## Contents

1 What can I do?  
1.1 Home ................................................................. 3  
1.2 Explore ............................................................... 4  
1.3 Create ................................................................. 5  
1.4 Script ................................................................. 15  
1.5 Host ................................................................. 18  
1.6 Sell ................................................................. 22  
1.7 Contribute ......................................................... 24  
1.8 Frequently Asked Questions .................................. 25  
1.9 Release Notes ..................................................... 254
High Fidelity is an open-source software where you can create and share virtual reality (VR) experiences. You can create and host your own VR world, explore other worlds, meet and connect with other users, attend or host live VR events and much more.

The High Fidelity metaverse provides built-in social features, including avatar interactions, spatialized audio and interactive physics. Additionally, you have the ability to import any 3D object into your virtual environment. No matter where you go in High Fidelity, you will always be able to interact with your environment, engage with your friends, and listen to conversations just like you would in real life.
You have the power to shape your VR experience in High Fidelity.

- **EXPLORE** by hopping between domains in the metaverse, shop the Marketplace, attend events and check out what others are up to!
- **CREATE** personal experiences by building avatars, domains, tablet apps, and more for you and others to enjoy.
- **SCRIPT** and express your creativity by applying advanced scripting concepts to entities and avatars in the metaverse.
- **HOST** and make immersive experiences to educate, entertain, and connect with your audience.
- **SELL** your creations to others and make money in the metaverse using the High Fidelity Marketplace.
- **CONTRIBUTE** to our endeavor by browsing our source code on GitHub.

### 1.1 Home

High Fidelity is an open-source software where you can create and share virtual reality (VR) experiences. You can create and host your own VR world, explore other worlds, meet and connect with other users, attend or host live VR events and much more.

The High Fidelity metaverse provides built-in social features, including avatar interactions, spatialized audio and interactive physics. Additionally, you have the ability to import any 3D object into your virtual environment. No matter where you go in High Fidelity, you will always be able to interact with your environment, engage with your friends, and listen to conversations just like you would in real life.

#### 1.1.1 What can I do?

You have the power to shape your VR experience in High Fidelity.

- **EXPLORE** by hopping between domains in the metaverse, shop the Marketplace, attend events and check out what others are up to!
HiFi Doc Documentation, Release 1.0

- **CREATE** personal experiences by building avatars, domains, tablet apps, and more for you and others to enjoy.
- **SCRIPT** and express your creativity by applying advanced scripting concepts to entities and avatars in the metaverse.
- **HOST** and make immersive experiences to educate, entertain, and connect with your audience.
- **SELL** your creations to others and make money in the metaverse using the High Fidelity Marketplace.
- **CONTRIBUTE** to our endeavor by browsing our source code on GitHub.

1.2 Explore

High Fidelity is a metaverse where you can connect and create with others. We invite you to explore VR worlds created in High Fidelity and interact with other users. You can visit your friend’s VR world, meet people, attend events and even go for a class on avatar creation. It’s an immersive and interactive experience with realistic visuals and audio.

If it's your first time using High Fidelity, you’ll start your journey a local space by yourself, where you can experiment with the controls and learn about the product. When you are ready, try exploring other domains using the GoTo and socialize with people around the metaverse. If you’ve visited High Fidelity before, you will return to the location where you last visited.

Throughout this chapter, learn how to make the most of your exploration:

1.2.1 Get Started with High Fidelity

We know that getting started with a new application can be difficult: installing the software and learning the controls is never any fun. Hopefully, this section will help you become familiar with our application, so that you can begin making friends and exploring the metaverse.

In This Section

Install High Fidelity

High Fidelity has two different installers. The Client Installer comes with everything you need to view and interact with High Fidelity’s content and users. However, you are unable to host content using this installer. The Client + Sandbox Installer has everything the client installer does, but it also enables you to host your own content and share it with the world.
Minimum System Requirements

In order to run High Fidelity in either VR or Desktop mode, ensure that your computer meets these minimum system requirements:

• Windows 10, 64-bit or MacOS High Sierra (10.13)
• 8GB+ RAM i5
• A wide range of Nvidia, AMD, and Intel graphics cards are supported

In addition, your network must have enough internet bandwidth to run High Fidelity:

• If you are using Interface only to explore the metaverse, then you must have internet speeds of at least 10 Mbps download/1 Mbps upload.
• If you are hosting a domain on your Sandbox, you need to add 10 Mbps upload for each user you want to concurrently visit your domain.

Note: We also run on Linux devices; however, we do not publish an installer for Linux machines. To use High Fidelity on Linux, you will need to build the application from our code base. For more information, see our Linux Build Guide.

Download High Fidelity Installer

You can download High Fidelity through Oculus store, Steam, or High Fidelity’s website:

• Download the Client Installer
• Download the Client + Sandbox Installer

If you intend to use High Fidelity in VR mode with a HMD, ensure that Steam VR is also installed on your system before launching High Fidelity.

Upgrade High Fidelity

High Fidelity is always changing, as we work to improve performance and add features that will enhance your experience in the metaverse. If a new version has been released, you will be prompted to upgrade your installation the next time you run High Fidelity. At any time, you can also download the latest release from our website.

You cannot upgrade if Sandbox or the Console is running in the background of your computer. Be sure to quit these applications before upgrading.

• For Windows, locate the High Fidelity app in your system tray. Right-click on the icon and select ‘Quit’. Alternatively, end the ‘server-console’ background process using the Task Manager.
• For OS X, locate the High Fidelity icon in the menu bar. Click on the icon and select ‘Quit’.

For more information on the latest releases, see our Release Notes.

Perform a Clean Install

If you’re facing problems when you load Interface and Sandbox, you can try performing a clean install. A clean install removes multiple files and settings that you may need once you install High Fidelity again.
Note: Ensure that you back up the following files before a clean install: Favorites, Wearables, Sandbox, and Entities. These settings will be deleted during the clean install.

Windows Clean Install

1. Click on the Start menu and type “Add or Remove Programs” in your Windows search bar.
2. Uninstall any versions of High Fidelity that are visible (Including any Steam installs).
3. Once High Fidelity is uninstalled, browse to your %Program Files% directory. Delete all folders related to High Fidelity. If you installed through Steam, these folders will be located at C:/Program Files(x86)/Steam/steamapps/common.

Warning: The next 3 steps will permanently delete your Sandbox content. If you wish to keep this content, copy %AppData%/Local/High Fidelity/assignment-client to another location on your computer before proceeding. Repeat for %AppData%/Roaming/High Fidelity/assignment-client.

4. Browse to your local %AppData% folder (usually C:/Users/<your_username>/AppData/Local). If you do not see the folder, make sure you can view hidden folders. In File Explorer, click View and make sure “Hidden Items” is checked. Delete all folders related to High Fidelity.
5. Browse to your roaming %AppData% folder (usually C:/Users/<your_username>/AppData/ Roaming). Delete all folders related to High Fidelity.
6. Re-install High Fidelity using the steps above. To restore your Sandbox content, copy the assignment-client folders you backed up back to their respective locations.

Mac Clean Install

1. Open your Applications folder and delete the High Fidelity folder.
2. Open the <username>/.config folder. This is a hidden folder than is accessible by going to Go > Home. Press the keyboard shortcut Command + Shift + . (period).
3. Delete the highfidelity.io folder.
4. Open the ~/Library folder by holding the Option key and clicking the ‘Go’ menu while in the Finder. The Library option should appear in the menu.
5. Browse to ~/Library/Application Support and delete the High Fidelity folder.
6. Empty the Trash.
7. Re-install High Fidelity using the steps above.

See Also
• Install Your Domain

Understand the Architecture

High Fidelity’s architecture shows how different parts of the system work together to give you the best VR experience.
Architecture Overview

High Fidelity’s architecture consists of the following components that work together and send data to each other for your VR experience.

- The **High Fidelity Interface** runs your personal experience in the metaverse. With it, you can **visit VR worlds, meet people, attend live events** and more.

- The **Domain Server** is the server that hosts a domain. The domain server **hosts the content** in the domain, and manages the **domain-wide settings**, such as audio spatialization, user permissions, and running scripts.

- The **Global Services** connect all of the servers together. These services are maintained by High Fidelity so that you can **sign in**, **move seamlessly between places**, and **purchase items on the Marketplace**.
High Fidelity Interface

The High Fidelity Interface (or simply ‘Interface’) is the main user interface for High Fidelity. It is used to explore the metaverse and engage with people from around the world. When you enter a domain, your Interface connects with the domain server that is hosting the virtual world, alongside any global services.

You can download and use the Interface on your computer or your Android phone using the Client-Only Installer.

Physics Engine

Your VR experience won’t be realistic without some physics. High Fidelity includes a physics engine that simulates behaviors of objects according to the Newtonian laws of physics. When an object falls to the ground and bounces, or when two or more objects collide, their movements are computed by the physics engine.

Each Interface runs its own physics engine, and the entity server coordinates the results to produce a consistent simulation across the entire domain.

Domain Server

A domain is a spatial simulation in High Fidelity that you can visit. It is computed by a stack of programs on one or more computers. You need a domain’s place name to visit a domain, just like you would need a web address to visit a website.

You can set up your own domain and host it on your local machine or on a cloud server to make it available to other users. Your domain’s server stack is a set of components that simulate and manage different aspects of the domain such as audio, entities, and avatars. Everything that you see, hear, and do in your domain is managed by the server stack.
Server Stack

The Domain Server is at the top of this stack and its job is to give out assignments to the other components. These components are called Assignment Clients, because from the perspective of the domain server, they are clients that take on different roles.

The server stack is not only controlling, managing and computing your domain as you see it, but also how it is seen by anyone visiting your domain. This means that the domain server hands out simulation assignments and provides their IP addresses to connecting Interface clients. The domain server is a single executable that spawns assignment clients that become the different mixers as requested. Each assignment client can function as one of the six types mentioned. The domain server determines which assignment client functions as which mixer.

Assignment Clients

Assignment clients control and manage various aspects of a domain. They also communicate directly with the Interface clients connected to a domain. There are six types of assignment clients:
### Avatar Mixer

This mixer is in charge of your virtual presence in any domain. It keeps track of where you are, which avatar you’re wearing, and how you move around the domain. For example, it tracks how you move your head while wearing a Head Mounted Display (HMD).

### Audio Mixer

Mixes all sounds, whether it’s voice or environmental. And it does this not just for avatars, but also for all the entities in a domain. The Audio mixer can customize a stereo mix for you based on your position relative to the audio source.

### Entity Server

Tracks all entities and their properties in a domain, from their description and position, to any behaviors attached to them in a script. If an entity is modified, the change is communicated to the entity server, which in turn relays the information to all clients currently visiting the domain.

### Asset Server

Provides copies of the models, audio files, scripts, and other media used by the domain. It functions like a Web server, but using protocols tuned to High Fidelity’s architecture.

### Agent

Executes user-written JavaScript programs. If you’ve written a script to get your avatar to clap, or create a bowling alley, the Agent will execute it. It can see entities, avatars, and send audio.

### Messages Mixer

Provides communication between scripts running in different programs connected to the domain, which could be Interfaces or Agents.

---

**Note:** Sandbox manages all these components on the domain server, five dedicated assignment clients, and as many agent assignments clients as needed. However, it is possible to spread the assignment clients over multiple computers, and even to divide each function among a hierarchy of assignment components, which may be on different computers. For instance, multiple audio mixers could be used to mix the audio in different geographic regions of the domain.

---

**Global Services**

High Fidelity maintains global services to connect different servers together.
Use Your VR Equipment

To get the best and most immersive experience in High Fidelity, we encourage you to use VR equipment, such as the Oculus Rift or HTC Vive. With these HMD devices and hand controllers, you will be able to interact with people in 3D, track body movements, and engage with the objects around you. We support the following VR equipment:

<table>
<thead>
<tr>
<th>Head Mounted Displays</th>
<th>Hand Controllers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oculus Rift (CV1 and S)</td>
<td>Oculus Touch</td>
</tr>
<tr>
<td>HTC Vive</td>
<td>HTC Vive</td>
</tr>
<tr>
<td>Windows MR</td>
<td>XBox One Controller</td>
</tr>
</tbody>
</table>

On This Page

• **Use Your VR Equipment**
  – **VR Controls**
  – **Comfort Mode**
  – **Change How You Move in VR**
  – **Motion Capture Using Vive Trackers**
VR Controls

**Oculus Touch Controls**

- Far Grab / Near Grab
- Toggle Tablet
- Manipulate Far Grabbed Object
- Jump*/Toggle Fly*
- Near Grab
- Push to Talk
- Forward / Teleport complete*
- Left Strafe* △ Right Strafe*
- Backward
- Teleport Laser On*
- Turn Left 10° ◀ Right 10°

* When enabled under Settings > Controls > VR Movement
1.2. Explore

* When enabled under Settings > Controls > VR Movement.
Comfort Mode

Motion sickness is a real problem for many people when they put on a HMD and enter VR. This happens because your eyes experience movement in VR, while your body stands still. If you experience motion sickness and discomfort using VR equipment, you are not alone.

“Comfort mode” is designed to decrease the effects of motion sickness while using High Fidelity. This mode:

- Disables sharp turns
- Decreases your field of vision by darkening the edges of the screen
- Adds a ground plane and grid

All of these features were developed to help you orient yourself when moving around in VR.

To enable comfort mode, go to Menu > Settings > Controls on your tablet. Use the slider to adjust how much of the environment you see in VR.
Change How You Move in VR

You can change many avatar movement settings in VR such as jumping, flying, and leaning behavior. To do so:

- In Desktop mode, go to **Settings > Controls** in the menu bar.
- In VR mode, open your Tablet and go to **Menu > Settings > Controls**.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VR Movement &gt; Teleporting</strong></td>
<td>Enables teleport controls to move seamlessly between positions within a domain.</td>
</tr>
<tr>
<td><strong>VR Movement &gt; Walking</strong></td>
<td>Enables walking controls to move within a domain.</td>
</tr>
<tr>
<td><strong>VR Movement &gt; Strafing</strong></td>
<td>Enables strafing controls (to walk sideways).</td>
</tr>
<tr>
<td><strong>VR Movement &gt; Jumping and flying</strong></td>
<td>Enables jump and fly controls.</td>
</tr>
</tbody>
</table>
| **VR Movement > Movement Direction** | This setting controls which direction you move in:  
  - **HMD-Relative**: Move in the direction your head is pointing.  
  - **Hand-Relative**: Move in the direction your dominant hand is pointing.  
  - **Hand-Relative (Leveled)**: Move in the direction your hand is pointing, without taking pitch into account. |
| **VR Movement > Dominant Hand** | Select ‘Left’ or ‘Right’. Teleport and turn controls move to the controller in the dominant hand. |
| **VR Movement > Rotation Mode** | This setting controls how you turn in VR:  
  - **Snap turn**: Rotate your avatar sharply to the left or the right.  
  - **Smooth turn**: Rotate your avatar smoothly as you turn to the left or right.  
  - **Comfort mode**: Enable comfort mode and decrease the field of vision while moving in VR mode. |
| **VR Movement > Control Scheme Selection** | This setting determines how you control your walking speed:  
  - **Default**: Your walking speed will remain the same, no matter how far forward you push your controller’s joystick. Fully pushing the joystick forward will make your avatar run.  
  - **Analog**: Your walking speed changes based on how far forward you push your controller’s joystick. Fully pushing your joystick forward will make your avatar run.  
  - **Analog++**: Your walking speed changes based on how far forward you push your controller’s joystick. You can use the slider to change the maximum walking speed in meters/second. Fully pushing your joystick forward will make your avatar run. |
| **VR Movement > Avatar leaning behavior** | This setting controls if and when your avatar leans in VR mode.  
  - **Auto**: This is the default setting. Your avatar will lean if you are standing in the real world.  
  - **Seated**: Your avatar will not lean if you are sitting in the real world.  
  - **Standing**: Your avatar will lean if you are sitting in the real world.  
  - **Disabled**: Your avatar can sit on the floor (experimental). |

**User real world height (in meters)**  
You can change your real world height for better tracking in VR mode.
Motion Capture Using Vive Trackers

You can enhance your High Fidelity experience using full body motion capture (mocap). High Fidelity currently supports mocap using HTC Vive Trackers.

Vive trackers need to be strapped to the body part you wish to track. You can replace the HMD and hand controllers with trackers if you only need to track the movement of your head and hands.

You can set up different mocap systems:

<table>
<thead>
<tr>
<th>Mocap System</th>
<th>Equipment Needed</th>
<th>Recommended Straps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head</td>
<td>HMD or 1 Vive Tracker</td>
<td>Head strap for Vive Tracker</td>
</tr>
<tr>
<td>Hands</td>
<td>Hand controllers or 2 Vive Trackers</td>
<td>Hand strap for Vive Tracker</td>
</tr>
<tr>
<td>Head + Hands + Feet</td>
<td>2 Vive Trackers + HMD + 2 Hand Controllers</td>
<td>Foot straps</td>
</tr>
<tr>
<td>Head + Hands + Feet + Hips</td>
<td>3 Vive Trackers + HMD + 2 Hand Controllers</td>
<td>Hip Strap: Drill a hole in the back of a thick leather belt and attach the tracker using a 1/4” screw.</td>
</tr>
<tr>
<td>Head + Hands + Feet + Hips + Chest</td>
<td>4 Vive Trackers + HMD + 2 Hand Controllers</td>
<td>Chest straps</td>
</tr>
<tr>
<td>Head + Hands + Feet + Hips + Shoulders</td>
<td>5 Vive Trackers + HMD + 2 Hand Controllers</td>
<td>Shoulder straps</td>
</tr>
</tbody>
</table>

**Note:** You can replace the HMD and hand controllers with trackers if you only need to track the movement of your head and hands.
Configure Your Mocap System

1. Strap your Vive trackers to your body as shown in the image.
2. Connect your trackers, HMD, and controllers to SteamVR.
3. In Interface, pull up your HUD or Tablet and go to Menu > Settings > Calibration.
4. Configure your mocap system by:
   - Selecting the right device for your head and hands. If you’re using a head tracker instead of an HMD, click ‘Use HTC Vive Devices in Desktop Mode’.
   - Selecting the body position of any additional trackers.
5. Click ‘Apply and Calibrate’.

6. Stand in a T-Pose until the timer counts down to zero:
   - Feet together
   - Arms out
   - Head looking straight ahead.

7. Check to see that each tracker is tracking the corresponding joint on your avatar.

8. You can also calibrate your trackers without using your tablet. Once you apply your configuration, stand in a T-Pose and hold the following four buttons together for 1 second: Left Trigger, Right Trigger, Left Menu Button, Right Menu Button. You can press the same buttons together for a second to remove your calibration from the trackers.

Note: When you setup your Vive, you choose which way to point the arrow as your reference. During calibration, it is important that you face the same direction. If you can not remember the arrow’s direction, press the Vive System Menu Button and look on the ground for a marker. This is important to make sure your joints are oriented correctly.
## Troubleshooting

<table>
<thead>
<tr>
<th>Issue</th>
<th>Troubleshooting Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>My calibration failed</td>
<td>• Check if your trackers are properly connected in SteamVR.</td>
</tr>
<tr>
<td></td>
<td>• Have you selected the correct configuration in your tablet and do you have enough</td>
</tr>
<tr>
<td></td>
<td>number of trackers to support that configuration?</td>
</tr>
<tr>
<td></td>
<td>• If you are performing and not in HMD, did you select to ‘Use HTC Vive in Desktop</td>
</tr>
<tr>
<td></td>
<td>Mode’?</td>
</tr>
<tr>
<td></td>
<td>• Are any of the trackers blinking? If so, they may need to be paired again.</td>
</tr>
<tr>
<td></td>
<td>• Do you have the correct number of dongles plugged in to your computer? You will</td>
</tr>
<tr>
<td></td>
<td>need one dongle per tracker. If you are performing with all 7, then you may need</td>
</tr>
<tr>
<td></td>
<td>a USB hub to handle them.</td>
</tr>
<tr>
<td>My sensor is jiggling a lot</td>
<td>Make sure the straps on the sensor are tightened.</td>
</tr>
<tr>
<td>My sensor keeps losing tracking</td>
<td>• If it’s the hip tracker, is your shirt is tucked in and not covering the puck?</td>
</tr>
<tr>
<td></td>
<td>Also make sure your headphone cord isn’t covering the puck.</td>
</tr>
<tr>
<td></td>
<td>• Can the base stations clearly see the tracker?</td>
</tr>
<tr>
<td></td>
<td>• Is the signal from the base station conflicting with another Vive setup nearby?</td>
</tr>
<tr>
<td></td>
<td>• Are you clear of reflective surfaces nearby? (such as picture frames, whiteboards,</td>
</tr>
<tr>
<td></td>
<td>shiny tables).</td>
</tr>
<tr>
<td></td>
<td>• Is the lighting consistent across the room (minimal outdoor lighting)?</td>
</tr>
<tr>
<td></td>
<td>• Try restarting SteamVR.</td>
</tr>
</tbody>
</table>

**Note:** Remember to charge your trackers when you aren’t using them so that you don’t have to deal with a low battery tracker negatively impacting your performance.

## Gamepad

If your HMD does not come equipped with hand controllers, you can use a gamepad. However, High Fidelity is best experienced with VR equipment or the keyboard in Desktop mode.
Explore in Desktop Mode

Desktop users are restricted to using their keyboard and mouse to do things in High Fidelity. On this page, you can find all of the shortcuts so that you can enjoy your experience in High Fidelity to the fullest.

**Note:** The shortcuts below are for Windows and Linux computers. If you’re running on a Mac, use the same commands, substituting the Command key for the Ctrl key.
Movement Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walk Forward</td>
<td>W or UP ARROW</td>
</tr>
<tr>
<td>Walk Backward</td>
<td>S or DOWN ARROW</td>
</tr>
<tr>
<td>Run</td>
<td>Hold SHIFT while using any shortcut to walk</td>
</tr>
<tr>
<td>Side Step to the Left</td>
<td>Q or Right Click + A or SHIFT + LEFT ARROW</td>
</tr>
<tr>
<td>Side Step to the Right</td>
<td>E or Right Click + D or SHIFT + RIGHT ARROW</td>
</tr>
<tr>
<td>Turn Left</td>
<td>A or LEFT ARROW</td>
</tr>
<tr>
<td>Turn Right</td>
<td>D or RIGHT ARROW</td>
</tr>
<tr>
<td>Jump</td>
<td>SPACE or PGUP</td>
</tr>
<tr>
<td>Fly</td>
<td>Hold SPACE or PGUP</td>
</tr>
<tr>
<td>Fly Down</td>
<td>C or PGDN</td>
</tr>
</tbody>
</table>

In-World Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handshake</td>
<td>X</td>
</tr>
<tr>
<td>Toggle Privacy Shield</td>
<td>CTRL + N</td>
</tr>
<tr>
<td>Open Tablet</td>
<td>TAB (when 'Desktop becomes toolbar' is not checked)</td>
</tr>
<tr>
<td>Select Item</td>
<td>Left Click</td>
</tr>
<tr>
<td>Grab Item</td>
<td>Left Click</td>
</tr>
<tr>
<td>Inspect Item</td>
<td>Right Click</td>
</tr>
<tr>
<td>Open Browser</td>
<td>CTRL + B</td>
</tr>
<tr>
<td>‘Toggle ‘Away from Keyboard’</td>
<td>ESC</td>
</tr>
<tr>
<td>‘Toggle ‘Mute Mic’</td>
<td>CTRL + M</td>
</tr>
<tr>
<td>‘Toggle ‘Show Statistics’</td>
<td>/</td>
</tr>
<tr>
<td>Screenshot</td>
<td>P</td>
</tr>
<tr>
<td>Push To Talk</td>
<td>T</td>
</tr>
</tbody>
</table>
Camera Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Person View</td>
<td>1</td>
</tr>
<tr>
<td>Third Person View</td>
<td>3</td>
</tr>
<tr>
<td>Mirror View</td>
<td>2</td>
</tr>
<tr>
<td>Pan In/Out</td>
<td>With mouse, use mouse wheel. On trackpad, drag two fingers up or down.</td>
</tr>
<tr>
<td>Take Screenshot</td>
<td>P</td>
</tr>
</tbody>
</table>

Avatar Sizing Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrease Avatar Size</td>
<td>–</td>
</tr>
<tr>
<td>Increase Avatar Size</td>
<td>+</td>
</tr>
<tr>
<td>Reset Avatar Size</td>
<td>=</td>
</tr>
</tbody>
</table>

Create and Edit Mode

These controls work when the Create app is open.

<table>
<thead>
<tr>
<th>Action</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>CTRL + Z</td>
</tr>
<tr>
<td>Redo</td>
<td>CTRL + Y</td>
</tr>
<tr>
<td>Delete Entity</td>
<td>DEL</td>
</tr>
<tr>
<td>Focus on Selected Entity</td>
<td>F</td>
</tr>
<tr>
<td>Align Grid to Bottom of Selected Entities</td>
<td>G</td>
</tr>
<tr>
<td>Duplicate Entity</td>
<td>ALT + Left Click + Drag</td>
</tr>
<tr>
<td>Parent Entity</td>
<td>CTRL + P</td>
</tr>
<tr>
<td>Unparent Entity</td>
<td>CTRL + SHIFT + P</td>
</tr>
<tr>
<td>Copy Entity</td>
<td>CTRL + C</td>
</tr>
<tr>
<td>Paste Entity</td>
<td>CTRL + V</td>
</tr>
<tr>
<td>Toggle Global/Local Translation</td>
<td>T</td>
</tr>
</tbody>
</table>

Gamepad

Instead of a keyboard, you can use a gamepad while experiencing High Fidelity in desktop mode.
Adjust Your Settings

You can adjust various settings in High Fidelity so that it runs to your preferences. Many of these settings are changed using the HUD (in Desktop mode) or Tablet (in VR mode).

The Tablet and HUD

In VR, all of your settings are found in your Tablet. The Tablet also gives you easy access to any apps that you install. Pull up the tablet by clicking the menu button on your controller.
In Desktop Mode, you have the option to use either the Tablet or a smaller version called the “Heads-up Display” or HUD. It contains the exact same options as the Tablet (settings, apps, etc), but it takes up less space on your screen. To enable the HUD, first enable the Developer menu by going to Settings > Developer Menu. Then, go to Developer > UI > Desktop Tablet Becomes Toolbar.

Enter or Exit VR Mode

You can enjoy High Fidelity with or with VR equipment such as head mounted displays (HMD), hand controllers and audio headsets. Our Desktop mode contains many of High Fidelity’s features such as audio, basic movements and gestures, and the ability to travel to different domains.

Keep in mind, however, that the most immersive and powerful experience is when you use VR equipment. Only then will you be able to interact with people in 3D, track body movements, and easily engage with the objects around you. Once you have set up your VR equipment, you can easily switch between VR mode and Desktop mode. To switch to VR mode, use one of the following methods:

1. From the HUD, click **Enter VR**.
2. Click the **Display** menu, then select your VR device.

To exit from VR mode, remove your headset, click **Exit VR** on the HUD or press **ESC** on your keyboard.
Change Your Audio Settings

High Fidelity spatializes audio in real-time based on the location of the listener and the sound sources. A good headset will enhance your experience. You can adjust your audio settings by going to Audio on your tablet or HUD.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mute microphone</td>
<td>Mute or unmute your microphone.</td>
</tr>
<tr>
<td>Warn when muted</td>
<td>Enable to receive a warning on your screen when your microphone is muted.</td>
</tr>
<tr>
<td>Noise reduction</td>
<td>Enable or disable noise reduction.</td>
</tr>
<tr>
<td>Audio level meter</td>
<td>By default, the audio level meter is visible on the top left corner of your screen. Uncheck this box to hide the meter.</td>
</tr>
<tr>
<td>Push To Talk (T)</td>
<td>Enable or disable Push To Talk for other users to hear you when your microphone is muted. When enabled, press and hold T to talk in desktop mode and press both triggers on your controllers in VR mode. This feature works only when you are focused on the Interface window.</td>
</tr>
<tr>
<td>Stereo input</td>
<td>Enable or disable stereo input. Stereo reproduces sound using two or more audio channels. This means that you will hear sound from various directions, like how you would in the real world.</td>
</tr>
<tr>
<td>Choose Input Device</td>
<td>Choose the microphone or input device of your choice from the list of devices displayed.</td>
</tr>
<tr>
<td>Choose Output Device</td>
<td>Choose the speakers, headphones, or other output device of your choice from the list of devices displayed.</td>
</tr>
<tr>
<td>People volume</td>
<td>Use the slider to increase or decrease the volume of other users in the domain.</td>
</tr>
<tr>
<td>Environment volume</td>
<td>Use the slider to increase or decrease the volume of ambient sound in the domain. For example, use this option to decrease the sound of background music or special effects like fireworks.</td>
</tr>
<tr>
<td>System Sound volume</td>
<td>Use the slider to increase or decrease your UI volume like tablet clicks. Changing this value will not affect the People or Environment volume.</td>
</tr>
</tbody>
</table>

Input Devices

An audio input device is any software or hardware device that accepts an output signal from another device. Examples of audio input devices include:

- A USB microphone
- A microphone headset that is plugged into your computer via the “Mic In” port
- Your sound card’s “Stereo Mix” device
  - Think of these devices as if they were microphones being held up to your speakers while they output sound
Output Devices

An audio output device produces a signal that drives another device’s input. Examples of audio output devices include:

- Desktop computer speakers
- Headphones
- Huge speakers in a movie theater

Set Your Perspective

You can choose how you view things around you by changing your perspective. To change your perspective:

- In Desktop mode, go to View in the menu on the top left corner.
- In VR mode, open your Tablet and go to Menu > View.
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Person</td>
<td>Select this setting if you want to change your perspective in High Fidelity to first person. In this view, you will not see yourself, only the environment around you.</td>
</tr>
<tr>
<td>Third Person</td>
<td>Select this setting to change your perspective to third person. In this view, you will see yourself, as well as the environment around you.</td>
</tr>
<tr>
<td>Mirror</td>
<td>Select this to change your perspective to mirror. In this view, you will see yourself and the space behind you.</td>
</tr>
<tr>
<td>Mirror</td>
<td>NOTE: You cannot move in mirror mode. It is for viewing only.</td>
</tr>
<tr>
<td>Independent Mode</td>
<td>Select this to change what you see through scripting instead of avatar’s movements.</td>
</tr>
<tr>
<td>Entity Mode</td>
<td>Select this to set your perspective to a specific entity, allowing you to move with entity as it moves.</td>
</tr>
</tbody>
</table>
Other Miscellaneous Settings

Here are some other settings you may like to change to optimize your experience.

General Settings

You can modify general settings such as user interface and privacy settings in High Fidelity.

- In Desktop mode, go to Settings > General in the menu on the top left corner.
- In VR mode, open your Tablet and go to Menu > Settings > General.

In-World Graphics Settings

You can make changes to the graphics in High Fidelity.

- In Desktop mode, go to Settings > Graphics in the menu on the top left corner.
- In VR mode, open your Tablet and go to Menu > Settings > Graphics.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>World Detail</td>
<td>You can control the detail visible to you in High Fidelity by moving this slider left to decrease and right to increase.</td>
</tr>
<tr>
<td>Show Shadows</td>
<td>Enable or disable viewing shadows. This setting is enabled by default. If you have a low game rate or are having trouble loading a domain, disabling this option may improve your performance.</td>
</tr>
</tbody>
</table>

Account Security Settings

You can change your account security settings in High Fidelity.

- In Desktop mode, go to Settings > Security in the menu on the top left corner.
- In VR mode, open your Tablet and go to Menu > Settings > Security.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account</td>
<td>Enable to stay logged in (in the current device) even if you exit High Fidelity.</td>
</tr>
<tr>
<td>Secure Transactions</td>
<td>Change your security picture.</td>
</tr>
</tbody>
</table>

See Also

- Configure Your Domain Settings

Get Our Android App

An unreleased version of High Fidelity is currently available to download for Daydream-enabled Android devices. As an unreleased application, it does not have the full functionality of the desktop or VR version of High Fidelity. However, you can explore a number of worlds, attend events, change avatars, and connect with friends directly from your phone.
Note: The app is designed for Daydream-ready phones only. Keep in mind, it cannot be used in Daydream View and is only available as a 2D application.

On This Page

- Get Our Android App
  - Visit Different Worlds
  - Movement Controls
  - In-World Controls
  - Avatar Controls
  - Discover and Make Friends

Visit Different Worlds

High Fidelity has many virtual places where you can interact with other users and participate in various activities or events. We have modified several of High Fidelity’s most popular virtual worlds to make them more accessible to Android users.

Find these domains by going to the Home tab in the menu.
Movement Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walking</td>
<td>When exploring a virtual world, use the arrows in the bottom left corner to turn and/or walk.</td>
</tr>
<tr>
<td>Turning</td>
<td>Drag your finger left or right across the screen to make your avatar turn.</td>
</tr>
<tr>
<td>Look up/down</td>
<td>Drag your finger up or down to change the angle of the camera.</td>
</tr>
<tr>
<td>Flying</td>
<td>Press the button on the bottom right with the winged avatar to fly (the longer you hold it, the higher you go!).</td>
</tr>
</tbody>
</table>

In-World Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>Switch to a bird’s eye view camera by pressing the My View button on the top right corner of the screen.</td>
</tr>
<tr>
<td>Mute</td>
<td>Your avatar in muted by default when you open the app. Press the mic button on the top right corner to unmute.</td>
</tr>
</tbody>
</table>

Avatar Controls

<table>
<thead>
<tr>
<th>Action</th>
<th>Controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Your Avatar</td>
<td>Change your avatar to one available in the list in the Avatar tab in the menu.</td>
</tr>
<tr>
<td>Set Display Name</td>
<td>Set your display name in the Avatar tab in the menu</td>
</tr>
</tbody>
</table>

Discover and Make Friends

<table>
<thead>
<tr>
<th>Action</th>
<th>Controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handshake</td>
<td>Add people to your Connections by shaking their hand. Press the handshake button which is above the flight button in the bottom right corner of your screen.</td>
</tr>
<tr>
<td>Connections List</td>
<td>Open the People tab in the menu to view your Connections.</td>
</tr>
</tbody>
</table>

Additional functionality such as opening the Tablet, using the Create Tools app, adding wearables, etc. are not yet available.

See Also

- Socialize with Others

1.2.2 Personalize Your Experience

Before you even enter High Fidelity, there are many ways to personalize your experience and make it your own. You can customize everything from the hardware you use, to the way the app works, to how you appear to others.

In This Section:
Change Your Avatar

When you first use High Fidelity, you will be wearing the default avatar, Woody. Your avatar is a representation of you in the metaverse. You can control how your avatar moves and speak to other users in-world using it.

On This Page:

• Buy an Avatar from the Marketplace
• Use Your Own Custom Avatar

Buy an Avatar from the Marketplace

We and our users have designed multiple avatars that are available for your use in the Marketplace.

To get an avatar from the Marketplace:

1. In Interface, pull up your Tablet or HUD and go to Market.

2. Search for avatars, or look for avatars under ‘Categories’.

3. You’ll see a list of avatars of different designs available. Click on any one you like and hit ‘Get’ (for free items)
or ‘Buy’ (for purchased avatars).

4. After completing your purchase, click ‘Wear’ to switch to your new avatar.

**Use Your Own Custom Avatar**

You can use an avatar that you created. Learn more about how you can create your avatar [here](#).

Once you have your avatar’s .fst file, you can upload it.

1. In Interface, pull up your tablet or HUD and click on Avatar.
2. In the Avatar window, click the link icon next to your current avatar.
3. Enter the .fst file’s URL and click ‘Confirm’.
4. If you want to access this avatar later without loading the .fst file information again, you can click on ‘Add to Favorites’ to save the current avatar information.

See Also

• Create Your Own Avatar
• Find and Use an Existing Avatar
• Package and Host Your Avatar
• Add Your Avatar to the Marketplace

Put On Wearables

You can customize your avatar’s appearance by adding wearables, such as a pirate’s hat, a pair of sunglasses, or even a pair of trousers that you designed. Like avatars, you can buy wearables from the Marketplace or use your imagination to create your own.

On This Page:

• Buy a Wearable from the Marketplace
• Wear Your Own Wearable

Buy a Wearable from the Marketplace

We and our users have designed many wearables that are available for your use in the Marketplace.

1. In Interface, pull up your Tablet or HUD and go to Market.

2. Search for a specific wearable, or browse all wearables by selecting the ‘Wearables’ category.

3. Click on any one you like and hit ‘Get’ (for free items) or ‘Buy’ (for purchased wearables).
4. After completing your purchase, click ‘Wear’ to put on your new wearable.

**Wear Your Own Wearable**

You can put on a wearable that you created. All wearable must be hosted in the cloud before they can be used with High Fidelity. Examples of cloud storage options include Amazon S3, Google Cloud Storage, GitHub, or Microsoft Azure. Alternatively, you can add your wearable to the Marketplace to sell to other users.

Once you have your wearable’s .fbx file, you can upload it.

1. In Interface, pull up your tablet or HUD and click on Avatar.

![Avatar interface](image)

2. In the Avatar window, click the hat icon next to ‘Wearables’.
3. Click ‘Add custom’ at the top of the window.
4. Enter the .fbx file’s URL and click ‘Confirm’.
5. Select the joint you’d like to use for your wearable. For example, a hat would be on your head, and fairy wings would be on your spine.
6. Fine tune the placement using the ‘Position’ and ‘Rotation’ options.
7. Check ‘Is soft’ if the item is rigged with your skeleton. This allows the item to move and bend with the avatar as it moves.
8. Click ‘Save’.

**See Also**

- Create Wearables
- Add Your Wearable to the Marketplace

**Install Tablet Apps**

You can enhance your experience in High Fidelity by installing tablet apps from the Marketplace. These apps enable you to get more out of High Fidelity. Do you want to record your avatar dancing, take selfies, or finger paint? Or are you looking to livestream to YouTube or make your avatar clap? Our Marketplace has a number of apps that can help you customize your experience in-world.

If you can’t find an app for what you’d like to do, you can create your own and upload it to our Marketplace for your personal use or to sell to others.

**On This Page:**
• **Install Tablet Apps**

• **Update an App**

• **Our Marketplace Apps**
  – Spectator Cam: Record or Livestream in High Fidelity
  – Finger Painting
  – Text to Speech

### Install Tablet Apps

We and many users have created tablet apps that are available in High Fidelity’s Marketplace. To find and install an app:

1. In Interface, pull up your HUD or Tablet and go to **Market**.
2. Search for a specific app, or browse all apps by selecting the ‘Apps, Scripts, & Tools’ category.
3. Click the app of your choice to purchase it and hit ‘Get’ (for free items) or ‘Buy’ (for purchased apps). Once you’ve purchased your app, you can view it in the **Inventory** app or ‘Recent Activity’.
4. After completing your purchase, click ‘Install App’ to start using the app in High Fidelity.
5. Once the app is installed, click ‘Open App’ to launch. You can also access the app through your HUD or Tablet.

### Update an App

When an app has an update available, a notification will pop up in the top right corner of the Interface when you open High Fidelity. Additionally, a red dot will appear on the **Inventory** icon of your HUD or Tablet to indicate an update is available to download.

To update your app:

1. In Interface, pull up your HUD or Tablet and go to **Inventory**.
2. In the **Inventory** app, click the notification that says “You have X item updates available.”
3. Scroll to the app you’d like to update and click on the menu.
4. Click the ‘Update’ button, then ‘Update for Free’.

### Additional Notes on Updating

• If you decide not to update an item, the old product will still be valid and listed in **Inventory**.

• Once you update an item, the previous version of the product will no longer be available.

• If you purchased multiple instances of an item, you need to individually update each instance of an item in your **Inventory**.
• When you update a product with the old version already in-world, it will remain intact until it is replaced manually with the updated version. If you attempt to import an entities JSON which contains the old version, the system will ignore the item, and not automatically replace it in-world. It is up to the end-user to replace the old instances with the new.

Our Marketplace Apps

We’ve listed some of the apps we created and how you can use them.

Spectator Cam

The Spectator Camera is a camera you can use to record or livestream what you and your friends do in High Fidelity. It is only available in HMD mode, and requires a recording software such as Open Broadcaster Software (OBS) to work correctly. Check out their official overview guide for more details.

Use the instructions above to install the Spectator Camera app.

To use the Spectator Camera app:

1. In Interface, pull up your HUD or Tablet and go to Spectator.

2. Turn on your Spectator Cam by moving the slider. You’ll see the camera appear in-world. By default, the app’s display shows you what you see in VR through your HMD. This is to show you what you’re recording without having to take off your HMD.

3. To record yourself, switch your display from your HMD’s view to the camera view. This lets you record videos of yourself, such as your avatar dancing or trying on different wearables.

4. Enable switching views with your controllers in the Spectator app on your HUD or Tablet.
• **Rift**: If you’re using Oculus Touch controllers, click the left thumbstick to switch views.
• **Vive**: If you’re using HTC Vive controllers, press on the center of the thumb pad to switch views.

**Finger Painting**

The Fingerpaint app lets your paint your environment, your own avatar, or even another user’s avatar. *Use the instructions above* to install the Fingerpaint app.

To use the Fingerpaint app:

1. In Interface, pull up your HUD or Tablet and go to **Body Paint**.
2. First, click ‘Options’ to select what you would like to paint on - the world around you, your avatar, or another user’s avatar (with their permission).
3. Click ‘Palette’ to select a color for your paint.
4. Click ‘Brushes’ to select the settings for your brush, such as stroke width, type of brush, and special effects.

![Brushes](image)

**Text to Speech**

The Text to Speech app synthesizes the text you type into speech. You can use this app if you don’t wish to use your voice, have microphone issues, or have disabilities.

*Use the instructions above* to install the Text to Speech app.

To use the Text to Speech (TTS) app:

1. In Interface, pull up your HUD or Tablet and go to **TTS**.
2. In the app, select the Pitch and Speed of your choice.
3. Type in the text you’d like your avatar to speak.
4. Hit ‘Speak’ to hear your text. Hit ‘Stop Last’ to stop your avatar from speaking.
1.2.3 Socialize with Others

High Fidelity is all about the people you meet and the experiences you have with them. High Fidelity enables people connected by interest, community, and friendship to come together from anywhere in the world.

Make Connections and Friends

In High Fidelity, you can establish a connection with someone else by shaking hands with them. With your hand controllers, place your hands near each other and hold the grip button. Desktop users can press and hold X on their keyboard.

Once you make a connection with someone, they will appear under Connections in the People app. You will also appear on their list of connections. You will be able to see where they are in the metaverse, and you can travel to them at any time.

To mark a connection as a friend, check the box next to their name in the People app. You can make yourself available to only your friends using the People app.
The People App

The People app provides a set of tools that help users manage their interactions with people in the metaverse. It gives you a list of the people nearby (in the same domain as you), and gives you easy access to all of your connections. From the People app, you can:
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile Picture</td>
<td>This is where your profile picture will be visible. Click on the image to view your profile. You can change this image on our website.</td>
</tr>
<tr>
<td>Display Name</td>
<td>You can change your display name at any time. By default, it will be ‘anonymous’. In the image above, the display name is ‘HiFi Docs’.</td>
</tr>
<tr>
<td>Set Availability</td>
<td>This feature allows you to appear online to select groups of users: Everyone, Friends and Connections, Friends Only, or Appear Offline. The users you appear online to will also be able to teleport to your location.</td>
</tr>
<tr>
<td>Master Volume</td>
<td>Set the volume of your audio in High Fidelity.</td>
</tr>
<tr>
<td>Nearby</td>
<td>This is the list of users who are nearby in the same domain as you.</td>
</tr>
<tr>
<td>In View</td>
<td>You can check this box to view only the users in front of you in a domain. This is useful when a domain has a lot of users.</td>
</tr>
<tr>
<td>Refresh Button</td>
<td>Click this button to refresh the list of users currently in the domain.</td>
</tr>
<tr>
<td>Connections</td>
<td>This is the list of users who are your friends and connections. You can also teleport to their location from this list.</td>
</tr>
<tr>
<td>Loud</td>
<td>Click this icon next to the user you want to mute. The user will be muted only for you, not for other users in the domain. The icon also displays how loudly a user is speaking.</td>
</tr>
<tr>
<td>Names</td>
<td>This is the list of users available in the domain. You will only see their display names.</td>
</tr>
<tr>
<td>Ignore</td>
<td>If you check this box next to a user, you and the user will not be able to see or hear each other.</td>
</tr>
</tbody>
</table>

**Use the People App as an Admin**

As an administrator in a domain, you will have privileges to maintain a domain. The People app will have an additional column that allows an admin to silence and ban users in the same domain.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silence</td>
<td>You can click the icon to mute a user. This user will be muted for everyone in the domain.</td>
</tr>
<tr>
<td>Ban</td>
<td>You can click here to ban a user from the current domain. The user will not be able to enter the domain using the same account. The banned user will still have access to other High Fidelity domains.</td>
</tr>
</tbody>
</table>

**Your Privacy Shield**

You can enable a privacy shield that protects your personal space in the metaverse. When it is enabled, other people will disappear if they get too close to you. Your privacy shield is disabled by default. To enable it, pull up your tablet.
or HUD and click Shield. In Desktop mode, you can also use the keyboard shortcut \texttt{CTRL + N}.

There is also a Shield icon on the top-left corner of your Interface window, next to the audio icon, that can be used to toggle your privacy shield.

![Avatar](image)

**Attend Live Events**

One of the great things about virtual reality is that you can attend events. High Fidelity regularly hosts events such as workshops, lectures on VR, and town hall meetings to meet our team. Click here to view all upcoming events. Events are a great place to meet others and share experiences with others around the world.

To attend an event, simply go to the hosted domain at the time of the event.

**Express Yourself**

There are many ways you can express yourself in High Fidelity, such as animating the mouth of your avatar or using gestures in the Emote app.

By default, all avatars will use a standard set of animations, such as your eyes blinking or your mouth opening and closing as you talk. When you are using a VR controller, your avatar will automatically mimic your hand gestures and movements.

**The Emote App**

The Emote app is a way for desktop users to express themselves without using VR controllers. With this app, you can display feelings by: crying, acting surprised, dancing, cheering, waving, falling, pointing, clapping, sitting, or showing love.
Chat with Users

High Fidelity doesn’t yet have a default text chat option that works well for both HMD and desktop users as most HMD users can’t type easily. Our extensible open-source scripting and UI gives you the ability to create the features you want, including text chat. There are some great scripts for chat that have already been built by community members, and a few are described below.

HiFi Local Chat

This clean, reliable, and well-written chat script was created by alpha user ctrlaltdavid.

To run the script:

1. In Interface, go to Edit > Open and Run Script from URL.
2. Paste this URL.
The script will start running and display a text chat window pop-up. You can use this window to chat with other users in the same domain who are running the same script. If text chat is important to you, you can add this to your default scripts so it’s always there.

![Local Chat](sandbox)

**Local Chat**

c81c74a-a5f2-4f8d-845d-c3d805c7eda

Instructions:
- Press "v" to show and Esc to hide.
- Or View > Local Chat to show / hide.
- Type your message at the flashing cursor and press Enter.

The chat is local to the domain that you're in.

Details: http://sandboxstudio.com/hifi/chat

COM Script Version 1

AlphaVersionD has authored an equally powerful and friendly script that runs on a domain. All users that visit a domain with the script can chat with one another, without installing a separate app or script. With this script, you have the power to enable chat on any of your own domains.

**Note:** You can run a script only in a domain where you have the right permissions. Ensure that you have the right permissions in a domain where you wish to use the COM Script.

To install COM Script in your domain:

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. Click the ‘zone’ icon to create a zone entity.
3. In the ‘Properties’ tab of the zone entity, paste this **URL**.

COM Script version 1 is now running in the zone in your domain!

**See Also**

- Bank and Shop
- Tutorial: Transfer Money and Items
- Give and Receive Gifts

**1.2.4 Travel Between Worlds**

High Fidelity is made up of many virtual places that let you participate in activities and interact with the people around you. Many of these places are beautifully detailed worlds that are interesting to explore at any time, while others were built to host events and engage with the people around you.

**On This Page:**

- GoTo App
- Visit a Friend
GoTo App

The GoTo app lets you travel between different places in the High Fidelity metaverse. Many of our places are created by users just like you.

To go and explore new places:

1. In Interface, pull up your tablet or HUD and go to GoTo.

2. If you know where you want to go, enter the domain address or place name. As you type, the matching places will show up. Otherwise, browse the open places under ‘Featured’ and ‘Places’.

3. Click on a place name to go to the place. If you have permissions, you will be transported to that location automatically.

The GoTo app also shows you a visual feed of snapshots that people have taken and shared in the metaverse. Clicking on a snap will take you directly to the place where the picture was taken.
Visit a Friend

Once you’ve made a friend, you can see where they are and even teleport directly to them.

1. In Interface, open your tablet or HUD and go to People.
2. Click ‘Connections’ and find the friend you want to visit.
3. Select their name then ‘Visit’.

See Also

• Socialize with Others

1.2.5 Interact with Your Environment

In High Fidelity, your experiences are shaped the world around you. When you enter a domain, all of the space around you is built with entities, or the building blocks of your environments. The walls of your room, the tree in the distance, or the animated butterfly that flew past are all entities.

Just like in the real world, you can interact with your environment by grabbing items or colliding with objects.

On This Page:

• Grab Objects
• Collisions
• Triggered Entities

Grab Objects

You can grab objects in High Fidelity using your mouse or hand controllers. You can grab an entity, hold it, throw it, and drop it depending on the entity’s properties.

• In Desktop mode, click and hold the left mouse button to grab and hold an entity.
• In VR mode, reach out towards the object and press the Grab button. The location of this button depends on the controllers you are using.
Collisions

You can collide (or run into) objects and other avatars in High Fidelity. Likewise, objects can collide with one another. We use physics to govern how entities behave when they collide with each other or with avatars.

Without this collision property set, objects will move straight through other entities and avatars. As you interact with your environment, take note on which objects have collisions enabled based on whether or not you can walk through them.

When creating your own entities, you can set the Collision property to turn on or off collisions.

Triggered Entities

Some entities have scripts (or triggers) that make them behave a certain way when you interact with it. For example, you can trigger a light switch to turn on or off when your hand passes through it, or make a pet walk when you grab its leash.

These triggers are scripted in the entities themselves by their creators. Because of this, the possible behavior is endless. We encourage you to explore and discover all of the cool ways you can interact with your surroundings.

See Also

- Apply Physics to Entities
- Define an Entity’s Behavior
- Add Sound to Entities
- Define Interactions with Avatars

1.2.6 Bank and Shop

If you want to buy items in High Fidelity, head on over to the Marketplace, where you can purchase more than 300 items built by digital artists and creators from around this world. High Fidelity uses their own cryptocurrency, High Fidelity Coin (HFC) to manage your transactions.

On This Page:

- **Buy High Fidelity Coin**
- **Shop the Marketplace**
- **Cash Out Your HFC**

Buy High Fidelity Coin

Currently, you can buy High Fidelity Coins (HFC) using Ethereum, a blockchain app that trades ETH (Ether). The Bank of High Fidelity manages HFC and we gradually increase the number of coins in circulation as the economy grows. 100 HFC is equal to 1 USD.

To get HFC:

1. Book an appointment at the Bank of High Fidelity.
2. At the time of your appointment, visit the TradingRoom domain.
3. We will provide you with a QR code that you can use to send ETH to the Bank of High Fidelity.
4. Once we receive the ETH amount you sent, we will send you the appropriate amount of HFC to your account based on the current exchange rate. This can take up to one business day.
You can also receive HFC from gifts from your friends or as prizes at events in High Fidelity.

**Cash Out Your HFC**

As you acquire more HFC through Marketplace sales, prizes, or gifts, you may cash out your HFC for USD. The minimum amount of HFC that you cash out is $25 (or 2,500 HFC), and the maximum cashout value is $2000 (or 200,000 HFC) per calendar month. High Fidelity may, at its discretion, issue the occasional exception to the maximum cashout amount.

To cash out HFC:

1. Book an appointment at the Bank of High Fidelity.
2. At the time of your appointment, visit the TradingRoom domain to meet with the banker.

The transaction will go through PayPal. USD will be paid to the email account specified via the appointment booking form. Time to payment receipt will be based on PayPal rules and guidelines.

**Shop the Marketplace**

The Marketplace contains all types of items to enrich your VR experience, including avatars, buildings, apps, wearables, toys and so much more. Each item will have a cost associated with it - some items are free, while others can be purchased.

Browse our items in the Marketplace either on our website or by using the Market App.

To buy an item:

1. In Interface, pull up your tablet or HUD and go to Market.
2. Browse to the item you want to buy.
3. Click on the item and hit Get (for free items) or Buy (for purchased
You can locate all items that you purchase in the **Inventory App**.

**The Inventory App**

The Inventory app provides an interface to manage your transactions, purchases, and HFC. From the Inventory app, you can:

- View your recent purchases, sales, gifts and other transaction history
- Change avatars, put on wearables and install/update apps
- Send HFC to your friends or anyone nearby
1.2.7 Give and Receive Gifts

Just like in real life, you can give money or presents to your friends in High Fidelity. You may wish to gift an item to a friend, send money to a connection, have a VIP zone in your domain, or play a poker game with your friends.

With the Commerce API, you can also award money or items using a coupon. A coupon is a way to send HFC or items to someone at a later time, even when you are not logged in to High Fidelity. For example, you can create a coupon to award the winner of a trivia game 250 HFC, or to give someone a soda when they buy something from a vending machine.
Send HFC to Others

To send money to a connection or someone nearby:

1. In Interface, pull up your tablet or HUD and go to **Inventory**.
2. In the **Inventory** app, click ‘Send Money’.
3. Send money to one of your connections or even someone nearby in the same domain.
   - If you want to send it to one of your connections, click ‘Connections; and choose the recipient from the list.
   - If you want to send it to someone nearby, click ‘Someone Nearby’ and choose your recipient by triggering or clicking on someone nearby to select them.
4. Add the amount you wish to send. This amount should be less than or equal to your HFC balance.
5. You can add an optional public message. Click ‘Submit’.
6. A window pops up confirming that your money has been sent.
Send Purchased Items to Others

After you buy something from the Marketplace, you can give it to a connection or someone nearby. To do so:

1. In Interface, pull up your tablet or HUD and go to **Inventory**.
2. In the Inventory app, click ‘Items’.
3. Scroll to the item you’d like to give and click on the menu.
4. Select ‘Gift’.
5. Send the item to one of your connections or even someone nearby in the same domain.
   - If you want to send it to one of your connections, click ‘Connections’ and choose the recipient from the list.
   - If you want to send it to someone nearby, click ‘Someone Nearby’ and choose your recipient by triggering or clicking on someone nearby to select them.
6. You can add an optional public message. Click ‘Submit’.
7. A window pops up confirming that your item has been sent.

**Note:** When you send an item to another user, it is removed from your **Inventory**.

### Create a Coupon

You can create a coupon when you want to send money or an item to someone at a later time, even when you are not logged in to High Fidelity.

**Note:** Currently, you can only use a coupon in a script. You will not be able to redeem a coupon anywhere in Interface.

1. In Interface, pull up your tablet or HUD and go to **Inventory**.
2. Choose whether you’d like to later send HFC or an item.
   - If you want to send HFC, click ‘Send Money’.
   - If you want to send an item, click ‘Items’ and scroll to the item you’d like to give. Click on the item’s menu and choose ‘Gift’.
3. Select ‘Create Coupon’.
4. Enter an optional public message explaining the purpose of the coupon.
5. The Tablet will now display a window with the ‘Authorization ID’ and ‘Coupon ID’. Copy both these values on your computer. Click ‘Close’.
6. Include the copied values in a *script where another user receives the HFC or item.*
Example: Use a Coupon to Hold a Raffle

Say you want to pre-authorize 10 of your High Fidelity Coins to be paid out to a user who wins a raffle that you host. In this example, curl is used to perform the redemption. But you can redeem a pre-authorized transfer using any script or tool that can perform HTTP PUT requests, such as High Fidelity Interface’s request JavaScript module or a simple PHP form on a website.

1. Create a Coupon to get an ‘Authorization ID’ and ‘Coupon ID’ value pair associated with a 10-HFC Pre-Authorized Money transfer.

2. Copy and paste the ‘Authorization ID’ and ‘Coupon ID’ to a text file on your computer.

3. Click ‘Close’, then ‘I’m All Set’.

4. Hold your raffle! In this example, a user with username steve has won the raffle.

5. Use the following curl command from the command line to dispense the money authorized in (1) to username steve:

   curl -X PUT -d authorization_id="<authorization ID from 1>" -d coupon_id="<coupon ID from 1>" -d username=steve https://highfidelity.com/api/v1/commerce/redeem

See Also

- Bank and Shop
- Tutorial: Transfer Money and Items

1.3 Create

High Fidelity enables people connected by interest, community, and friendship to come together and express their creativity with each other. We invite you to personalize your own experience by creating avatars and wearables, building immersive experiences, and developing apps to make the metaverse your own.

No matter your level of expertise, High Fidelity provides the tools you need to create anything you can imagine.

Throughout this chapter, learn how to create, build, and bring to life your own VR experience:
1.3.1 Create Tools

To build and create things in High Fidelity, you need to become familiar with the tools available to you. We’ve created our own custom tools (including the Create app and Shapes app). In addition, you can use many external tools to fine-tune your creations. These tools can help you create anything from a cool avatar or a baseball hat, to a magic themed domain.

On This Page

- Create Tools
  - The Create App
  - Marketplace Item Tester
  - External Creator Tools

The Create App

Use the Create app to create any type of entity. In Interface, pull up your HUD or Tablet and go to Create to get started. With the Create app, you can:

- Add any type of entity and import externally created models and materials.
- Edit entity properties, such as its appearance, position, and behavior.
- Expose a grid that assists you with the layout and placement of entities.
- Display the Entity List, which lists all the entities in the domain. When you’re using an HMD, the entity list will be an additional tab in the Create app. In Desktop mode, the Entity List is its own window.

Note: We have received reports that Interface may crash when using a laptop and external monitor with the Create app. If you experience the crash, we recommend that you either a) disable the Nahimic service on your laptop or b) always use the Create Tools on your primary monitor.

Entity List

The Entity List shows you all entities in the local domain. You can filter by entity type and by distance from the current location.

At the top of the Entity List, you can switch between ‘Local’ and ‘World’ view. When set to ‘Local’, the position, size, and rotation settings for entities are set in reference to the parent entity. When set to ‘World’, these settings are set in reference to the world’s default position.

When you select an entity in the Entity List, you can:

- Find an entity: You can double-click an entity on the list to view it in your domain. You will see the entity with a bounding box and arrows around it.
- Lock an entity: A locked entity cannot be edited. Select an entity and click the lock icon on the top of the window.
- Change visibility: You can hide or make an entity visible. Select an entity and click the eye icon on the top of the window.
- Name an entity: Name an entity when you select it on the list.
• Delete an entity: Delete an entity by clicking on the red bin icon on the top-right corner of the window.

**Marketplace Item Tester**

Once you have created an item, you can test it prior to using it or submitting it to Marketplace. The Marketplace Item Tester reviews all kinds of content, including tablet apps, avatars, content sets, entities, and wearables. Using it, you can verify that your item works the way you expected, and that it does not have any script errors.

To use the Marketplace Item Tester:

1. In Interface, open the menu from either the menu bar (in Desktop) or your Tablet (in VR mode).
2. Go to **Menu > Settings > Developer menu** to enable the developer menu.
3. Open the **Developer** menu and go to **Marketplace Item Tester**.
4. You can load items in two different ways:
   - Click ‘Load File’ to load an item from your local computer or network. Browse to your file to open it.
   - Click ‘Load URL’ to load an item hosted in the cloud.

3. Next to the item you loaded, you will see an icon indicating the type of content. If it is incorrect (or we fail to identify it), you can change it using the dropdown list.
4. Click the icon to load your content in world.
External Creator Tools

We’ve listed some external tools you might want to use to create avatars and 3D models.

Adobe Fuse

**Note:** There are community reports where users are unable to easily open Adobe Fuse once installed. To work around this issue, open it multiple times successively until you are able to open the application.

Use Adobe Fuse to create a custom avatar. The default heads, torsos, arms, and legs in Adobe Fuse can help you start your customization.

Mixamo

Mixamo is a rigging system that will rig your avatar’s skeleton for you. You do not need any advanced knowledge of rigging to create simple animations for your avatar.

Blender

Blender is an open-source 3D modeling creation suite which supports everything from modeling and rigging, to animation and simulation. You can also use Blender to fine tune your avatar, and ensure that the materials and textures render correctly in High Fidelity.

Maya

Maya is a subscription based 3D modeling toolset that you can use to create 3D models to import into High Fidelity.
Blocks

Blocks is a 3D modeling tool you can use in VR. Blocks lets you create models easily regardless of your experience. You can create something on Blocks through Steam or download it for the VR equipment you are using.

See Also

- Entities
- Create New Entities
- Shopping the Marketplace
- Tutorial: Create an Avatar with Fuse
- Tutorial: Rig Your Avatar in Mixamo
- Tutorial: Modify Materials and Textures Using Blender

1.3.2 Avatars

When you first use High Fidelity, you will be wearing the default avatar. Your avatar is a representation of you in the metaverse. You can make your time in High Fidelity unique by creating an avatar of your own. All custom avatars must be hosted somewhere in the cloud so that High Fidelity can access it.

In This Section:

Create Your Own Avatar

There are two ways to create your own avatar: you can either create your own from scratch using 3D modeling tools such as Adobe Fuse, Mixamo, and Blender, or you can download an existing avatar from external sources such as TurboSquid or CGTrader.

Note: If you get an avatar from an external source such as TurboSquid or CGTrader, it is likely that the skeleton does not match our avatar standards. To use these avatars with High Fidelity, use the High Fidelity Avatar Exporter for Unity to correctly map the skeleton and package your avatar.

If you want to create an avatar from scratch, this page covers the steps needed to create, rig, and package your avatar.

On This Page

- Create Your Own Avatar
  - Create an Avatar from Scratch
  - Community Tools for Avatars

In This Section

Avatar Standards Guide

This document outlines the standards you should follow when creating your avatar. Your avatar uses bones to animate the character’s limbs and define the scale variable of limbs. You can add custom bones to further adjust the avatar’s
shape. Customization of your avatar can be fine-tuned using blendshapes to animate the face and scripting to define advanced behaviors.

**On This Page:**
- Glossary
- Reference Pose
- Skeleton
- Flow Bones
- Look-at Vectors
- Blendshapes
- Other Considerations

**Glossary**

As we delve deeper into creating custom avatars, we may use terminology that you are unfamiliar with. Here are some terms you might come across:

- Avatar - A virtual representation of a person or NPC.
- Mesh - The collection of 3D vertices and triangles for the avatar model. Without this, the avatar is invisible.
- Bones - A component of a skeleton that defines a “limb” such as an arm, leg, etc. Each bone may be animated as a separate limb in your avatar.
- Skeleton - A hierarchy of joints.
- Rigging - The process of creating a skeleton of the avatar model.
- Blendshapes - Variations of the topology that defines how the mesh is modified to create various “shapes”.
- FST file - The main avatar file, which contains information about the skeleton, blendshapes, FBX file and textures used by an avatar.

**Reference Pose**

For the Reference pose, use a T-Pose which complies with the specifications below. You may wish to refer to the properly configured example avatar fbx with source files.

- The character must face along the positive direction of the Z-axis.
- The arms must be spread along the X-axis. The left arm should therefore be pointing along the positive direction of the X-axis.
- The top of the character’s head must be up, in the positive direction of the Y-axis.
- The character’s hands are flat, palms facing the ground, with the thumbs parallel to the X axis.
- The character’s feet need to be perpendicular to the legs (with the toes pointing along the Z-axis as shown). The feet must not be rotated around the Y-axis (meaning the toes of the left foot should not point inward toward the right leg or outward away from the right leg).
You can download the standard High Fidelity skeleton here. This skeleton conforms to the specifications above.

**Skeleton**

The standard humanoid skeleton of your avatar should follow HumanIK Skeleton with some modifications made for Mixamo. This skeleton system will work with the input systems already in place in High Fidelity, and will allow users to use their input devices to control their avatar’s arm and finger movements (if they have any).

High Fidelity avatars should match the following standard skeletal structure. Each of these joints can be animated.
Flow Bones

The `sim` and `flow` prefixes are reserved for flow bones, such as clothing, hair and tails. These bones should not be animated by an animator. (Many thanks to Akazukin for the model Ouka Miko() used in this diagram!)

For example, consider a full cape that surrounds the avatar:

Alternatively, you can use the `flow` prefix, separating the name and joint number with an underscore. The same cape as above would look like:

Look-at Vectors

The look-at vectors are driven by the z-vector of the eye joints.

The +z axis of the eye joints should go through the center of the pupil, and should continue to do so as the eye joint is rotated.

The eye joints are defined in the FST.

Blendshapes

High Fidelity uses blendshapes to animate your avatar’s face. Blendshapes allow you to specify a new state for your avatar’s mesh, and facial positions are animated by moving between the different states of your avatar’s expressions. Blendshape behaviors are defined in your avatar’s FST file, and are added to the avatar mesh using a 3D modeling tool like Blender (Shape Keys) or Maya. Adobe’s Fuse program and Mixamo pipeline allow you to export blendshapes.
as part of your FBX, but if you are modeling an avatar from scratch, you will likely need to specify your own facial expressions.

High Fidelity avatars support a number of blendshapes for creating different facial expressions.

**Basic Blendshapes**
- EyeBlink_L: Blinking action for the left eye.
- EyeBlink_R: Blinking action for the right eye.
- JawOpen: Opening of the jaw.

**Audio Blendshapes**
These blendshapes are used when you speak. Your eyebrows are blendshapes that react to a change in volume. They will move upwards when your voice gets louder. These include:
- BrowsU_C: Center of the brow going up
- BrowsU_L: Outside corner of the left brow going up
- BrowsU_R: Outside corner of the right brow going up

Other audio blendshapes are randomly mixed when you speak. These include:
- MouthSmile_L: Left side of the mouth lifting up to a smile
- MouthSmile_R: Right side of the mouth lifting up to a smile
- LipsFunnel: Funneling of the lips, as when you say “Oh!”
- LipsUpperClose: Upper lips rolled inwards

**Eyelid Offset**
To ensure that the top of the eyelid rests on the iris, blendshapes are used to track the current position of the eye along with your head orientation.
- EyeBlink_L: Blinking action for the left eye
- EyeBlink_R: Blinking action for the right eye
- EyeOpen_L: Opening of left eye
- EyeOpen_R: Opening of right eye
- BrowsD_L: Outside corner of the left brow moving down
- BrowsD_R: Outside corner of the right brow moving down

We apply a small procedural offset to the blendshape coefficients to prevent sleepy or crazy eye lids:
- If you are looking straight ahead: The EyeBlink and EyeOpen coefficients will be 0.
- If your eyes begin to look upward: EyeBlink, EyeOpen, and BrowsU start changing in value, reaching the values of −1, 1, and 0.5 respectively at 16.3 degrees. This will have the effect of raising your lids and brows as you look upward.
- If your eyes begin to look downward: EyeBlink and EyeOpen start changing in value. EyeBlink reaches a value of 0.5 at 32 degrees. EyeOpen will reach a value of 0.5 at 27 degrees. This will have the effect of lowering your lids as you look downward.

Tweaks to your blendshapes can be made with a 3D modeling tool, or directly in your avatar’s FST file. In the FST file, blendshapes are defined with the syntax:

Here is an example of modifying your blendshapes in your FST file:
Other Considerations

File Optimization

Content creators will have limited bandwidth on servers (read small print on any unlimited host plans) so optimization is important, for both the end users and content creators. The more polygons and larger textures you use, the more bandwidth you are using from your servers per load. Optimally, keep your avatar models under 20 MB.

Textures

We recommend that you try to keep total size of all the textures per avatar below 8 MB. They should be always smaller than 1024x1024, unless all the textures are in a single file. If using multiple texture files, then smaller the better, especially if you can make the textures smaller. Remember that you can get a lot more detail through roughness and normal mapping, than just textures. It is suggested that you keep Albedo at a smaller size than your roughness for best detail through light reflection instead of color variation.

Avatar Collision Hulls

When you wear different avatars, you’ll notice that each avatar has a different collision shape or collision hull. The collision hull is the invisible area around your avatar that is used to used to detect when other avatars or entities collide with you.

Depending on the avatar’s design, the collision hulls can be very large or small. This occurs because High Fidelity analyzes the shape of the avatar’s torso (from hips to head) and tries to find the best shape that encloses the mesh. For example, if your avatar has large hips or perhaps a fully extended tail, High Fidelity thinks that the tip of the tail is part of your hips, and makes a very large collision hull. To reduce the size of the collision hull, you can add skeleton joints to your avatar’s tail.

See Also

- Create Your Own Avatar
- Find and Use an Existing Avatar
- Add Flow to Your Avatar

Create an Avatar from Scratch

The steps involved in creating your avatar are:

1. Create an avatar with 3D character modeling tool such as Adobe Fuse, Blender or Maya.
2. Rig and animate your avatar with an animation tool such as Mixamo.
3. Fine tune your avatar using a tool such as Blender or Maya.
4. Package the model in High Fidelity for use as an avatar.

Note: If you intend to upload and sell your avatar to the Marketplace, you need to set your base material color to white (some apps default to grey). This ensures that the avatar renders correctly for all users and that it will be accepted into our Marketplace.
Check out this YouTube playlist for one way to create and customize your own avatar. Here, we use Adobe Fuse to create our avatar, Mixamo to rig our avatar automatically, and Blender to adjust the rendering on our avatar. We also have written instructions on the same process:

- Create an Avatar with Fuse
- Rig Your Avatar in Mixamo
- Modify Materials and Textures with Blender

Community Tools for Avatars

As you’re creating your avatar, remember that High Fidelity is an open-source project. Many of our community members have created plug-ins, add-ons, toolkits, skeletons and more to help you create content, including avatars. Here are a few for you to play around with.

Blender Add-on by Menithal

Plugin (‘‘Project Hermes’’) is a plugin for Blender to allow for easier content creation and importing for the High Fidelity Metaverse Platform. It features:

- Material Tools: Allows for easier pipeline to apply materials to objects so that they are ready to use in High Fidelity.
- Armature Tools: Adds a skeleton that is compatible with High Fidelity and let you configure bone names for use in advanced scripts.
- Avatar Converters: Translates and fixes models and materials from MMD and Mixamo so that they work in High Fidelity.
- Export Tools: Exports avatars and scenes so that they can be used in High Fidelity.
- Import Tools: Imports primitive entities from High Fidelity so that you can make modifications to them.

Install it here: https://github.com/Menithal/Blender-Hifi-Addon

Have a project you’ve been working on that you’d like us to share? Let us know by editing this page in GitHub!

See Also

- Find and Use an Existing Avatar
- Package and Host Your Avatar

Find and Use an Existing Avatar

You can download avatars for use from external sources such as TurboSquid or CGTrader. Once you get the avatar, you will need to process it in Unity using the High Fidelity Avatar Exporter. This tool imports most avatars into Unity, maps their skeleton using Unity’s humanoid tool, and exports them as FST and FBX files to import in-world.
Avatar Guidelines

Many external sites like TurboSquid and CGTrader provide avatars that you can use. However, note that not all of the avatars you find may work in High Fidelity. To improve the chances that your downloaded avatar is compatible with High Fidelity, we’ve compiled a list of guidelines to help you “sanity check” it prior to use.

You should ensure that:

- You downloaded a real-time models (rigged for run-time, not rigged for render).
- You have the correct downloaded files
  - An FBX model for your avatar. We do not support other 3D model formats.
  - (Optional) One or more image files to give your avatar color and texture. Sometimes, these are already embedded in your FBX model and you won’t have any additional image files in your download.
- Your avatar is rigged.

**Note:** If your avatar is not rigged, you can use Mixamo to rig it. If you use Mixamo, you do not necessarily need to use Unity and the avatar exporter. Because Mixamo already uses a skeleton that we support, you can use our Avatar Packager to import your avatar into High Fidelity.

High Fidelity Avatar Exporter for Unity

High Fidelity supports only one standard type of rigging for avatars. Because many avatars do not match this skeleton, we created the High Fidelity Avatar Exporter for Unity (also known as the “avatar exporter”) to convert human-like avatars with a humanoid bone structure (body, head, and limbs). The avatar exporter also automatically packages your avatar for use in High Fidelity.

**Note:** The avatar exporter was written to improve the process of rigging and mapping the skeleton rig. This will not affect the animations or materials in your avatar. To adjust the materials, you will need to use a 3D modeling tool such as Blender or Maya and make modifications to your avatar prior to using the avatar exporter in Unity.

You will need the following to use this tool:

- Unity (Recommended versions: 2017.4.17f1 - 2018.2.12f1)
- High Fidelity (v0.77.0 or higher)
- High Fidelity Avatar Exporter for Unity (v0.4.1)

Please note that the recommended version of Unity is not the latest version. If you are using a newer version of Unity, we recommend that you apply a T-Pose to your avatar. To do so, go to the ‘Inspector’, and click ‘Pose’ near the bottom of the panel. Select ‘Enforce T-Pose’ from the drop-down. Click ‘Apply’ and ‘Done’. We recommend doing this after correcting any issues with remapping bones.
Install the Avatar Exporter

You need to install the extension for every Unity project that you have. Keep in mind, however, that you can import and export multiple avatars in a single Unity project.

1. Download the avatar exporter from High Fidelity.
2. In Unity, open the ‘Project’ window at the bottom.
3. Right-click the ‘Assets’ folder, then select Import Package > Custom Package.
4. Navigate to the avatarExporter package (with a .unitypackage extension). Click ‘Open’. You can also double-click the package on your computer to import it automatically.
5. In the ‘Importing Package’ window, review the list of files to be imported and check for conflicts with files
already in your project. If a conflict exists, save any local changes somewhere outside of your project.

6. Click ‘Import’. The package’s files are added to the Assets folder. You should now have a ‘High Fidelity’ menu in Unity.

Create an Avatar Package

1. If you don’t already have your model open in Unity, you need to import your model. Use any of the following methods:
   - Drag and drop the FBX file into the ‘Assets’ folder of your ‘Project’ window.
   - In the ‘Project’ window, right-click the ‘Assets’ folder, then select Import Package > Import New Asset. Navigate to the FBX file and click ‘Import’.
   - In Unity, open the ‘Assets’ menu, then select Import Package > Import New Asset. Navigate to the FBX file and click ‘Import’.

2. In the ‘Project’ window, select your avatar’s FBX file. In the ‘Inspector’, open ‘Rig’. For ‘Animation Type’, choose ‘Humanoid’ and then click ‘Apply’.

3. Click ‘Configure’ to investigate and tweak the mapping of your avatar.
4. All bones mapped in Unity are highlighted in green and can be selected. Check if anything is missing. Any errors will appear in red. The minimum required bones for mapping are Hips, Spine, Chest, and Head. If either of these are missing, you must manually add bones before continuing. You can do this by dragging the bones from the ‘Avatar Configuration’ panel to the ‘Inspector’ panel.
Note: Avatars in High Fidelity must have a Chest bone. If your avatar does not have a chest bone, the avatar exporter may suggest a suitable alternative from the ‘Avatar Configuration’ panel. If the exporter doesn’t suggest an alternative and Humanoid doesn’t correctly map the Chest, then you will get an error and need to manually map a bone to the Chest from ‘Avatar Configuration’.

5. If you made any changes, click ‘Done’.

6. Click on the FBX file in the ‘Assets’ manager.

7. Make sure that you have the avatar exporter installed. Open the ‘High Fidelity’ menu in the top menu bar, then select ‘Export New Avatar’.

8. Give your avatar project a name. The default project location is your local user’s Documents\High Fidelity Projects directory, which is created automatically for you. Though we recommend that you keep your avatars in this directory, you can change it to another location on your computer.

9. Click ‘Export’.

Your avatar package has been created! The File Explorer will open to your new avatar project.
**Note:** If you are using any external textures with your avatar model, copy those textures to your local user's Documents\High Fidelity Projects\avatar\<project name>\textures directory. Otherwise, they may not show up on your avatar.

---

**Test Your Avatar**

We encourage you to “spot check” your avatar in Unity before exporting it with the High Fidelity Avatar Exporter for Unity. Check for the following:

- Confirm that there are no extraneous objects attached to your model. For example, this Mech avatar has a ground blue object included in the model. All extraneous objects will be imported into High Fidelity and may affect the rendering or animation of your avatar.

- Test your bone movements. In Unity’s ‘Inspector’, open ‘Rig’. For ‘Animation Type’, choose ‘Humanoid’ and then click ‘Apply’. Go to ‘Muscles & Settings’ to test your avatar’s bone configuration and ensure that it works as expected.

After exporting your avatar package, you can also test it in High Fidelity using the Marketplace Item Tester to ensure that it works before you host it. If something doesn’t look right, tweak your avatar in Unity, then update it with the High Fidelity menu.

If everything looks good, you need to host your avatar then change your avatar to wear it.

---

**Troubleshooting Tips**

Many of the errors you will encounter describe issues with the avatar’s skeleton. These are fully documented here: Troubleshooting with the Avatar Packager. Here are some other issues you may encounter after using a downloaded avatar and using the avatar exporter:
<table>
<thead>
<tr>
<th>Issue</th>
<th>Troubleshooting Tip</th>
</tr>
</thead>
<tbody>
<tr>
<td>You receive a warning in Unity: “Character is not in T pose.”</td>
<td>Go to the ‘Inspector’, click ‘Configure’, and then select ‘Pose’ near the bottom of the panel. Select ‘Enforce T-Pose’ from the drop-down. Click ‘Apply’ and ‘Done’. We recommend doing this after correcting any issues with remapping bones.</td>
</tr>
</tbody>
</table>
| In Unity, your avatar is a solid color.                              | This suggests that the materials or shaders you are using are not supported.  
1. Click and drag your model into the ‘Scene’ window.  
2. Select all of the unsupported materials. These will be one solid color, such as pink.  
3. In the ‘Inspector’, change the ‘Shader’ to one of the ‘Standard’ options. All materials should now show up correctly. |
| Your avatar is grey.                                                 | One of the following issues could have occurred:  
1. Make sure you copied your avatar’s textures into the project’s textures folder  
2. Verify that your textures are in a format that we support (PNG, JPEG, JPG, TGA, TIF, or TIFF).  
3. If your textures are embedded in your avatar: select the FBX file, go to ‘Inspector’, and click ‘Extract Textures’. Extract your textures into your asset’s folder. You can do the same with Materials. |
| Your avatar is tied up into knots or laying down.                   | This could mean that your skeleton is not right. Re-open your avatar in Unity and run through steps 2-5 of Create an Avatar Package again. Update your project in Unity (go to Update Existing Avatar, then browse to your avatar package). If it still doesn’t work, ensure that you are testing the correct file that the avatar exporter created. |
| Your avatar’s skin doesn’t move properly with animations.           | The avatar exporter was written to improve the process of rigging and mapping the skeleton rig. This will not affect the animations in your avatar. To adjust the animations, you will need to use a 3D modeling tool such as Blender or Maya and fix the skin weighting on the avatar prior to using the avatar exporter in Unity. |

See Also

- Package and Host Your Avatar

**Package and Host Your Avatar**

At a minimum, avatars in High Fidelity must have an FBX model and an associated FST file that includes information about how your avatar looks and behaves. Together, these two files (with any optional texture or script) form an “avatar package”. There are two ways you can create an avatar package: by using the Avatar Packager in Interface or the High Fidelity Avatar Exporter for Unity in Unity.

Once you have packaged your avatar, you need to host it on the cloud so that High Fidelity can access it and correctly
render your avatar for all users.

On This Page

- Package and Host Your Avatar
  - Package Your Avatar
  - Host Your Avatar
  - Troubleshooting with the Avatar Packager

Package Your Avatar

If you’re reading this page, you likely already built your own FBX model or found and downloaded a model that you want to use in High Fidelity. Therefore, all that remains is to package your avatar and create the FST file. This file includes information about the skeleton, blendshapes, textures, and scripts used by your avatar.

We provide two ways to create an avatar package: either through Unity or through our Avatar Packager.

High Fidelity Avatar Exporter for Unity

In some cases, you will want to download an avatar from an external website and use that avatar in High Fidelity. The High Fidelity Avatar Exporter for Unity (also known as the “avatar exporter”) converts human-like avatars and packages them for use in High Fidelity.

Once you have successfully used the avatar exporter to package your avatar, you must host it somewhere on the cloud. You can host it on our servers (using the Avatar Packager), or simply copy the avatar package to your own servers.

Avatar Packager

The Avatar Packager is a tool in Interface that identifies potential errors in your avatar’s FBX model and then creates an FST file for you. Then, you can optionally use the Avatar Packager to upload your avatar to the Marketplace and host it on our servers.

To package your avatar using the Avatar Packager:

1. In Interface, go to Edit > Avatar Packager.
2. In the Avatar Packager window, click ‘New Project’.
3. In the Create Project window, fill in the following details:
   - Name: The name you want for your avatar.
   - Project Location: The folder path where your avatar’s files are stored. The Avatar Packager will create a new folder for your project at this location. The package will contain your FBX model, an FST file, and any scripts/textures in your avatar.
   - Avatar Model: The location of your avatar’s FBX model.
   - Texture Folder: If your avatar has textures in a separate folder, specify the folder location. If your avatar’s textures are embedded in the FBX, you do not need to specify anything.
4. Click ‘Create’.
At this point, you have successfully packaged your avatar. If you choose to host your avatar on the cloud with your own servers (and not use High Fidelity’s servers), you can close the Avatar Packager here and upload your FST file and FBX model to the cloud location of your choice.

**Host Your Avatar**

Before you can use a custom avatar, you must first host its FST and FBX files in a place that is publicly accessible to High Fidelity. We recommend hosting them on our own servers using the Avatar Packager, but you can also use any cloud platform including Amazon S3, Google Cloud Storage, Microsoft Azure, Dropbox, etc.

If you want to upload it to High Fidelity’s servers or sell your avatar on the Marketplace, use the Avatar Packager:

1. If this is a new avatar, first use the Avatar Packager to create an FST file. When you proceed with Step 3 below, you will upload this new project to our servers.
2. If you want to host an avatar that has already been packaged:
   1. In Interface, go to Edit > Avatar Packager.
   2. In the Avatar Packager window that opens, click ‘Open Project’.
   3. Navigate to your FST file and click ‘Open’.
3. Click ‘Upload’ to upload your avatar’s files to High Fidelity’s servers. The Avatar Packager will display any errors or warnings that you may want to resolve prior to uploading. View Troubleshooting with the Avatar Packager to determine whether a fix is required to have a usable avatar.
4. Once your avatar is uploaded to the servers, click ‘View in Inventory’ to view your custom avatar. Unless you submit it for review, your custom avatar will remain in Draft mode, and will not be visible to others. To sell your avatar, you need to submit it for review on the Marketplace.

![Inventory Screen](image)

**Note:** If you make any changes to your custom avatar, you will need to update it through the Avatar Packager to see your changes. To update, select your project and click ‘Open Project’ in step 1. Follow the same steps to update your avatar.

**Troubleshooting with the Avatar Packager**

The Avatar Packager will notify you of any errors or warnings that may affect the way your avatar looks and behaves in High Fidelity. This is a list of the errors you may encounter, along with basic instructions on how to fix your avatar. **Errors** (in red) must be fixed before you upload your avatar, while **Warnings** (in orange) may or may not affect whether your avatar will show up and behave correctly in High Fidelity.

**Note:** Many of the errors you will encounter describe issues with the avatar’s skeleton. The troubleshooting tips below will attempt to fix the errors in Unity.
However, if the bone structure of the model does not resemble a humanoid skeleton (with two legs, two arms, hips, chest, spine, and head), then it is likely not compatible with High Fidelity. You will not be able to fix these avatars in Unity alone. Instead, you will likely need advanced knowledge of building, rigging, and mapping bones in a 3D modeling tool such as Blender or Maya.

Add Flow to Your Avatar

You can simulate physics on your avatar’s hair, clothes, and body parts with a little bit of scripting and the help of High Fidelity’s *Flow technology*. The concept of “Flow” simply mimics the natural movement of hair and other attachments on your avatar. You can easily change your avatar’s flow settings using the Flow App. In order to use the Flow App, your avatar must contain flow threads.

<table>
<thead>
<tr>
<th>On This Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Add Flow to Your Avatar</td>
</tr>
<tr>
<td>− Prepare Your Avatar</td>
</tr>
<tr>
<td>− Flow App</td>
</tr>
<tr>
<td>− Resources</td>
</tr>
</tbody>
</table>

Prepare Your Avatar

In order to use the Flow technology, your avatar must contain flow threads, which are sets of connected joints in your avatar. Each flow thread must comply with the following rules:

1. The first joint is connected to an existing avatar joint, such as “Hips”.
2. Every joint in the thread should be named flow_[TYPE]_[INDEX] or sim[TYPE][INDEX], where TYPE defines a group of joints that share a common physics setup and INDEX is an integer. For example, if the thread is used to simulate a skirt, all the “skirt” joints are named flow_skirt_01, flow_skirt_02, etc.
While experimenting, feel free to use Mannequin with Hair, whose hair has flow threads already configured.

**Flow App**

Download and run the Flow app to configure your flow settings.

The Flow app will show up as an icon on your HUD or tablet. Click this icon to open the Flow app.

**Display Panel**

The Display panel affects how your avatar looks while the Flow app is open. Using these options, you can choose to view meshes and collisions to help you determine what your final flow configuration will look like.
### Joints Panel

The **Joints** panel lists all of the available flow threads, and lets you configure the behavior of your joints.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avatar</td>
<td>Hides or displays the avatar mesh.</td>
</tr>
<tr>
<td>Collisions</td>
<td>Activates or deactivates collisions.</td>
</tr>
<tr>
<td>Debug</td>
<td>Hides or displays the debug shapes.</td>
</tr>
<tr>
<td>Solid</td>
<td>Enables either a solid or wireframe display for debug shapes.</td>
</tr>
</tbody>
</table>

#### Options

**Radius**: Determines the thickness of segments and knots (needed for collision testing).

**Gravity**: Sets the how each joint will respond to gravity. A positive value will lift your joints in the air, while a negative value will respond to gravity and be pulled towards the ground. Larger values will cause the movement to happen more quickly.

**Stiffness**: Defines how susceptible the flow threads are to movement.

**Damping**: Determines how easily the bones oscillate or move around the joints.

**Inertia**: Changes the rotational velocity of the bones.

**Delta**: Controls the amount of time between each joint movement.
Collisions Panel

The **Collisions** panel controls the collision spheres that define the interactions between flow threads and the joints in your avatar. Each collision sphere is positioned using an existing avatar joint and offset: as you increase the radius of a collision sphere, you increase the distance between the flow thread and the joint. You can only have a maximum of 4 collisions defined for your avatar.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius</td>
<td>Controls the collision sphere radius.</td>
</tr>
<tr>
<td>Offset</td>
<td>Controls the collision sphere offset.</td>
</tr>
</tbody>
</table>

Output Panel

The **Output** panel displays the resulting FST data for your avatar’s flow configuration, based on what you entered in the **Joints Panel** and the **Collisions Panel**.

Copy this data directly into your avatar’s FST file to complete the flow process.
Resources

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow App</td>
<td>This app lets users easily update Flow settings without the need for scripting or advanced knowledge of avatars.</td>
<td>flowAppCpp.js</td>
</tr>
<tr>
<td>Mannequin with Hair</td>
<td>This avatar is properly rigged to work with Flow. Use this as an example for your own avatar models.</td>
<td>Mannequin with Hair</td>
</tr>
</tbody>
</table>

See Also
- Avatar Standards Guide
- Package Your Avatar
- Get Started with Scripting

Customize Avatar Animations

You can express yourself by overriding High Fidelity’s standard set of animations with your own custom animations such as dancing, juggling, or waving. Any custom animations you set up will be independent to each avatar you own and wear.

Note: We often update our process for importing custom animations to make it more user friendly. As you develop custom animations, keep in mind that you may need to modify them in the future as our custom animation support continues to improve.
Prerequisites

As we delve deeper into creating custom animations, we may use terminology that you are unfamiliar with. Here are some terms you might come across:

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avatar animations</td>
<td>Avatar animations are FBX files that define how your avatar moves. For example, turn_left.fbx is the standard animation file for your avatar turning left.</td>
</tr>
<tr>
<td>Animation roles</td>
<td>Animation roles are triggers that map to an action that an avatar can perform. For example, turnLeft is an animation role that makes your avatar turn left while standing in place. This animation role is mapped to the turn_left.fbx file. You can see this in action by pressing the left arrow key or A in Desktop mode or using your hand controllers in VR mode.</td>
</tr>
<tr>
<td>Avatar Animation JSON or Animation Graph File</td>
<td>The standard animation system blends and layers a series of animations from FBX files using a JSON data file. This JSON file is called the Animation Graph file, and it specifies exactly which animations to play and how they are blended. It also determines the order of operations, so that operations like Inverse Kinematics occur after the rest of the body has been animated by traditional means. By default, every avatar uses the same Animation Graph file.</td>
</tr>
</tbody>
</table>

Prepare Your Custom Animation

Before you replace the existing standard animations, you need to prepare your custom animation file. Use our Avatar Standards Guide and keep the following guidelines in mind:

- Animations must have standard joint names for High Fidelity.
- Animations must have standard joint orientations (y down the bone).
- Key frames must have key frames for every joint at a uniform interval of 30 frames per second.
- Locomotion animation phase has the left ankle in passing position on the first frame. Try to match this phase if you want your locomotion animation to blend with the default set.

Once you create your animation:

1. Export your animation from the external tool of your choice as an FBX file.
2. Upload your animation FBX file to a cloud server and copy the URL.

Replace Standard Animations

You can have your avatar use your custom animations by overriding the default animations. There are two different ways to do this:

- **Override Using a Script**: Write a script to override standard animations.
- **Create a Custom Avatar Animation JSON file**: You can modify this file or create a new data file.
Override Using a Script

You can write a script and use the MyAvatar namespace to override an existing animation or animation role.

We’ve listed the methods you can use to replace the standard animations on your avatar.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyAvatar.overrideAnimation</td>
<td>This method can be used to play any animation on the current avatar. It will move smoothly from the current pose to the starting frame of the custom animation. For example, if your avatar is waving, this script will stop your avatar waving and play the custom animation provided.</td>
</tr>
<tr>
<td>AnimationCache.prefetch</td>
<td>This method fetches a resource. You can use this to fetch a custom animation you’ve hosted on a cloud server. If you do not prefetch your animations before playing them, you might see a t-pose appear briefly as the animation is downloaded.</td>
</tr>
<tr>
<td>MyAvatar.restoreAnimation</td>
<td>This method stops the override function from playing any custom animation. Your avatar will go back to playing the standard animations.</td>
</tr>
</tbody>
</table>

Note: This process to replace an existing animation will take complete control of all avatar joints. Inverse Kinematics of the hands and head of HMD users will be disabled.

You can also override an existing animation role mapping:

1. Use MyAvatar.getAnimationRoles to view the list of roles for the current avatar.
2. You can replace the animation for each role with a custom animation (FBX file) using MyAvatar.overrideRoleAnimation.

We’ve listed the animation roles and their description. These are frequently updated, so we recommend using MyAvatar.getAnimationRoles to get the latest animation roles before continuing. The standard animation FBX files for these roles can be found in the High Fidelity source code repository on GitHub.

<table>
<thead>
<tr>
<th>Animation Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>rightHandGraspOpen</td>
</tr>
<tr>
<td>rightHandGraspClosed</td>
</tr>
<tr>
<td>rightIndexPointOpen</td>
</tr>
<tr>
<td>rightIndexPointClosed</td>
</tr>
<tr>
<td>rightThumbRaiseOpen</td>
</tr>
<tr>
<td>rightThumbRaiseClosed</td>
</tr>
<tr>
<td>rightIndexPointAndThumbRaiseOpen</td>
</tr>
<tr>
<td>rightIndexPointAndThumbRaiseClosed</td>
</tr>
<tr>
<td>leftHandGraspOpen</td>
</tr>
<tr>
<td>leftHandGraspClosed</td>
</tr>
<tr>
<td>leftIndexPointOpen</td>
</tr>
<tr>
<td>leftIndexPointClosed</td>
</tr>
<tr>
<td>leftThumbRaiseOpen</td>
</tr>
<tr>
<td>leftThumbRaiseClosed</td>
</tr>
<tr>
<td>leftIndexPointAndThumbRaiseOpen</td>
</tr>
<tr>
<td>leftIndexPointAndThumbRaiseClosed</td>
</tr>
<tr>
<td>idleStand</td>
</tr>
<tr>
<td>idleTalk</td>
</tr>
<tr>
<td>walkFwdShort_c</td>
</tr>
<tr>
<td>walkFwdNormal_c, walkFwdFast_c</td>
</tr>
</tbody>
</table>

1.3. Create 81
<table>
<thead>
<tr>
<th>Animation Roles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>walkFwdJog_c, walkFwdRun_c</td>
<td>Walking forward at 3.2 m/s. Walking forward at 4.5 m/s.</td>
</tr>
<tr>
<td>idleToWalkFwd, idleSettle</td>
<td>Short transition from standing idle to walking forward. Transition from walk to idle.</td>
</tr>
<tr>
<td>walkBwdShort_c</td>
<td>Walking backward at 0.6 m/s.</td>
</tr>
<tr>
<td>walkBwdFast_c, jogBwd_c, runBwd_c</td>
<td>Walking backward at 1.6 m/s. Jog backward at 2.3 m/s. Jog backward at 3.1 m/s.</td>
</tr>
<tr>
<td>turnLeft</td>
<td>Standing turning in place animation.</td>
</tr>
<tr>
<td>turnRight</td>
<td>Standing turning in place animation.</td>
</tr>
<tr>
<td>strafeLeftShortStep_c</td>
<td>Sidestep at 0.1 m/s.</td>
</tr>
<tr>
<td>strafeLeftStep_c, strafeLeftWalk_c, strafeLeftWalkFast_c, strafeLeftJog_c</td>
<td>Sidestep at 0.5 m/s. Side walk at 1.0 m/s. Side walk at 2.6 m/s. Side jog at 3.0 m/s.</td>
</tr>
<tr>
<td>strafeRightShortStep_c, strafeRightStep_c</td>
<td>Sidestep at 0.1 m/s. Sidestep at 0.5 m/s.</td>
</tr>
<tr>
<td>strafeRightWalk_c, strafeRightFast_c, strafeRightJog_c, stepLeftShort_c, stepLeft_c, strafeLeftAnim_c, stepRightShort_c, stepRight_c, stepLeft_c, strafeLeftAnim_c</td>
<td>Sidewalk at 1 m/s. Sidewalk at 2.6 m/s Side jog at 3 m/s. HMD step left at 0 m/s. HMD step left at 0.5 m/s. HMD strafe left at 2.5 m/s. HMD step right at 0 m/s. HMD step right at 0.5 m/s. HMD strafe right at 2.5 m/s.</td>
</tr>
<tr>
<td>fly</td>
<td>Flying idle.</td>
</tr>
<tr>
<td>takeoffStand</td>
<td>Standing jump takeoff.</td>
</tr>
<tr>
<td>TAKEOFFRUN</td>
<td>Running jump takeoff.</td>
</tr>
<tr>
<td>inAirStandPreApex</td>
<td>Standing jump in air on the way upward towards the jump apex.</td>
</tr>
<tr>
<td>inAirStandApex</td>
<td>Standing jump in air at apex of the jump.</td>
</tr>
<tr>
<td>inAirStandPostApex</td>
<td>Standing jump in air on the downward arc of the jump.</td>
</tr>
<tr>
<td>inAirRunPreApex</td>
<td>Running jump in air on the way upward towards the jump apex.</td>
</tr>
<tr>
<td>inAirRunApex</td>
<td>Running jump in air at apex of the jump.</td>
</tr>
<tr>
<td>inAirRunPostApex</td>
<td>Running jump in air on the downward arc of the jump.</td>
</tr>
<tr>
<td>landStandImpact</td>
<td>Standing land.</td>
</tr>
<tr>
<td>landStand</td>
<td>Standing land.</td>
</tr>
<tr>
<td>LANDRUN</td>
<td>Running land.</td>
</tr>
</tbody>
</table>

Create a Custom Avatar Animation JSON file

If you’re not comfortable using a script, you can edit or replace the existing Avatar Animation JSON file to override the standard animations.

**Note:** If you create a custom JSON file for your avatar’s animations, you will not inherit any updates made to the standard animations’ JSON file. You can perform a text merge to the latest version at any time.

The JSON file shows which animation role is mapped to which animation FBX file. You can replace standard animation FBX files with your custom animation’s FBX files. Or, you can write a new JSON file with the new mappings for each animation role.

To replace standard animations:

1. Upload your custom JSON file to a cloud server and copy the URL.
2. In Interface, pull up your HUD or Tablet and go to Avatar.
3. Click on the Settings icon on the top-right corner.

OR

1. Open your avatar’s FST file in a text editor.
2. Add your Animation Graph file’s URL.
Note: You will need to run your avatar’s files through the Avatar Packager to include the changes in your FST file.

```
animGraphUrl = "URL"
```

Examples

- Here is the current default avatar-animation.json file.
- This scoot-animation.json file replaces the idle and walk animations with a sitting pose. This example shows how you can replace some of an avatar’s default animations.

Advanced Topic: AnimNode System

The Avatar Animation JSON file contains a hierarchical tree of nodes called the AnimNode System. The AnimNode system defines how an avatar moves and is described in the Animation Graph JSON file.

The movement of an avatar is determined by a complex blend of procedural animation, pre-recorded animation clips, and inverse kinematics. This blend is calculated at every frame to ensure that the avatar body follows physics and controller input as rapidly as possible. It must handle animation for desktop users, HMD users, and users wearing a full set of HTC Vive trackers. It must be configured on the fly as sensors are added and removed from the system. It should also be open to extensions so unique animations and avatar configurations are possible. These functionalities are handled by the AnimNode system.

We’ve listed some features of the system:

- The AnimNode system is a graph of nodes.
- Some nodes are output only, such as pre-recorded animation clips.
- Other nodes produce output by processing nodes below it in the graph and blending the results together.
- By manipulating the node hierarchy, certain animation actions will occur before or after other animation actions.
- The node parameters can be dynamically changed at runtime. This flexibility is necessary to achieve good visual results.
- The system is in the default Animation Graph JSON file and is loaded during avatar initialization.

Key Concepts

The AnimNode system operates like an expression parse tree. For example the following expression: \(4 + 3 \times 7 - \frac{5}{(3 + 4)} + 6\), can be represented by the following parse tree.
This parse tree can then be evaluated at runtime to compute the actual value. In this tree, the leaf nodes are values and interior nodes are operations that combine two or more sub-trees and produce a new value. The tree is evaluated until there is a single value remaining, which should be the result of the entire expression: \( 30.2957142 \).

In the expression case, the output value of each node is a floating point number, and operations can be implemented simply by evaluating each sub-tree, and then combining them with an arithmetic operation, such as addition or multiplication.

The AnimNode system works on a similar concept. Except the value of each node contains all of the avatar’s joint translations and rotations. Leaf nodes can be static avatar poses, such as the T-pose or can be a single frame of an animation clip. Interior nodes can perform operations such as blending between two or more sub-trees, or combining the upper body of one animation with the lower body of another.

See Also

- Avatar Standards Guide
- Script
- API Reference: MyAvatar

Tutorial: Create an Avatar with Fuse

Using Adobe Fuse, you can create a custom avatar. The default heads, torsos, arms and legs in Adobe Fuse can help you start your customization.

1. Launch Adobe Fuse.
2. Pick your default body parts.

3. Click ‘Customize’. Here, you can customize the avatar. For example, you can customize how the eyebrows are shaped, facial expressions, how long the fingers are and much more.

4. Click ‘Clothing’. Here, you can choose the clothing you’d like. In addition, you can set hair and facial hair options. In the next step, you will be able to modify the materials, colors, and textures of each clothing item you
5. Click ‘Texture’. Here, you can modify your clothing choices with custom materials and colors.

6. Save your avatar.

See Also

• Tutorial: Rig Your Avatar in Mixamo
• Tutorial: Modify Materials and Textures Using Blender
• Create Your Own Avatar

**Tutorial: Rig Avatars in Mixamo**

Mixamo is a rigging system that will rig your model’s skeleton automatically for you. You do not need any advanced knowledge of rigging to create simple animations for your avatar.

In this tutorial, we will use the avatar that we created in Adobe Fuse.

1. Open your avatar in Adobe Fuse.
2. Go to **File > Animate with Mixamo**.
3. Save your avatar with a name and wait while it is exported to the auto-rigger.
4. Once your avatar is processed, Mixamo’s auto-rigger will show your animated avatar. Ensure that Facial Blendshapes are ‘Enabled’ and Skeleton LOD has been set to ‘Standard’. These settings ensure that your avatar will work properly in High Fidelity.
5. If you made changes to your settings, click ‘Update Rig’. Mixamo will re-process your avatar with these updates.

6. Click ‘Finish’ to start applying animation.

7. Once your avatar has been successfully rigged, you can download it and modify it further using a 3D software of your choice. When prompted, select Format as ‘FBX’ and Pose as ‘T-pose’.
See Also

- Tutorial: Create an Avatar with Fuse
- Tutorial: Modify Materials and Textures Using Blender
- Create Your Own Avatar

**Tutorial: Modify Materials and Textures Using Blender**

Blender is an open-source 3D modeling tool that you can use to fine tune your avatar and ensure that the materials and textures render correctly in High Fidelity.

In this tutorial, we will walk you through simple modifications you can make to your avatar using Blender. You will need to import an FBX file for your avatar. If you don’t have one, see our tutorials for Fuse and Mixamo.

1. In Blender, go to **File > Import > FBX (.fbx)**.

2. Choose your avatar’s FBX file and click ‘Import FBX’. This will open your avatar in the main view.

3. By default, you will not see the materials on your avatar. You can change your view using the toolbar at the bottom of the view.

4. To get a better view of your avatar, change the lamp settings:
• From the Outliner, click the Lamp node in Blender.
• For ‘Type of Active Data to display and edit’, choose the ‘Data’ icon.
• Change the lamp to Sun.

5. From the Outliner, open the ‘Armature’ tree and select the item you want to fine tune. You can also click on the item directly on your model.

6. Using the toolbox below, you can change the materials and texture of each body part as desired. We’ve included an example below that changes our avatar’s eyelashes. You can follow similar steps for other avatar items.

7. When you’re done changing your materials and textures, go to File > Export > FBX (.fbx).
8. Change the ‘Path Mode’ to ‘Copy’, then click the ‘Embed Textures’ icon. This makes sure that all of the textures are embedded into your model.
9. Give your avatar a unique name.
10. Click the ‘Export FBX’ button.

Now, you are ready to bring your avatar into High Fidelity.

Example: Update Eyelashes from an Image

1. Save this texture to a directory where you will remember.
2. From the Outliner, open the ‘Armature’ tree and select ‘Eyelashes’.
3. In the Toolbox below, click the ‘Materials’ icon.
4. Click the ‘+’ button next to the material list to create a new material slot.
5. Click ‘+ New’ to add a new material.
6. Rename the new material to ‘lashes’.
7. At the bottom of the Blender window, switch to ‘Edit Mode’.
8. Under the material list, click ‘Assign’.

1.3. Create
9. Scroll to the ‘Transparency’ section. Check the Transparency box and change the ‘Alpha’ value to 0.00.

10. Scroll to the ‘Specular’ section. Set the specular color to black.


12. Click ‘+ New’ to add a new texture.

13. Scroll to the Image section. Click ‘Open’ and find the lashes texture named ‘mixamo_eyelashes’ you previously downloaded. Click ‘Open Image’.

14. Check the ‘Alpha’ options in the following sections: Image, Preview, Texture, Influence.

15. Go to File > External Data > Pack All Into .blend. This will include the texture in your model.

See Also

- Tutorial: Create an Avatar with Fuse
- Tutorial: Rig Your Avatar in Mixamo
- Create Your Own Avatar
Tutorial: Add a Scattering Effect

Subsurface Scattering (SSS) is the diffuse reflection caused by light entering a material, being absorbed, scattered, and eventually exiting the material. It’s critical for surfaces like paper, marble, wax, and realistic skin. You can add this effect to your avatar in High Fidelity.

Subsurface Scattering Map and Value

Subsurface scattering (SSS) is noticeable when light passes through a thin translucent object like an avatar’s skin. It causes the diffusion of light within the shallow top layer of skin.

Scattering is composed of a “scattering value” and a “scattering map”. By setting the scattering value and map, you can influence how light scatters on the geometry of an avatar.

- The scattering value is a \([0,1]\) number which sets the amount of scatter.
- The scattering map is gray scale image that masks the areas of scatter. It is based on the avatar’s UV map.

Add Scattering to an Avatar

You can easily add scattering to an avatar by adding the value and map to the avatar’s FST file.

1. Create a custom avatar and package it using the Avatar Packager.
2. Create a scattering map for the subsurface scattering in Adobe Photoshop or its equivalent.
3. Open your avatar’s FST file in a text editor of your choice.
4. Add scattering information to the avatar’s FST file. For example:

```plaintext
materialMap = { "body_mat": { "scattering": 1.0, "scatteringMap" : "![skinMap.jpg](http://.../skinMap.jpg)" } }
```

5. Wear the avatar to observe the scattering effects in High Fidelity.

Here’s an example of the scattering effect. The left image has no scattering and the right image has scattering. You can see the red diffuse reflection along the shadow line.
Here are the scattering skin maps for this avatar.

You can also check out the following avatars that have scattering effects:

- Matthew
- Priscilla

See Also

- Create Your Own Avatar
- Package Your Avatar
- PBR Materials
1.3.3 Wearables

Wearables are objects that attach to your avatar. They can be hats, skirts, glasses, headphones and anything else that can accessorize your look in-world. You can express your individuality by creating wearables of your own and putting them on.

Before you can use a custom wearable, you must first host its FBX and JSON files in a place that is publicly accessible to High Fidelity. You can use any cloud platforms including Amazon S3, Google Cloud Storage, Microsoft Azure, Dropbox, etc. You can also upload your custom wearable to the Marketplace and sell it to others.

Build a Custom Wearable

Wearables are simply 3D models that are customized to fit on your avatar. Therefore, the first step in creating your wearable is to build one or find an existing model that you want to use.

There are a few different applications you can use to build and edit the 3D model for your wearable, including:

- Blender
- Maya
- Google Blocks
- Oculus Medium
- Tiltbrush

Some guidelines to follow when you’re building soft wearables like clothes:

- Your soft wearable should be designed to fit a particular type of avatar. Since avatars vary in size and structure, a soft wearable designed to fit one avatar may not fit another one as well.
- The soft wearable should be slightly larger than the avatar to avoid clipping. Clipping is when one 3D object goes through another 3D object without colliding.
- Your soft wearable’s shape should match the avatar’s.
- The soft wearable should have similar or the same weights as the avatar.

When building soft wearables like clothes, ensure that you are making it to fit the avatar well. Its mesh should match the avatar’s and have a higher weight.

When building your model, be sure to follow these guidelines to ensure that it is compatible with High Fidelity. Once you’re done editing your model, export the file as an FBX or OBJ file. You’ve now created your own custom model!

After hosting your wearable in the cloud or the Marketplace, put it on and adopt your new look.

See Also

- Add Your Wearable
- Get Your 3D Model
- Put On Wearables

1.3.4 Entities

To build any content in High Fidelity, whether it is an object that you interact with, or a domain to host an event, you need entities. Entities are the building blocks of the virtual world, and are used to build items as simple as boxes and squares to complex animated items such as butterflies and stereo equipment. Each entity type has its own set of properties that define its appearance and behavior.
In This Section:

**Create New Entities**

We are continually surprised by the ingenuity and creativity of the content creators in our community. You too can join this community by creating new entities. The easiest way to start building is to use High Fidelity’s Create app.

To add a new entity to your domain:

1. In Interface, pull up your tablet or HUD and go to Create.
2. In the Create app, select the type of entity you want to create.
3. Depending on the entity type, the behavior will be different:
   - For model and material entities, enter the URL that you want to import.
   - For all other entities, an entity with the default settings will appear in front of your avatar.
4. Edit the properties of your entity so that it looks and behaves like you want it to.

Types of Entities

You can choose from the following entity types:

- **MODEL** entities are 3D models that you can import in-world.
- **CUBE** entities are used to create basic box shaped entities.
- **SPHERE** entities are used to create basic sphere shaped entities.
- **LIGHT** entities are balls or beams of light that are used to add local lighting effects and spotlights to an area.
- **TEXT** entities display text against a flat plane, similar to a whiteboard or blackboard.
- **IMAGE** entities display an image from a specified URL.
- **WEB** entities display a web page from a specified URL. Only 20 web entities can run at the same time in a domain.
- **ZONE** entities are 3-dimensional areas that allow you to create a custom lighting environment.
• **PARTICLE** entities create dynamic effects that are made of many small parts, such as smoke clouds or falling water.
• **MATERIAL** entities modify the existing materials on other entities and avatars.

**See Also**
• The Create App
• Add a Material Entity
• Change How Entities Look
• Import Your 3D Model
• Tutorial: Create a Gold Spotlight
• Tutorial: Display a YouTube Channel
• Tutorial: Modify a Zone Entity’s Properties

**Change How Entities Look**

You can edit an entity’s size, color, position and rotation using your mouse or trackpad. To edit an entity, open the Create app and either select the entity or find it in the Entity List.

**Note:** You can select and edit multiple entities at once. The behavior will be different based on the type of property you’d like to set:

• Numbers: When using the slider, an offset will be applied to each of the original values. When typed in, the new value will replace the original values for the selected entities.
• All other field types (checkboxes, input fields, etc): The new value will replace the original values for the selected entities.
### Change the Color of an Entity

You can manually change the color of most entity types in the Create app. With the entity selected, click on ‘Properties’ and scroll down to the ‘Color’ settings. Here are the different color settings you can configure:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Supported Entity Type(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>The color of an entity.</td>
<td>Cube, Sphere</td>
</tr>
<tr>
<td>Light Color</td>
<td>The color of the light.</td>
<td>Light</td>
</tr>
<tr>
<td>Text Color</td>
<td>The color of the text.</td>
<td>Text</td>
</tr>
<tr>
<td>Background Color</td>
<td>The color of the background of the text.</td>
<td>Text</td>
</tr>
<tr>
<td>Key Light</td>
<td>The color of the lighting in the zone. Select ‘Inherit’ to keep the lighting of the domain, or ‘Off’ to remove all lighting.</td>
<td>Zone</td>
</tr>
<tr>
<td>Sky Box</td>
<td>The color of the ceiling in the zone. Select ‘Inherit’ to keep the color of the domain, or ‘Off’ to remove all color.</td>
<td>Zone</td>
</tr>
<tr>
<td>Ambient Light</td>
<td>Configures the amount of ambient light in the zone. Ambient light reflects on content within your zone.</td>
<td>Zone</td>
</tr>
<tr>
<td>Haze &gt; Haze Color</td>
<td>When Haze is turned on, the color of the haze in the zone.</td>
<td>Zone</td>
</tr>
<tr>
<td>Haze &gt; Glare Color</td>
<td>When Haze is turned on, the color of the glare based on the key light.</td>
<td>Zone</td>
</tr>
<tr>
<td>Bloom</td>
<td>Configures how much the bright areas of the scene glow.</td>
<td>Zone</td>
</tr>
<tr>
<td>Color &gt; Color</td>
<td>As the particle moves, sets the color of the particle at the start, middle, and finish.</td>
<td>Particle</td>
</tr>
<tr>
<td>Color &gt; Color Spread</td>
<td>The spread of color that each particle is given, resulting in a variety of colors.</td>
<td>Particle</td>
</tr>
</tbody>
</table>

### Set the Size of an Entity

For cube, sphere, text, image and web entities, you can change its size directly in your environment by selecting and dragging the small boxes inside the object.

For all entities, you can also set the size manually in the Create app. With the entity selected, click on ‘Properties’ and scroll down to the ‘Size’ settings. Here are the different size settings you can configure:
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Supported Entity Type(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Spatial</em> &gt; Local Dimensions</td>
<td>The size of the entity. If the entity is parented to an avatar, this value remains the original dimensions value while the avatar scale changes.</td>
<td>All</td>
</tr>
<tr>
<td><em>Spatial</em> &gt; Dimensions</td>
<td>The size of the entity. If the entity is parented to an avatar, this value will change if the avatar scale changes.</td>
<td>All</td>
</tr>
<tr>
<td><em>Spatial</em> &gt; Scale</td>
<td>Sets the size of the entity as a percentage of its original size.</td>
<td>All</td>
</tr>
<tr>
<td>Fall-Off Radius</td>
<td>The distance from the light’s center where the intensity is reduced.</td>
<td>Light</td>
</tr>
<tr>
<td>Spotlight Cut-Off</td>
<td>Affects the size of the spotlight’s beam; the higher the value, the larger the beam.</td>
<td>Light</td>
</tr>
<tr>
<td>Line Height</td>
<td>The height of each line of text.</td>
<td>Text</td>
</tr>
<tr>
<td><em>Emit</em> &gt; Dimensions</td>
<td>The outer limit radius that particles can be emitted from.</td>
<td>Particle</td>
</tr>
<tr>
<td><em>Emit</em> &gt; Radius Start</td>
<td>The inner limit radius that particles can start emitting from.</td>
<td>Particle</td>
</tr>
<tr>
<td><em>Size</em> &gt; Size</td>
<td>The size of each individual particle in the entity.</td>
<td>Particle</td>
</tr>
<tr>
<td><em>Size</em> &gt; Size Spread</td>
<td>How far apart a particle is from other particles.</td>
<td>Particle</td>
</tr>
</tbody>
</table>

**Rotate an Entity**

All entities can be rotated directly in your environment by selecting and dragging the circles around the object.

You can also set the rotation manually in the **Create** app. With the entity selected, click on ‘Properties’ and scroll down to the ‘Rotation’ settings. Here are the different rotation settings you can configure:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Supported Entity Type(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Spatial</em> &gt; Local Rotation</td>
<td>The orientation of the entity relative to its parent.</td>
<td>All</td>
</tr>
<tr>
<td><em>Spatial</em> &gt; Rotation</td>
<td>The orientation of the entity with respect to world coordinates.</td>
<td>All</td>
</tr>
</tbody>
</table>

**Note:** You can switch between ‘Local’ and ‘World’ using the keyboard shortcut T.

**Move an Entity**

All entities can be moved directly in your environment by selecting and dragging the object to the correct location. Alternatively, you can use the arrows around the object to move it in only one direction.

You can also set the position manually in the **Create** app. With the entity selected, click on ‘Properties’ and scroll down to the ‘Position’ settings. Here are the different position settings you can configure:
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
<th>Supported Type(s)</th>
<th>Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spatial &gt; Local Position</td>
<td>The position of the entity relative to its parent.</td>
<td>All</td>
<td></td>
</tr>
<tr>
<td>Spatial &gt; Position</td>
<td>The position of the entity with respect to world coordinates.</td>
<td>All</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** You can switch between ‘Local’ and ‘World’ using the keyboard shortcut T.

**See Also**
- *Interact with Your Environment*
- *The Create App*
- *Define an Entity’s Behavior*

**Define an Entity’s Behavior**

An entity’s behavior controls its interactions with other entities and avatars in High Fidelity. Can an entity be grabbed, does it collide with other entities and avatars, or can a change in the domain’s gravity affect it? You can check and change an entity’s behavior by editing its properties.

**Note:** You can select and edit multiple entities at once. The behavior will be different based on the type of property you’d like to set:

- **Numbers:** When using the slider, an offset will be applied to each of the original values. When typed in, the new value will replace the original values for the selected entities.
- **All other field types (checkboxes, input fields, etc):** The new value will replace the original values for the selected entities.

**On This Page**

- *Define an Entity’s Behavior*
  - *Set an Entity to Respond to Physics*
  - *Set Entity Behavior on Collision*
  - *Make an Entity Grabbable*
  - *Set an Entity to Trigger Scripts*
  - *Make an Entity Cloneable*
  - *Set an Entity to Cast a Shadow*

**Set an Entity to Respond to Physics**

If you want an entity to *respond to physics* or other entities and avatars, you need to make it dynamic. This allows a box to respond to gravity or a ball to bounce when it hits the floor. If an entity is not dynamic, it is static and has no
gravity and no velocity. It can only be moved by a user.

To make an entity dynamic:

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. Select your entity on the ‘Entity List’ window or just click on it.
3. Go to the ‘Properties’ tab, scroll down to ‘Collision’ and check ‘Dynamic’.

### Set Entity Behavior on Collision

When an entity has no collision properties, it moves through other entities and avatars like it’s not a solid object. For an entity to collide when it comes in contact with another entity or avatar, its collision properties need to be changed. With the entity selected, click on ‘Properties’ and scroll down to the ‘Collision’ settings.

Here are the different collision settings you can configure:

<table>
<thead>
<tr>
<th>Collides With</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static Entities</td>
<td>Your entity will collide with static entities. Keep in mind that if your entity is a static entity, it will not collide with another static entity. Only a dynamic entity can collide with a static entity.</td>
</tr>
<tr>
<td>Kinematic Entities</td>
<td>Your entity will collide with kinematic entities. Kinematic entities have velocity, but are not dynamic. Their behavior is controlled by an external script.</td>
</tr>
<tr>
<td>Dynamic Entities</td>
<td>Your entity will collide with other dynamic entities.</td>
</tr>
<tr>
<td>My avatar</td>
<td>Your entity will collide with your avatar.</td>
</tr>
<tr>
<td>Other avatars</td>
<td>Your entity will collide with other user’s avatars.</td>
</tr>
<tr>
<td>Collision Sound</td>
<td>You can make your entity emit a sound whenever it collides with other entities.</td>
</tr>
</tbody>
</table>

### Make an Entity Grabbable

Your entity’s grab properties determine how it behaves when you or another user interacts with it. By default, ‘Grabbable’ and ‘Follow Controller’ are checked.

Here are the different grab settings you can configure:

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grabbable</td>
<td>You or other users can grab this entity.</td>
</tr>
<tr>
<td>Follow Controller</td>
<td>Your entity will follow the movements of your hand controller instead of your avatar’s hands. If your avatar’s arms are shorter than your real arms, your entity will be grabbed where the controller is (at a distance from your avatar’s hands).</td>
</tr>
</tbody>
</table>

### Set an Entity to Trigger Scripts

If you want your entity to trigger a script when you, other users, or other entities come in contact with it, you can do so by editing its properties.
Here are the different trigger settings you can configure:

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger-able</td>
<td>Your entity can trigger a script. For instance, you can trigger a cube entity to run a script asking for a tip every time you click it.</td>
</tr>
</tbody>
</table>

### Make an Entity Cloneable

You can clone your entity to create other entities with the same properties as yours. While creating clones, you can set how long they’ll exist, how many clones you can create, how the clone responds to physics, and if the clone is an avatar entity.

Here are the different clone settings you can configure:

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clone-able</td>
<td>Your entity can be cloned. You can change the ‘Clone’ settings to manipulate your cloned entity’s behavior.</td>
</tr>
<tr>
<td>Clone-able &gt; Clone Lifetime</td>
<td>Select this option to set how long (in seconds) your clone will exist.</td>
</tr>
<tr>
<td>Clone-able &gt; Clone Limit</td>
<td>Select to set a limit to how many clones you can create. If you don’t want to have a limit, set the value to 0.</td>
</tr>
<tr>
<td>Clone-able &gt; Clone Dynamic</td>
<td>Select to make the clone entity a dynamic entity.</td>
</tr>
<tr>
<td>Clone-able &gt; Clone Avatar Entity</td>
<td>Select to specify if a cloned entity is created as an avatar entity. An avatar entity doesn’t exist in the Entity Server. Instead, it is specific to a user’s Interface client. For instance, say a user comes to visit the coffee shop in your domain. The user grabs a coffee cup that’s been cloned. Once the user is done visiting, the cloned entity leaves with their avatar, ensuring there isn’t any clutter left behind. This feature ensures that your entity is cloned locally for each avatar.</td>
</tr>
</tbody>
</table>

**Note:** A user does not need create permissions to clone an entity or edit an unlocked entity.

To make entities cloneable in your domain (this can only be done with unlocked entities):

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. Select the entity of your choice in the ‘Entity List’ window.
3. Go to the ‘Properties’ tab, scroll down, and check ‘Cloneable’.

Keep in mind that any user can now clone the entities that are cloneable. If you don’t want any users to clone your entity or any entities in your domain, you can either:
1. Lock any entities you don’t want cloned.

2. Deny lock unlock permissions to the users of your choice.

OR

1. Set entity filters to prevent users from editing entities in your domain.

**Set an Entity to Cast a Shadow**

You can make your entity behave like a real world object by making it cast a shadow on other entities and avatars. In High Fidelity, entities cast shadows only from the key light, not from the light entities. The key light is a parallel source of light, like the sun.

Here are the different shadow settings you can configure:

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cast Shadow</td>
<td>Your entity will cast a shadow on other objects and avatars.</td>
</tr>
</tbody>
</table>

**See Also**

- [Apply Physics to Entities](#)
- [Define Interactions with Avatars](#)
- [Add Sound to Entities](#)
- [Interact with Your Environment](#)

**Apply Physics to Entities**

Your High Fidelity VR experience is made realistic with the help of a physics engine. High Fidelity uses this engine to simulate an object’s behavior according to the Newtonian laws of physics. For example, if you hit a ball with a bat in High Fidelity, the physics engine computes these movements and makes the ball spin away from the bat after collision. You can modify an entity’s physics behavior using the Create app.

**On This Page:**

- [Apply Physics to an Entity](#)
- [Change an Entity’s Velocity](#)
- [Set How a Moving Entity Slows Down](#)
- [Set an Entity’s Friction and Bounciness](#)
- [Set an Entity’s Density](#)
- [Set How an Entity Moves in a Gravitational Field](#)

**Apply Physics to an Entity**

To apply physics properties to an entity:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select or add any entity of your choice.
3. In the ‘Properties’ tab, scroll down to the **Physics** section. When you first create an entity, the physics properties are set to the default values you can see in the image below.

![Physics Section](image.png)

**Change an Entity’s Velocity**

Velocity is the speed of an object in a certain direction. All entities that have a position and orientation in High Fidelity will have linear and angular velocity. These velocities might be zero, but they still exist.

**Linear Velocity**

**Type:** 3D Vector  
**Unit:** meters/second  
**Default Value:** (0,0,0)

You can choose to make an entity move in a specified direction by changing its linear velocity. The direction is determined using the x, y, or z coordinates in a 3D Cartesian coordinate system. The 3D Cartesian coordinate system helps you determine the position of your entity in space. Every time your entity moves, its x, y, and z coordinates change to show you the new position. To change an entity’s linear velocity:

1. In the ‘Properties’ tab, scroll down to ‘Linear Velocity’ property. The default value is 0.0000.
2. Say you want to move a cube entity upwards in a straight line. Change the Y value for linear velocity to 0.1000 and see your cube start moving.
3. If you want your cube to change direction, change the x and z values to 0.1000.

**Angular Velocity**

**Type:** 3D Vector  
**Unit:** radians/second  
**Default Value:** (0,0,0)

Angular velocity is the speed at which an object is rotating in a certain direction. It is measured in radians/second. To change an entity’s angular velocity:

1. In the ‘Properties’ tab, scroll down to the ‘Angular Velocity’ property. The default value is 0.0000.
2. Change the X value to see your cube entity start rotating around an axis.
3. If you want your cube entity to change its angular velocity direction, change the Y and Z values.

Set How a Moving Entity Slows Down

Type: Scalar  Range: 0 - 1  Default Value: 0.00

In High Fidelity, damping represents how much of an entity’s linear or angular velocity is lost over time. All moving objects we see in the real world experience some friction with air, reducing their velocities over time. Damping is used to approximate this effect of the real world in High Fidelity. So if the damping of an object is 0.00, it will not lose any velocity and it will not slow down. If the damping of an object is 1.00, it will lose all its velocity and stop immediately. If you want to throw a ball and have it slow down over time, you can add a damping value to do so.

To set the linear damping of an object:
1. In the ‘Properties’ tab, scroll down to the ‘Linear Damping’ property. The default value is 0.00.
2. Change the linear velocity of a cube to any value.
3. Change the ‘Linear Damping’ value to 1.00 to make the cube stop moving. You can change this value to anything between 0.00 and 1.00 to make an entity slow down over time.

To set the angular damping of an object:
1. In the ‘Properties’ tab, scroll down to the ‘Angular Damping’ property. The default value is 0.00.
2. Change the angular velocity of a cube to any value.
3. Change the ‘Angular Damping’ value to 1.00 to make the cube stop moving. You can change this value to anything between 0.00 and 1.00 to make an entity slow down over time.

Set an Entity’s Friction and Bounciness

Range: 0 - 1  Default Value: 0.5000

When a dynamic entity collides with another entity, it can react in a number of ways depending on its physics properties. The values you set for friction and bounciness determine this reaction. By default, both values are 0.5000.

Friction is a measure of how slippery an object is. When an entity with low friction collides against another object, it will slide a good distance before coming to a stop. On the other hand, an entity with high friction will slow down much faster. To set the friction of an entity:
1. In the ‘Properties’ tab, scroll down to the ‘Friction’ property.
2. Change the value to anything between 0.0000 and 1.0000. An entity with a friction of 0.0000 will be very slippery, while an entity with a friction of 1.0000 will have a coarse or sticky surface.

Bounciness is the energy an entity conserves during collision. For example, a ball will conserve more energy and bounce more than a heavy cube. To set the bounciness:
1. In the ‘Properties’ tab, scroll down to the ‘Bounciness’ property.
2. Change the value to anything between 0.0000 and 1.0000. An entity with a bounciness of 0.0000 will conserve no energy, while an entity with a bounciness of 1.0000 will conserve all of its energy.
Set an Entity’s Density

**Type:** Scalar  **Unit:** kg/meter³  **Range:** 100 - 10000  **Default Value:** 1000.0000

An entity’s density is the ratio of its mass to its volume. For example, an entity with low density is made of light materials such as wood, while an entity with high density is made of dense materials such as iron.

In High Fidelity, the maximum (10000) and minimum (100) values of density were chosen for stability. It’s difficult to perform stable physics calculations between objects of very disparate masses (such as a light feather and an iron ball). To help keep the environment stable, we picked conservative density limits.

To change this value, scroll down to the ‘Density’ property in the ‘Properties’ tab. Change it to the value of your choice.

Set How an Entity Moves in a Gravitational Field

**Type:** 3D Vector  **Unit:** meters/second²  **Default Value:** (0,0,0)

In the Create app, ‘Gravity’ is the acceleration of the entity, as if it were in a uniform gravitational field. This property controls how an entity behaves when you change the gravity of a domain. For example, if a ball is floating in zero gravity, it will float downwards when you increase gravity downwards.

To change this value, scroll down to the ‘Gravity’ property in the ‘Properties’ tab. Change it to the value of your choice.

See Also

- Define an Entity’s Behavior
- Interact with Your Environment
- Define Interactions with Avatars

Add a Material Entity

You can add a material entity to an object in your domain. A material entity contains specific material data that determines the texture and shading of an object. For example, if you want to create a castle in your domain, you need your walls to look like they’re made of rough gray stone. You can do this by adding a castle wall material entity to your walls.

Before adding a material entity, make sure you have created a material using the PBR Materials Guide.

On This Page:

- Generate a Material Entity
- Add a Material Entity
  - Use the Material Entity JSON File
  - Use the materialData Field

Generate a Material Entity

To add a material to your object in High Fidelity, you need to specify the material data in a JSON file or add the material directly into the Create app.

This is what the JSON file for a sample castle wall material looks like:
This file contains all related material data, such as the color, roughness, and other texture and shading information. Note that you can edit this information programmatically with the Material EntityType in our API, and define its properties using EntityProperties-Material.

Add a Material Entity

Use the Material Entity JSON File

Once you have your material entity JSON file, you can add it to an object in High Fidelity. Let's add the castle wall material to a box entity in your domain.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create a wall. Click the ‘Cube’ icon to add a box entity and change the dimensions to make it resemble a wall.
3. Go to the Create tab and click on the ‘Material’ icon to add a material entity. Enter the material’s JSON file URL when prompted. You will see the material entity represented as a small sphere.
4. