds).

To learn how to use, read the [official How to Use guide](#) or watch my [Patreon video](#) that explains how to set it up and how everything works.

This is NOT a light probe replacement! Only objects with light volume compatible shaders will be lit by light volumes! Light probes are still needed in your scene (and light volumes includes a tool to make them easier to set up too)!

Bakery is NOT a requirement to use light volumes. They can be baked with Unity's built-in lightmapper.

Baked Volumetric Fog

Uses the data from VRC Light Volumes to make raymarched volumetric fog. The setup is simple but there are lots of options in the shader that are covered in [this video I made](#).

If you implement it in a world, make sure to have an option to turn it off because it is performance heavy!

Example screenshots from a world I uploaded, [Chinatown at Night](#).





Sunshine Flare

This includes not just a lens flare but also godrays that come from a directional light.

The intensities of both can be adjusted individually (I think their default values are far too intense). I prefer a more subtle flare and godray effect of around 0.1 for both instead of at the default intensity of 1.

1cd367dd-b4a7-4066-9380-e3e99891a410.png

LTCGI

LTCGI is a package made by pi for realtime area lights. These lights can also be animated, textures, or even animated textures (like video screens, which is also the most common use case). If the LTCGI light is static, it can also cast shadows on baked static objects.

The biggest limitation is that these lights can only effect objects that have an LTCGI compatible shader.

demoapp.jpg

vketch_booth-9708c86187c17f66cb1d0e5bc0704279.jpg

For more information, please read the [LTCGI documentation](#).

Combining real-time and baked lighting

Real time lights do not calculate indirect lighting in Unity by default. Sometimes, environments can be so big that baking all of the direct shadows (like from a directional light) is not feasible. In this case, it can be acceptable to have a directional light which casts realtime shadows. In addition, the indirect lighting, aka global illumination or GI, can be baked. This will result in sharp shadows across the landscape and beautiful baked lighting in the places that need it.

To do this, set the directional light from Baked to Mixed. This combines realtime shadows with baked GI.

Make sure to do all the other steps like setting baked objects to static (at least "contribute GI").

Adjusting per-object texel density is very important! Especially for this use case in particular!

Prefabs

<https://www.youtube.com/embed/r9l0jaRbL4o>

Prefabs allow for an object and all its components to be saved into a single asset. These are especially useful for things like light fixtures or pickupables! Combine a light source and a 3D model into a prefab! Combine a pickupable, its vrc_pickup script and its collider(s) into a prefab!

Check out the [Udon prefabs section](#) [docs.google.com] of VRCPrefabs. I also recommend [Vowgan's Collision Sounds](#) [vowgan.booth.pm] , they are incredibly satisfying.

Day/Night Cycle

This is a fork (edit) I made of an Udon system originally made by Nova_Max. It has the ability to make the directional light (sun) rotate around the world, along with a moon, stars, and changing cubemaps depending on the time of day.

Physics

<https://www.youtube.com/embed/dfjIVe0XXTU>

Pickupable objects require 3 components:

- Rigid body: allows object to interact with physics [called a rigid body because there is no deformation]
- Collider: allows the collider to collide with other colliders (so the object will not simply pass through everything)
- VRCPickup: allows the object to be grabbed in VRChat

To make a VRChat object pickupable, add the VRCPickup component to the GameObject. This will automatically add a rigidbody (if there isn't one already).

The collider types [in order of lowest to highest performance cost]:

- Sphere [1 vector]
- Capsule [2 vectors and the distance between them]
- Box [a volume]

A mesh collider will form its collider to the mesh. If applying to an object with a rigidbody, the mesh collider should be set to convex so it can collide with other objects with rigidbodies that have convex mesh colliders.

A convex collider will also limit the collision mesh to 256 triangles.

In addition, the naming convex relates to convexity; the collider is not concave; for example, any openings, like on a bowl or a cup would be covered with a convex collider.

When creating colliders for objects with a rigidbody, it is best to use a primitive or a combination of them for a result that is both performance and believable.

The orthographic view mode (where everything is of equal distance to the camera) is extremely helpful for editing colliders. To enable it, click on the box in the top right corner of the scene view. For example, on a beer bottle, I would use a capsule collider around the part that can roll, and a box collider inside the capsule that goes to the top and bottom of the bottle so it can stand up.

For optimizing pickups, make sure that all of them are under a single gameobject at the root of your hierarchy. Every object with a rigidbody not only needs to update *its* transform every frame, but also the transform of all of its parent gameobjects. Ensuring that all the pickups are under a single parent keeps your hierarchy organized while also not wasting performance.

For even more believability, you can apply physics materials to colliders; for example, if you have a carpet floor, you can apply a high friction physics material to it, and a low friction physics material to an ice sheet. Several physics materials come with the [Unity Standard Assets](https://assetstore.unity.com) [assetstore.unity.com] .

<https://www.youtube.com/embed/SuUNnswH94>

DO NOT USE MESH COLLIDERS FOR COMPLEX MESHES! (This is extremely taxing!)

