

The VRChat Creation Compendium

Concisely explains every aspect of Avatar and World creation for VRChat. Additionally teaches all one needs to know about computer graphics and game design to get started, assuming no prior knowledge.

This guide is not intended to be an end-all-be-all resource but a concise introduction to the many different paths that VRC Content Creation can take you.

Originally posted on Steam Guides:

<https://steamcommunity.com/sharedfiles/filedetails/?id=2190684978>

- [Preface](#)
- [Gathering Tools](#)
- [Fundamental Knowledge](#)
 - [Meshes](#)
 - [Materials](#)
 - [Optimization](#)
 - [Important Notes](#)
 - [Blender](#)
- [Avatars](#)
 - [Basic Upload](#)
 - [Materials and Shaders for Avatars](#)
 - [Custom Avatar Workflow](#)

- [Custom Avatar Creation \(Unity\)](#)
- [Avatar Optimization](#)
- [Custom Avatar Creation \(Blender\)](#)

- [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 \(Advanced Users\)](#)

- [Conclusion](#)

Preface

Anyone who's tried to learn creating content for VRChat understands the lack of collected, concise learning materials. Tutorials are scattered, with many being barebones, outdated or go against best practices (including VRChat's official documentation). The goal of this compendium is to congregate up-to-date tutorials and documentation, along with distilled knowledge to help both those who want to start creating and those who already are. Especially for VRChat (a VR application that must render in real-time), learning the techniques to make content that both looks appealing and performs well is essential.

Text, sections and video links enclosed in this blue info box are optional reading; they will typically explain why something functions the way it does.

Topic headers will have links to the official documentation (provided they exist). You do not need to read these but I would recommend doing so if you are stuck or are curious about a certain topic.

In Unity, you can click the circle with the ? (in Unity 2019+) on any component and it will take you to the respective documentation.

Gathering Tools

To upload custom content to the game, you must install a specific version of Unity, the VRChat SDK, and use a VRChat account (not Steam or Oculus) of at least New User (Blue) ranking.

Installing Unity

Follow this official tutorial by VRChat to get started by installing the Creator Companion (used for managing projects) and Unity.

<https://www.youtube.com/embed/0u1g0TYoJsU?si=IIFWB-vuZqnxN9ji>

You will want to install the Unity Hub. Download [here](#).

Later on, more advanced users may also want [Blender](#), a free 3D modeling software. You can download it from its site.

Unity Interface

Create a new project in the VRChat Creator Companion.

The video shows the Unity interface, helpful for creating both avatars and worlds.

Note that every GameObject contains a Transform component; the transform of an object defines its position, rotation and scale on the X, Y, and Z axes.

TL;DR

- Q or middle mouse drags
- W adjusts position
- E adjusts rotation
- R adjusts scale

Tips

- Don't forget to switch between center vs pivot and local vs global!
- Hold alt while left clicking on a sub-hierarchy arrow to completely expand/contract it!
- Press F to focus on a selected object; hold F while dragging to follow.
- Learn hotkeys to speed up your workflow!
- If you think you are doing something inefficiently, there is probably a faster and easier way to do it!

Fundamental Knowledge

Applicable to both avatars and worlds. Necessary for those who want to become more professional, but the more advanced stuff is not entirely necessary to know for the most basic edits.

Meshes

The geometry of the models are called a mesh, and are made of polygons, or 2D shapes: triangles (3 sides), quads (4) and n-gons (5+, typically undesirable).

These polygons are made of vertices (the points), edges, and faces.

“Originally posted by **ck12.org**:



Topology

The makeup of those polygons on a mesh is called the topology. Good topology is essential to working on models effectively.

Normals

Every polygon has a normal vector, perpendicular to its coplanar 2D plane. This helps determine what parts of the mesh are "inside" and "outside".

By default Unity does not render the back faces of polygons, called backface culling.

If you can only see the back faces of your avatar (or a part of it), chances are the normals are inverted. This will be especially noticeable; it happens to a lot of avatars when custom shaders are disabled.

Materials

The following section on materials will cover what is needed to know for both avatars and worlds.

Specifics for avatars and worlds will be covered in their respective sections.

Color or detail is applied to the mesh by using a material. It combines a shader and textures.

If a part of your model is a shadeless magenta, something is wrong with the material slot (typically a referenced shader is missing) or there is no material applied to that part of the model.

To create a material in Unity, right click in the project window>New>Material. The default shader is the Standard shader, which is a physically based, or PBR shader.

Shader

Component of a material which determines how it interacts with light. The 2 types of shaders you will work with in VRChat most often are [PBR shaders](#) for photorealism, and toon shaders for a toon or anime look.

Watch the video below as it will help you understand the textures section better.

https://www.youtube.com/embed/_LaVvGikBDs?start=57

TL;DR everything reflects and absorbs some light.

[Textures/Maps](#)

The 2D images a material combines which are projected onto the triangles.

Projected onto those triangles are textures (AKA maps), which are 2D images. Unity utilizes several different types of textures, combined into a material, which you can then apply to a 3D model.

Texels (texture pixels)

Texture types:

Albedo (AKA basecolor) texture

The color of the surface, but with all the shading information removed.

Normal map

A mostly blue-purple image, which fakes 3D by altering the angle (the normal vector) at which light reflects off of the surface (at a specific texel).

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



A stone wall texture and its corresponding normal map texture

Occlusion (AKA Ambient Occlusion, AO) map

Puts indirect shading information into the material, simulating parts where it would be harder for light to escape. Separating the occlusion map from the albedo map allows for more intricate shading according to the lights in the scene.

Smoothness map

Controls how clear the reflections in a material are.

Unity has an idiosyncratic way of handling these, and because the industry standard is roughness maps, not smoothness maps, I would recommend using an alternative more feature packed PBR shader such as:

- [ORL's Shaders](#)
- [Silent Filamented](#)
- [Mochie standard](#)
- [z3y's shader](#)

These shaders offer many more options than Unity's standard shader and will increase the visual fidelity of your avatar or (especially) world. Further notes on each in their respective sections.

DON'T GO OVERBOARD WITH THE SMOOTHNESS! If a material isn't shiny (like unpolished wood or brick), don't make it overly shiny! It will just look wet or wrong!

[Metallic map](#)

Defines which parts of the material are metallic.

[Specular Map](#)

For use with the alternative specular workflow. I would only recommend this workflow to beginners if your material comes with a specular map.

[Emission map](#)

When activated, the emission property of the material is used to define the intensity and color of light it emits. The emission map is used to define how the emission is different across the material.

Diffuse map

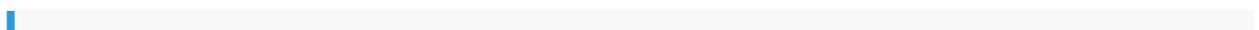
A base color map with balanced shading (no clearly directional light source). This is the texture you would get if you simply took a picture of a surface. If a model only has one texture then it would be considered a diffuse texture (you will find this a lot on toon shaded models).

This texture has fallen out of favor in the PBR workflow for the albedo and ambient occlusion texture combination, which allows for more variable shading of a material and better compression when using multiple albedo maps (like for different skins).

You can read a more in-depth guide on PBR shading written by Silent [here](#).

UV mapping

Specifies how the 2D textures should be projected onto the faces.



Originally posted by **Wikipedia**:



The best way to think of it is that your mesh is made of paper, and you have to decide where you want the cuts in order to lay it flat.

UV stands for the vertical and horizontal axes U & V, since W,X,Y & Z are already used.

Unity Texture settings:

Types

- Default- all the textures on a material except the normal map(s)
- Normal- normal maps must be set to this in order to function properly
- Sprite- if you want to have a 2d image displayed in 3D space (ie use it for a picture hanging on a wall)

SRGB

Should be disabled on grayscale textures (e.g. height, AO, metallic) or else they won't function properly!!!

Mip Map Filtering

If the texture looks too blurry from a distance, try changing the filtering algorithm from Box to Kaiser.

[MipMapping Explained](#)

Filter Mode

Set to Trilinear for the best result.

Aniso Level

Makes textures not appear blurry when viewed from angles. I would recommend setting it to 16.

In Unity the setting is a slider but only works in powers of 2. If it is set to a non power of 2, it will go down to the next lowest power of 2 (e.g. 10 would go down to 8).

Texture compression settings

Max resolution

The maximum resolution the texture renders at.

For world textures I typically wouldn't recommend setting it higher than 1K (1024x1024).

Crunch compression

A lossy compression algorithm. I would recommend setting it to around 20 for the best compromise between file size and quality.

Optimization

Optimization is incredibly important for VRChat. Being in VR, it should run at a high frame rate so the users can play the game comfortably. I've experienced many times when my friends or I had to quit because we were feeling too nauseous.

Material Slots

Each material slot incurs a draw call on the CPU. Ideally on an avatar or 3D model you will have 1 material slot per each mesh that requires a different shader.

Lowering the number of draw calls is one of the most effective ways to increase performance.

Merging multiple material slots into 1 is done through texture atlasing. This can be automated in the CATS Blender Plugin.

Polygon/Triangle count

Keeping this number low without terribly sacrificing quality is the balancing act you will have to perform in developing content for this game. Reducing the number of polygons is done through decimation; in Blender this is mostly done through the decimate modifier.

Shaders

Many shaders come with fancy special effects that can tank performance. Transparencies and especially grab passes are ones to look out for and use sparingly!

[Setting Up For Quest Creation](#)

Important Notes

Organize your files! It sounds boring but it will save you a ton of headache in the long run!

I would recommend creating a dedicated folder for VRChat project assets, and splitting the assets up by first:

- Avatars
- Worlds

And then:

- Textures
- Models (if a texture goes with a specific model group the two in the same folder)
- Prefabs
- Shaders

I would recommend organizing your textures by type, similar to how they are categorized in texture websites like <https://ambientcg.com/>.

What's important is that the organization works for you; you can adjust it however you like.

Name your files descriptively! You should be able to know what a file contains without having to open it!

Hitting the Tab key in a text box jumps to the next text box!

Reset GameObject transform

On the transform component of a GameObject, click the (gear on the top right) and click reset. This will return the object to XYZ position & rotation to 0,0,0, and the XYZ scale to 1,1,1.

Alternatively you can just reset the position, rotation, or scale individually via the same dropdown.

Copy & Paste Components

- Click the gear on any component
- click copy
- click the gear of another component
- click "Paste as New"

If working in Blender, there are many addons (free & paid) that can speed up your workflow!

3rd most important tip: if you are creating something custom, **use reference photos!** You will not be able to create something in 3D from memory as quickly or as good as you would from a reference photo! There are many subtleties and details you forget!

2nd most important tip: **if there's something specific you're trying to do, don't try to figure it out yourself, look it up!** Speaking from experience, you will likely become frustrated and do it in a suboptimal way. There are many tutorials and answered questions that will help you immensely. If you can't find the answer you're looking for, post a question! It's almost certain other people have it too and that someone has the answer. Even experts ask questions.

And the #1 most important tip: if something goes wrong in Unity, **READ THE CONSOLE!!!**

Unity Tips

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

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

<https://www.youtube.com/embed/thA3zv0IoUM>

Blender

For those interested. Learning Blender is not always necessary. There are plenty of 3D assets, including free ones, that you can find on the [Unity Asset Store](#) or [Booth](#).

As experts like Blender Guru have said, it is generally more efficient not to make your own assets unless you need something really specific or unique. Time is money after all.

As for making world geometry, using the free Unity add-on Probuilder is often easier and quicker, especially for prototyping, as you will see if you choose to read the modeling section for worlds in this compendium.\

Probuilder has its limitations, particularly in making more complex geometry and shapes (which you should not attempt at first).

If you want to make more complex geometry for a world, I would recommend making a simplified version in Probuilder first so you can enter VR sooner and get the scale right, then export that into Blender and make finer adjustments there.

<https://www.youtube.com/embed/e5oWR6mwIY4>

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

<https://www.youtube.com/embed/85Xu93bsN34>

https://www.youtube.com/embed/ebzGMqVg_h0

^^ WATCH THESE FIRST ^^

[Blender Basic Tutorials playlist](#) <--- Good for learning specific tools, workspaces or processes.

Round trip world workflow: <https://vrclibrary.com/wiki/books/maebbies-precise-solutions/page/blender-3-28x-mapping-config-for-mapping>

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

<https://www.youtube.com/embed/7DT0dtarYv4>

<https://www.youtube.com/embed/7pZVngqwgDo>

[What is a .blend1 file and why do you \(not\) need them?](#)

Avatars

Covers basic to advanced topics on avatar uploading and creation.

Basic Upload

Finding avatars

Avatars are commonly found on **digital asset marketplace sites** like booth.pm and Gumroad. Jinxxy is another alternative, more similar to the content on Gumroad. Avatown is an alternative that kind of sells everything.

[Booth](#): Anime girl aesthetics similar to those found in actual anime. Heavy Japanese creator presence and you have to pay in Japanese Yen (or through Paypal).

Since the site can be hard to navigate, [Boothplorer](#) helps a ton with finding avatars and compatible clothing.

[Gumroad](#): Heavier on western creators. Mainly furry and western anime-inspired humanoids with e-girl/e-boy aesthetics (and usually very unoptimized - sorry, just had to mention that).

[Jinxxy](#): Basically Gumroad but focused on VRChat exclusively.

[Avatown](#): Sells all genres. Also focused on VRChat exclusively.

Guide to free avatar models [\[gitlab.com\]](#)

Avatar base and accessory hunting [\[vrchat.fandom.com\]](#)

Uploading Avatars

<https://www.youtube.com/embed/bSwMz4WcajQ>

[Creating Your First Avatar](#) (Official Guide)

Below is a (DATED) alternative guide for importing FBX models (non-full packages) utilizing [Pumkin's Avatar Tools](#).

1. Create a new Avatar project in the VCC, and add Pumkin's Avatar Tools. New options will appear on the top row of Unity: VRCSDK>Show Control panel & Tools>Pumkin's Avatar Tools.
2. Don't forget to add shaders such as liltoon or poiyomi if your avatar needs toon shading.
3. Import your avatar's FBX file and all of its textures.
4. Right click in the Project window. Select Create>Material. In the Inspector window, change the shader to one like poiyomi (or VRChat>Mobile>Toon Lit or VRChat>Mobile>Standard Lite for Quest). Duplicate (Ctrl+D) this material enough times until you have a material for every texture.

OR

Click on your FBX. In the inspector, click the materials tab, and then click extract materials. Change the shader (if needed) and fill in the textures.

5. Click on an empty material, and drag a texture to it in the Inspector window. Do this for all your textures.
6. Click on the FBX file in the Project window. In the Inspector, click rig, and under the dropdown select humanoid (if the avatar is a humanoid) and click apply.
7. Under the Materials tab of your FBX, click and drag the materials into their material slots, then click apply.

8. Drag the FBX into the Scene window. Reset the transform by clicking  (the gear icon) and clicking Reset Transform. [This will center the avatar and set its scale to 1].

9. At the top bar, go to Pumkin>Tools>Avatar Tools. Click "Select from Scene" and then  "Quick Setup Avatar". Also click "Fill Visiemes" and "Fill Eye Bones".

10. At the top bar, open VRCSDK>Show Control Panel. Login, go to the builder tab, auto-fix all the red errors.
11. Fill in the info.

12. Click “Build and Publish for Windows”.

You can change your avatar's size/scale through Pumkin's tools!

If you scale without the tools don't forget to re-position the viewpoint!

Setting a solid color background is easy! Under the camera component, change Clear Flags to “Solid Color” and click the color box to pick a custom color!

You can also make interesting custom avatar PFPs through Pumkin’s Avatar Tools!

Upload an image as an avatar’s profile pic by importing an image, and in the Inspector change Texture Type from "Default" to "Sprite" and the camera from "Perspective" to "Orthographic".

(Uploading an avatar can be confusing for a beginner. I hope to make a concise video tutorial in the future. I will also explain common errors some users may encounter and how to overcome them. It will likely be aimed toward complete beginners, teaching only what you need to know for first uploading pre-made avatars in Unity.)

Troubleshooting

If your avatar has unshaded purple-pink sections:

1. Find the material (the file with the sphere thumbnail)
2. Change the shader (the dropdown near the top) to a different one(like Poiyomi, VRChat>Mobile>Standard Lite or VRChat>Mobile>Toon Lit

It is caused by Unity looking for a custom shader that isn’t there.

You could change it to your own custom shader or Unity’s standard, but VRChat’s shaders are simple, optimized, and look identical even when people have shaders blocked in their safety settings, so it’s best to use them first.

Later you can decide whether a more fully featured shader (Silent’s, Xiexie’s or Poiyomi) is worth using instead; of course, these shaders are more complex and require learning and toying with their features to maximize their potential.

BTW Silent’s Cel Shader is a rewrite of Cubed’s Cel Shader (which is depreciated).

You will probably want to unmap your jaw in your avatar’s rig. Click on the FBX, go to the rig tab, configure, head, select the bone assigned to the jaw and press delete, then click apply.

Ensure that the ARMATURE of your model (not your model) is set to an XYZ scale of 1,1,1.

Converting and uploading to Quest

[VRChat Cross Platform Setup](#)

My easy Quest conversion method:

1. Duplicate your avatar.
2. Press F2 and append "Quest" to the name in the hierarchy.
3. Import [EasyQuestSwitch](#), [VRCQuestTools](#) and [liINDMFMeshSimplifier](#)
4. Use VRC Quest Tools to make a Quest-ready version of the avatar.
5. Under the Upload tab of the VRCSDK window, "Switch Build Target to Android" (this will take a while).
6. Use Pumkin's Avatar Tools to remove incompatible components from the avatar (like Dynamic Bones and constraints). VRCQuestTools should already do this so this step usually isn't required.
7. Use the liINDMFMeshSimplifier component on meshes to reduce their poly count. Remember that this is Quest where people don't care as much about graphics so please be aggressive in the poly count reduction.
8. Open the VRCSDK window, fix any other errors and upload!
9. Switch back to Windows build in the VRCSDK window when done.

Video guide of how to make Quest *optimized* avatar:

<https://www.youtube.com/embed/8MkgXcidxg4>

Alternatively you can use the "Bake" settings in the Blender CATS plugin to get a quicker but inferior Quest Optimized avatar. Quest optimized avatars can be used as fallback avatars for non-Quest compatible avatars.

Materials and Shaders for Avatars

Default Shader

The standard shader is tailored to work on realistic hard surfaces, not features that are found on many character models such as [skin](#), [hair](#), eyes or cloth. If you are going for a photorealistic style, it might be best to seek out more specialized PBR shaders to suit your needs.

Quest Shaders

VRChat's shaders, like Standard Lite and Toon lit, are required for use on Quest Avatars. Outside of this use case they should generally be avoided since their feature set is vastly reduced for often minimal performance gain on PC (if not using lots of special effects on other shaders).

Toon Shaders

Two of the most popular and versatile toon shaders for avatars are [Poiyomi's](#) and [lilToon](#).

Poiyomi has many features and has an extensive [written](#) and [video documentation](#).

lilToon also has many features. It has documentation both in [English](#) and in [Japanese](#).

[Silent's](#) is another option. It also has many features and a [written documentation](#).

You can try them out for yourself and see which one you prefer.

Heightmaps

Heightmaps should generally be avoided on avatars. (They break at intense angles plus are more performance intensive. They are more ideal for certain use cases in environments, not typically on characters.) For more info on heightmaps, look at the Materials for Worlds page.

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 Poiyomi, or both Silent's [photorealistic SSS](#) and toon shaders.

Custom Avatar Workflow

VRoid

For those who don't want to deal with much of the nitty-gritty involved with most workflows of avatar creation, and instead would prefer a more cohesive, traditionally artistic approach, there is an option called VRoid.

What you do is create a model using [VRoid Studio](#) [vroid.com] and upload it via the [VRM Converter for VRChat](#) [booth.pm] , a Unity tool.

Simple editing

The simplest editing of avatars is often recoloring textures as you can simply use a 2D image manipulation program, like Photoshop, [GIMP](#), Krita or a web based one like [Pinetools](#). Just drag in the texture, edit it, and export it back to Unity. If you overwrite the previous texture then it will update automatically. Please create a texture with a new name first so that it is separate from the original and isn't ever overwritten if you re-import the package.

Frankensteining an avatar

It means combining various different parts, such as body bases, faces, hair, clothes, etc. into a single model with a single humanoid rig.

Typically these parts *USED TO BE* taken from MMD models (see the bottom of this page).

Now they are more commonly found on **digital asset marketplace sites** like booth.pm and Gumroad. Jinxy is another alternative, more similar to the content on Gumroad. Avatown is an alternative that kind of sells everything.

[Booth](#): Anime girl aesthetics similar to those found in actual anime. Heavy Japanese creator presence and you have to pay in Japanese Yen (or through Paypal).

Since the site can be hard to navigate, [Boothplorer](#) helps a ton with finding avatars and compatible clothing.

[Gumroad](#): Heavier on western creators. Mainly furry and western anime-inspired humanoids with e-girl/e-boy aesthetics (and usually very unoptimized - sorry, just had to mention that).

[Jinxy](#): Basically Gumroad but focused on VRChat exclusively.

[Avatown](#): Sells all genres. Also focused on VRChat exclusively.

Guide to free avatar models [gitlab.com]

Avatar base and accessory hunting [vrchat.fandom.com]

The process of frankensteining an avatar involves gathering all the parts you want and then using tools (either in Unity or Blender) to put all the parts together.

Some of the Unity tools include [VRCFury](#) and [Modular Avatar](#). Both have documentation and unofficial tutorials on YouTube.

Making Avatars in Blender

For making avatars through Blender, first install the [CATS plugin](#). Import the assets into the same .blend file via CATS (don't forget to click Fix Model!). Pose them to where they overlap the base model, and merge them via CATS through the Custom Model Creation tab.

Apply decimate modifiers to objects with a high poly count. Before atlasing your textures, save a new file (increment easily in Blender Save As menu by clicking the +).

When making avatars in Blender, DON'T APPLY DESTRUCTIVE CHANGES UNTIL YOU ABSOLUTELY HAVE TO!

A destructive change is one that cannot be easily reverted without Ctrl+Z, like clicking apply on a modifier, or atlasing your textures.

Before you make destructive changes, SAVE AS A NEW FILE!!!

You can easily increment files in Blender's Save As menu by clicking the +.

This workflow allows for your avatar to be easily customized later while also being able to keep it optimized when you export to Unity.

Additionally, you also have multiple save points you can return to in case you made an accident along the way, rather than having to start over.

If that fails, Blender also saves auto-backups (File>Recover>Auto Save...).

Version control (like Git) can be a lifesaver here if spending significant time on a single avatar. For more information, see the Version Control subsection under Worlds

“ Originally posted by Tupper:

What's an MMD model, and where do I get one?

“What is "MMD"?

MMDs are model files used by the program MikuMikuDance. This program is typically used to create music videos, animations, and etc. If you search for "MMD dance" on YouTube, you'll find tons of examples of these models.

They're a great way to import models because they come with a full rigging, full textures, and full UVMaps. All you have to do is set it up for VRChat, which is what I cover in my guide.

Unfortunately, if you don't want to be an anime character, MMDs typically are anime-styled. You might want to look into importing a Source Filmmaker/GMOD model, or look at Models Resource[www.models-resource.com] for a huge repository of models ripped from games.

Where do I get MMDs?

A lot of people say to look on DeviantArt for models, but most of those are "kitbashes"-- in other words, someone put together a bunch of parts to make an original character. These might look good and unique, but they're often extremely high polycount. That's why I recommend the following two sites for finding MMDs:

Nicovideo[3d.nicovideo.jp] - A great resource for extremely high-quality models. Lots of Kancolle.

Bowlroll[bowlroll.net] - Also a great resource, but a bit harder to navigate.

Both of these sites are in Japanese, so I'd suggest using Google Chrome for automatic translation of the pages. Also, both sites require a login-- I just use my Twitter login for Bowlroll, and I made an account for Nicovideo.

VERY IMPORTANT - Respect the rules

Artists put a ton of blood, sweat, and tears into making their models. Oftentimes, they have a license for use with the models. Ensure that you read the license, and obey it to the greatest extent that you possibly can.

Although you will not get banned from VRChat for using a model outside of the licensing terms, it is possible that the artist will issue a DMCA takedown request to VRChat to remove the model. In this case, the model will be removed from the servers, and VRChat will likely notify you.

Oftentimes, if you message the creator on social networking tools and explain what you plan on using the model for, they will be okay with it. You aren't making money off the model, so sometimes artists are happy to let you "cosplay" as their character.

As an example, the model I use in VRChat is [electrica2014](#)'s rendition of the Touhou character, Cirno. I messaged [electrica2014](#) on Twitter using Google Translate (they do not know English), and explained the use. They were happy to let me use the model, since I asked.

As far as the Touhou licensing goes, ZUN's licensing on Touhou is very open-- it allows for the use (and sale of) fanworks. The closest thing that VRChat comes to in licensing is "cosplay" (in my opinion), so if the license mentions that cosplay is okay, you're probably fine.

Finally, violating the licensing rules of MMDs has a secondary effect-- if it becomes prevalent, MMD artists will be less prone to releasing their models publicly in the future. Obviously, we don't want that-- so just ask!

Tupper's Tutorials - Creating Custom Avatars from MMDs
A Guide for VRChat

By: Tupper

This guide consists of a growing set of videos I've made on importing, optimizing, and setting up custom avatars from an MMD source.

(I copy-pasted this part rather than just linking to it because the rest of the guide is outdated)

Custom Avatar Creation (Unity)

Putting Things Together

1. Create a new avatar project in the VCC.
2. Typically you will start with a base from one of the sites mentioned in the previous page (like booth.pm or [Boothplorer](#)).
3. You can recolor an avatar by editing the textures in an image manipulation program like Photoshop, GIMP, Krita or online.
4. You may also get clothing and props. These can be added non-destructively with tools like VRCFury or Modular Avatar. Static meshes, like hand props, are typically easier to add than ones with armatures that need to be fitted to the base, like clothes.

Editing Textures

Find the image texture of what you would like to edit (e.g. the hair texture) and bring it into an image editor (like GIMP or Photoshop). If you want to edit the color, select what you want to edit and mess around with the color settings. If you completely want to change the color set it to grayscale first and then change the hue.

Physbones

Physbone components are used to add reactive bone chains to avatars; they can be used to make many types of things move, such as hair and clothing. They can also be interacted with, both by the wearer and by other users (if those settings are enabled).

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

PhysBone colliders

Allows for collision of PhysBones on your own avatar.

<https://www.youtube.com/embed/pT6wO01dwTU>

Contacts

<https://www.youtube.com/embed/LOZu6e8ozns>

<https://www.youtube.com/embed/Cp0ysawmizM>

Toggles

Toggles are objects that can be toggled on or off on your avatar in-game. Examples include articles of clothing or props.

The easy way to make toggles is to use either [VRCFury](#) or [Modular Avatar](#).

The hard, tedious, old-fashioned way to make toggles is covered in the video tutorial below:

https://www.youtube.com/embed/XqtSg6_W07Y

Puppet Menus

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

Custom Gestures/Expressions

The easy way is to do this through tools like [VRCFury](#).

The hard way is as follows:

Locate the Examples3 folder in the SDK in Unity. Click on vrc_AvatarV3HandsLayer.controller, duplicate it (Ctrl+D), then rename it descriptively. Create a new subfolder in your avatar's folder for animations, and place the new controller there.

In your avatar descriptor, under Playable Layers, click Customize then under FX Default Non-Transform. Drag the controller into the slot.

Duplicate your avatar in the scene. Open the Animation window (Ctrl+6). Create a new animation (e.g. Smile). Click record. Then click on the body in the hierarchy, and in the Skinned Mesh Renderer open Blendshapes. Drag them to make the expression you want (in this case Blendshapes like Blink Happy, a slight smile, and slightly raised eyebrows.)

When done, click the top diamond at frame 0 in the animator, Ctrl+C, then Ctrl+V at exactly frame 1.

In the Animator window, navigate to the hand movement (in this case Open), and set the animation file in the Motion slot in the Inspector window.

[AV3 Walkthrough - Part 3 - Weights, Blending, and Tracking - Avatars 3.0](#)[\[ask.vrchat.com\]](#) (Internet Archive)

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

<https://www.youtube.com/embed/PXyXF9nH--Q>

For controllers that have individual finger tracking (like Valve Index controllers):

<https://www.youtube.com/embed/2DFT0LsOalw>

Avatar Audio

Put multiple sound effects on avatars in an optimized way by using [Animator Play Audio](#).

<https://www.youtube.com/embed/BmXCXP6UsQ8>

Audiolink on Avatars

<https://www.youtube.com/embed/Gp1woO9RzGY>

Voice Activated Effects

<https://www.youtube.com/embed/kgs5gz8BPtY>

Twist Bones

Twist bones come with some avatars; they are extra bones that help prevent certain bone rotations from looking weird or collapsing in on themselves. They have to be set up.

The tutorial is somewhat outdated; now instead of using Unity rotation constraints, use [VRChat rotation constraints](#) instead.

https://www.youtube.com/embed/_GmAupH5Co0

OSC

OSC stands for Open Sound Control, but that doesn't matter. What OSC does is allows parameters on your avatar to be connected to external applications or devices. OSC is used for impressive

things like heart rate monitors and face tracking.

[OSC Official Overview](#)

OSC Unofficial Overview video:

https://www.youtube.com/embed/OHjN_q6RqGY

Upgrade Old Projects

<https://www.youtube.com/embed/M9l0SVF0fjY>

Unity 2018 and SDK 2 projects (very old ones, pre-2021):

https://www.youtube.com/embed/C06_sDZCVY8

~~Dynamic bones~~ (no longer supported, replaced with PhysBones)

Dynamic bones are from a [paid Unity add-on](#) (superseded by PhysBones, which basically do the same thing and more) that allows for bones to delay their return to their neutral position using physics. It is used for things like hair or animal ears.

If your avatar package used dynamic bones, you do NOT need to buy and import the Dynamic Bones package. You just need to download the [Dynamic Bones stub](#) so VRChat can automatically convert the dynamic bones components to PhysBone components, and you can modify those from there.

(DATED, Avatars 2.0) Custom Animations

Avatars 2.0 is now deprecated. Do not use it for a new project.

To add these, find the CutsomOverrideEmpty file in the VRCSDK. Make a duplicate of it (ctrl+D) and rename it (something like CustomOverride) (press F2).

Click on it and the animation slots will appear in the inspector window with nothing in them. If empty, the animation will be the VRChat default; if filled with a .anim file, it will be overwritten and that animation will be used instead.

To apply the animations to the avatar, click on the avatar and drag the CustomOverride onto the Custom Standing Anims and Custom Sitting Anims slots in the avatar animations component in the inspector window.

Avatar Optimization

Easy Optimization in Unity

Watch these two videos, and install the following two add-ons:

[d4rkAvatarOptimizer](#) - effortlessly and non-destructively optimizes avatars in Unity on upload. Sizable performance gain for essentially no extra work.

[Avatar Performance Tools](#) - Makes it easy to see what is taking up the most VRAM in an avatar, and thus what textures should be downscaled.

<https://www.youtube.com/embed/v7UVyMMP1ww>

<https://www.youtube.com/embed/8F9K1QEPF6A>

Duinrahaic's Avatar Optimization Thread

Don'ts of Avatars

Don't put automatically playing audio on your avatar (like music that plays when it loads in). That audio will also replay any time a player reloads avatars, like every time they change their safety settings. It's one of the most obnoxious components you can put on an avatar, it's completely unsolicited, and everyone in the world hears it at an equal unspatialized volume, so please don't use it.

Don't put real-time lights on your avatar.

Avoid gratuitous particle effects; they are extremely obnoxious, tank the framerate, and are often nausea, epilepsy, and anxiety inducing.

DO NOT PUT FULL SCREEN WARPING EFFECTS ON YOUR AVATAR. This is the single worst component type you can apply to an avatar, as it is usually unsolicited and the effect is literally sickening, as in nausea, epilepsy, and anxiety inducing.

Avatar Optimization Tips

Use [D4rksAvatarOptimizer!](#)

[Read the limits](#)

[Read the documentation](#)

Remove unused objects by setting them to the EditorOnly tag!

Decimate meshes using [liNDMFMeshSimplifier!](#)

Use the [Blender CATS Plugin](#) (or it's [unofficial updated version](#) that works with newer versions of Blender)!

[Pumkin's Avatar Tools](#) can make it super easy to remove unnecessary stuff from your avatar!

Avatar download size

1. On each texture file (image, typically .jpg or .png), in the inspector window, under compression, select normal quality (if it isn't already)
2. Enable crunch compression
3. Click apply

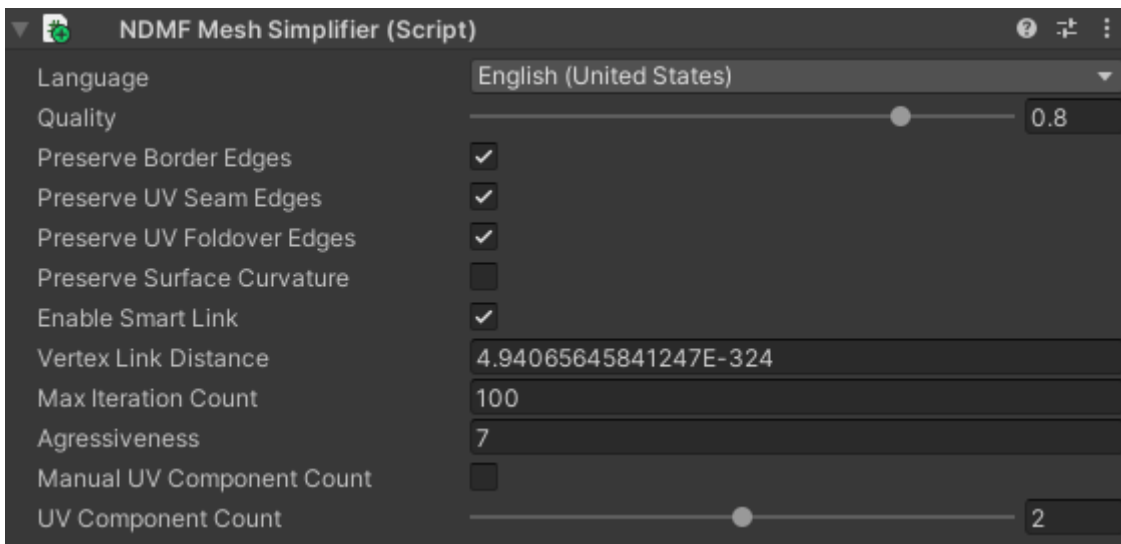
(You can do this with multiple textures selected to expedite the process)

Polygon count

The total number of triangles on your model.

In Unity, the simplest way to reduce this is to set toggleable objects that you aren't going to use to the EditorOnly tag at the top of the inspector tab. This means that said object and all its components will not be uploaded.

Another way to reduce polygon count is via decimation. In Unity, install the liltoon packages to the VCC, and then add the liNDMFMeshSimplifier to your project. Then add the NDMF Mesh Simplifier component to each mesh that you want to reduce the polygon count of. Place your camera at a distance, then adjust the slider and toggle between the first 4 checkboxes to find the best result. Remember that most people will not be viewing your model up close!



In Blender, you can either use quick decimation via [CATS](#) (or it's [unofficial updated version](#) that works with newer versions of Blender).

Or (for better results) you can use CATS to separate the model (like by loose parts) and apply decimate modifiers to those parts individually.

Here is a [guide of how to use liiNDMFMeshSimplifier](#) (in Japanese but can be auto-translated). It includes pictures.

1728818443-HQINtKuCyvOZm5f6w3LP8bT1.png

An image from the guide linked above that shows different quality settings and the corresponding triangle counts.

Material Count

The easiest way to reduce this is by having [d4rkAvatarOptimizer](#) do it for you in Unity. Just add the d4rk avatar component to the root of your avatar and adjust the settings, and it will tell you how many materials it will reduce.

The following instructions using CATS are if you are working with your model in Blender.

Combine identical materials via CATS Optimization dropdown, Material tab.

Create atlases of different materials. This will merge multiple textures/materials into one (reducing draw calls & increasing performance).

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

Bone count

The fewer the better. You can reduce the number of bones by merging their weights into their parents. Many MMD models come with far too many bones on parts like clothing than is optimal for VRChat. CATS has a bone merging section.

PhysBone component

This is put on the parent bone of the to-be-affected child bones. The fewer the better. For example, if you want PhysBones on your hair, you can put a single dynamic bone component on the head bone (and exclude all the non hair bones) instead of putting a separate dynamic bone component on each chain of hair bones.

PhysBones collider

The colliders can be easily removed though [Pumkin's Avatar Tools](#).

Avatar Audio

Put multiple sound effects on avatars in an optimized way by using [Animator Play Audio](#).

<https://www.youtube.com/embed/BmXCXP6UsQ8>

Quest Optimization

<https://www.youtube.com/embed/8MkgXcidxg4>

You can view all of the restrictions [here](#).

[Quest Content Optimization](#)

The Sins of Avatar Stats (review)

-

Absurd material counts (USE [d4rkAvatarOptimizer](#), or MERGE SAME MATERIALS IN CATS, COMBINE DIFFERENT MATERIALS USING THE MATCOMBINER IN CATS)

- **Unassigned PhysBones collision checks**
- **Gratuitous PhysBones colliders / collision checks**
- **Gratuitous PhysBones components**
- **Gratuitous** PhysBone count (MERGE CHAINS VIA CATS!!!)
- **Seizure-inducing (or other obnoxious and unsolicited) effects**
- **Absurd polygon count**
- **Absurd particle count**
- **Realtime lights** (especially per pixel and shadow casting lights)

Custom Avatar Creation (Blender)

CATS

CATS is the Swiss Army knife of VRChat avatar creation in Blender, especially when converting non-VRChat avatars into ones that are VRChat compatible. It also makes it easy to fix a plethora of problems that come with 3D models.

[CATS Documentation \(learn CATS here\)](#)

Lip sync

To automatically generate lip sync via CATS, go to the Visemes dropdown. You must have at least 3 mouth shape keys to start: AA, OH, & CH. Once you have those open CATS, open the Visemes menu, select the shape keys and click “Create Visemes”. [This will create the 15 visemes necessary for lipsync in VRChat.]

If the 3 slots don’t automatically fill, try translating your model via the Model Options dropdown. If the shape keys aren’t already in your model you must create them using Blender.

In Unity, fill the visemes via Pumkin’s Tools.

(Automatic) Eye Tracking

Allows an avatar’s eyes to dart around and focus on objects of interest. Created through CATS. The tooltip popups will guide you.

Applied in Unity first through Pumkin’s Tools, then set the eye rotations in the Descriptor.

Deleting unseen geometry

If you have clothes on top of your base mesh, delete the polygons underneath that should be covered! This will not only increase performance, it will prevent clipping issues!

Select the base mesh, go into wireframe mode and select the faces to delete. Then navigate to the

Object Data Properties tab (inverted green triangle), create a new vertex group, and with nothing else but the faces you want to delete selected, assign them. Now rename the group descriptively and deselect.

Before you delete these faces, save a new file first!

Rigging

<https://www.youtube.com/embed/Np3StFeVmLk>

Many models and bases will likely come pre-rigged. If you rig your own model it must conform to the [VRChat Rig Requirements](https://docs.vrchat.com/docs/vrchat-rig-requirements)[docs.vrchat.com].

If using a humanoid avatar ensure that in the model import options it is set to humanoid.

If using a non-humanoid avatar, use the generic rig [and it is ideal that you use your own animation controller].

Rigging can be done in Blender by using rigify.

<https://www.youtube.com/embed/-gIL6VZ-bkE>

<https://www.youtube.com/embed/f2pTkW-1JkE>

Rigging can also be done quickly and easily through the website mixamo.com.

[Quick Start - Mixamo Avatar Creation](https://vrchat.fandom.com/wiki/Quick_Start_-_Mixamo_Avatar_Creation)[vrchat.fandom.com]

Weight Painting

The way the bones are mapped to the mesh is through weights. Bone weights can be painted onto the mesh via weight painting.

Weight painting is hard. There is no perfect weight painting. You will suffer when weight painting. It's OK and a necessary evil that every 3D artist must endure if they want to properly rig their characters.

<https://www.youtube.com/embed/dKZrzG5r13g>

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

Clothing

Clothing models can be searched for on the sites mentioned in the previous page (like booth.pm or [Boothplorer](#)).

You would then parent the clothing using VRCFury or Modular Avatar in Unity or CATS in Blender.

The quickest and easiest method to create custom clothing in Blender is to select all the faces of your avatar that you want there to be clothing, marking seams, UV unwrapping, adding a texture and adding a solidify modifier to make the clothing look like it has depth and isn't shrink wrapped to the avatar.

Additionally you can sculpt the clothing mesh to your liking. For simple edits you can just use the Grab brush (G on keyboard).

You can also sculpt using Blender's physics based cloth brushes and bake the extra detail into a normal map.

Clothing can more realistically be created in Blender by "stitching" 2D planes with cloth physics applied to them.

https://youtu.be/HPz5gk_AT6w

https://youtu.be/7ULae_1GVpc

I would not recommend this method as it is tedious.

If you are especially interested in making 3D clothing, you can either buy the Garment Tool for Blender by bartoszstyperek, or custom model your clothes. Custom modelling your clothes for your specific model will usually make the rigging much easier and more accurate, plus it will usually result in a lower polygon count. Unfortunately, Marvelous Designer no longer sells a perpetual license, so I can't recommend it even to people who really want to get into designing 3D clothes.

Hair

For an anime-styled 3D character, you're going to want geometry-based hair (as opposed to particle-based). There are different ways to model hair depending on what you want your final result to look like.

<https://www.youtube.com/embed/BqWYgrXw7Jk>

<https://www.youtube.com/embed/AcjGeo1x7rA>

Texture Painting

The process of painting textures onto models. For non-PBR texture painting (just a diffuse texture) Blender works fine. Pre-made image textures can be edited through GIMP or Photoshop.

Substance Painter is by far the most robust PBR-texture painting application but also the most expensive. Alternatives include ArmorPaint (€16 for the precompiled version) and the free Quixel Mixer.

<https://www.youtube.com/embed/FeJFg5ullrc>

Again if you are just painting a diffuse texture just use Blender.

Full-Body Tracking

VRChat advises against using rig hacks because they may break in future updates.

<https://www.youtube.com/embed/2sfTEBAI8sA>

Complete avatar creation (NOT FOR BEGINNERS!!!)

Are you masochistic enough to make every part of an avatar yourself?

Rainhet's avatar courses seem to be the most streamlined and simplest for creating an avatar from scratch in Blender.

There is a [step-by-step playlist](#) and an [abridged playlist](#).

Worlds

Worlds

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://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.

Reflection Probe Guide [\[gitlab.com\]](#)

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 Cloteness 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

vkct_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](#) [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]}

Worlds

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

Worlds

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>

Conclusion

Encouragement message

You've made it this far, don't give up! Stay curious and never stop learning new things!

The best way to improve your art is through practice and taking all feedback into account, learning what you should keep doing and what you should avoid!

Creation can be incredibly satisfying, but not when you don't know what you're doing! Keep taking in educational and informational resources!

More learning resources:

[\[docs.vrchat.com\]](https://docs.vrchat.com)

VRCLibrary

(Not VRChat specific)

[\[docs.unity3d.com\]](https://docs.unity3d.com)

Official Unity Tutorials (not HDRP, URP/LWRP, 2D, ECS nor DOTS)

- concise Unity tutorials
- Entertaining and concise Blender tutorials
 - In-depth game technical analysis
 - Game design analysis

Still can't find what you're looking for or need the answer to a question?

Exert a valiant effort in looking it up (first check official documentation)

Post to the [\[forum.unity.com\]](https://forum.unity.com) / [\[ask.vrchat.com\]](https://ask.vrchat.com)

My Video tutorials

I now have my own concise scripted video tutorials! Check out my YouTube channel!

If you want to make your own tutorials, I would recommend watching these videos first:

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

<https://www.youtube.com/embed/J7aTFzQKamA>

Why I Wrote This Compendium

In my experience learning how to create worlds and avatars for VRChat, I was unsatisfied with the lack of concise and up-to-date resources that follow best practices. In asking many creators on VRChat how they learned their skills, none gave me a satisfying, replicable method.

Many in VRChat I have spoken to wanted to create their own models, but had not a clue how to start. It seemed to me that what those people needed was a holistic guide, not segmented ones. That was my intention in writing this compendium.

Translation

If you can translate this compendium into another language, please do so and link to this English original.

Feedback

I believe that constructive criticism is the most useful type of feedback one can receive; it doesn't criticize you, it critiques your actions or your work, it tells you exactly what you're doing wrong and gives you ideas on how to improve. If you can comment on how to improve this guide I implore you to do so and may incorporate your suggestions and/or information into a revision. We're all learning together.

This compendium and the written content within is not affiliated with VRChat Inc.