It's best to add one or multiple primitives (box, sphere, or capsule) collider(s) to the object and make the colliders conform to the mesh that way.

In a 3D modeling program like Blender, you can create a set of simpler meshes in the same model, turn off their mesh renderer, and give the simpler meshes a mesh collider. This will save performance and most players won't notice the difference.

<https://www.youtube.com/embed/bh9ArKrPY8w>

<https://www.youtube.com/embed/Nm2TsnKVGi8>

Unity also uses different types of joints. These can be useful for creating things like physical doors.

<https://www.youtube.com/embed/f4xikqJdkwM>

Level Design

Have you ever pulled on a door handle, only when you should have pushed? Level design, like door design, is best when you never even notice it.

Through level design you can make people perform actions subliminally. It may sound obvious, but people won't do what they can't do.

As a level designer it's your job to form your levels in such a way that you get people to do what you want them to without them even realizing it.

For example, don't want people to congregate around a certain area (like the stairs)? Through design, you can make it far less likely that people will gather there (like by making it smaller and less inviting than the areas around it).

Through design you can greatly influence how players act.

Recommended videos:

<https://www.youtube.com/embed/RwlnCn2EB9o>

<https://www.youtube.com/embed/MMggqenxuZc>

<https://www.youtube.com/embed/NGsfEVDsSSI>

<https://www.youtube.com/embed/TX58AbJq-xo>

<https://www.youtube.com/embed/yY96hTb8WgI>

- design applies to everything we use, not just video games

One extremely important and often overlooked aspect of VR level design is changes in elevation. When walking around your playspace, the elevation should not constantly change. Even strafing in your playspace but moving up or down an incline in-game can be extremely disorienting.

In VR, changes in elevation should be

- Expected
- Deliberate
- Consistent
- Short
- Performed entirely with artificial locomotion (thumbstick or jumping, ideally just one and not both simultaneously for greatest comfort)

VRChat has a layer called Walkthrough; if you set an object to this layer, physics objects will still collide with it but players will pass through. I would recommend setting much of the furniture to this layer.

{If you're super interested in level design, you can read this [123-page document on the topic by industry experts](#) [docs.google.com] .}

Particles

Particles [docs.unity3d.com]

<https://www.youtube.com/embed/FEA1wTMJAR0>

An important note about particles, you may have seen that if you roll your headset in-game that particles will rotate with your head motions. Untick “Allow Roll” under the renderer tab if the option is present. This will fix the issue.

Post Processing

You can easily apply post processing to your scene by using the quick function in VRWorldToolkit.

Read [Silent's post](#) for more info.

Motion Blur, Depth of Field, and any Screen Space effects (reflections, ambient occlusion, etc.) should not be used in VR. If you want to use screen space reflections on your materials you can use Moochie's shaders, but SSR often breaks, especially in VR, and it has a huge performance cost, so if you do use it (which you almost always shouldn't) use it sparingly!!!!!!!!!!

Screen distortion/warping effects should never be used in VR. They are extremely nauseating.

Just like a real camera or your eyes, this can automatically adapt the exposure of the virtual camera(s) based on how much light it is looking at. Go into your post processing volume and under auto exposure the most important settings are minimum and maximum EV. If you hover over these settings, it will give you a brief explanation and if you need more information you can click on the circle with the question mark that is on all Unity components.

I (based on experience and no empirical evidence or research, so don't quote me on this) recommend setting the minimum EV to around -0.7 (at least for interiors), the maximum EV to 0.7 and the exposure compensation to 0.7. If you notice the exposure flickering too much or too fast you can set the minimum and maximum to the same value.

https://www.youtube.com/embed/_PzYAbPpK8k

Audio

Audio [docs.unity3d.com]

Create a new empty GameObject and add the component VRC Spatial Audio Source. This will automatically add an Audio Source component.

If it is world music, uncheck enable specialization.

To compress the audio, set the compression format to Vorbis, and adjust the slider lower for more compression. If the audio is spacialized, check "Force to Mono".

[VRC Spatial Audio Source Documentation](#)

[Audio-basics-crash-course](#) [ask.vrchat.com]

Udon

<https://www.youtube.com/embed/i-lAiYnBcOA>

Udon is how to make code in VRChat worlds. It is a visual scripting language, which means it uses blocks and nodes rather than traditional lines of code.

UdonSharp

If you already know C# and would rather use it to write your Udon scripts using lines of code, you can use UdonSharp to do so.

Note that UdonSharp behaviors are not MonoBehaviors. Not all methods in Unity's MonoBehaviors are exposed in Udon, which limits what you can code.

[VRChat Udon Tutorial Videos Playlist](#)

Many people view coding as a daunting black box. As a programmer myself, I can assure you that a basic understanding of it's principles is not as complex as some portray it. The best advice I can offer you is:

- Take it slow, don't try to learn everything at once
- Don't dabble or attempt to modify a piece of code until you know what it does
- Go in with a logical mindset

The last one is paramount. Think about the exact logical steps a computer would have to accomplish to achieve a desired result. Then putting it together will mostly be a process of translating that logic into computer code.

Oh, and the 2 most important tips:

READ THE CONSOLE!!! and

WRITE COMMENTS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

Unity runs on C#, so if you would like to start learning a programming language, C# would be a great place to start. As of writing Brackeys is creating a tutorial series on programming for Unity. [HOW TO PROGRAM in C# YT playlist](#)

Basics

Data types:

- Boolean - true or false
- Int - an integer
- Float - a decimal

- String - a collection of characters

Note that ints in C# are 32-bit, which means they have a limit of ~2.1 billion, positive or negative. Exceeding this limit will result in overflow, which will cause errors.

Alternatively you can use the community-developed UdonSharp. It will translate C# script into Udon Behavior. It is more reliable, and if you have basic coding experience I would recommend using it over Udon's visual script.

Final Things (Version control, optimizations)

Occlusion Culling [docs.unity3d.com]

Set your static GameObjects to static [this will set the sub-parameters, occlusion and occludee static to true]. Then go to Windows>Rendering>Occlusion Culling, and hit bake before uploading your world.

Unity won't render avatars the player can't see; since rendering avatars is what tanks performance the most, good VRChat level design should subconsciously guide the player to look only at the other players they are interacting with, without rendering a bunch of poorly performing avatars in the background that they don't care about.

Version Control

If you are putting a lot of time and effort into a world, version control is a must (lest you end up like me and lose your work). Version control tracks the changes you make to your project and allows you to revert your project to previous states (indispensable if something goes wrong).

Working on a world, you are most likely going to be one person on a single computer, so I would simply recommend using [Sourcetree](https://www.sourcetreeapp.com) [www.sourcetreeapp.com] .

<https://www.youtube.com/embed/TbdWXbRIPo0>

<https://www.youtube.com/embed/16clYZU0Soo>

Optimization (Review)

World Creation, Optimization, and Community Labs Tips [docs.vrchat.com]

Quest Content Optimization [docs.vrchat.com]

- Set everything that is static to static [will allow for lighting and occlusion culling to be baked]
- Set all lights to baked
- bake the lighting
- Bake occlusion culling

Remember, occlusion culling only affects the *visual* rendering of objects! Objects out of view will still be processed and affect performance!

Make sure to include toggles on performance heavy gameobjects, for example:

- toggles for physics objects
- toggles to turn off shadows for realtime lights (if you have a realtime light with shadows enabled in your scene)
- MIRRORS!!! This should be obvious but they should have toggles and be off by default!

Use the Unity profiler! It will help you find things in your world that are causing performance to drop the most! This is especially useful for figuring out how to optimize scripts!

<https://www.youtube.com/embed/sBpXij9G3OY>

World Work in Blender

(Advanced Users)

Level Design in Blender

You can greybox your basic level design and shapes in Unity using Probuilder, and using Probuilder's export feature or the FBX exporter (download in package manager) you can export your assets for further modification and detailing, with the scale intact.

[See Maebbie's guide for a round-trip workflow](#)

<https://www.youtube.com/embed/1sPXOYgrZdE?list=PLHPI0SFKkUjNIZs5K9BV4pgXiLrQpEu2D>

You could also make interiors completely in Blender using ArchViz add-ons, like Archimesh!

Custom bevel profiles will be extremely useful for detailing (from chairs to stairs, baseboards and columns, etc.).

<https://www.youtube.com/embed/D2MwHYxc2uM>

You can move around the viewport in fly/walk mode in Blender using SHIFT+`, similar to the scene navigation in Unity.

If you want to create a scene in Blender to match a photo (REFERENCE PHOTOS ARE SUPER IMPORTANT!!), using the free fSpy is fantastic to do the first thing, which is lining up the 3D camera!

Blender now has support for VR scene viewing, with support for Oculus and Windows MR. You can view it with a Vive or Index by opting into the SteamVR beta. (Blender uses OpenXR, a standard SteamVR only just started supporting in its beta.)

Editing Assets in Blender

Topology is important; if you have bad topology on your mesh, it will be unnecessarily difficult to work with.

<https://www.youtube.com/embed/VudCtEtNXbl>

Plenty of meshes you find online will have horrible, unworkable topology.

There are programs/addons to remesh, including (surprisingly good) free ones:

<https://www.youtube.com/embed/l88hQj16lhA>

Modeling your own assets in Blender

Unlike avatars, the assets you will model for a world use a different workflow, known as hard-surface modelling.

(SUPER IMPORTANT) learn the difference between destructive and non-destructive modelling:

<https://www.youtube.com/embed/0SbyO33YtL4>

<https://www.youtube.com/embed/LdQwmfPyuw0>

- just watch this one, you don't have to follow along. It uses paid add-ons called HardOps and BoxCutter but they are not necessary to understand the concepts and techniques taught in the video.

Hard Surface Modeling in Blender - YT playlist, requires familiarity with Blender

Bake normal maps for high-poly meshes in Blender, then lower the poly count (advanced)

<https://www.youtube.com/embed/0LuEqvO9QsE>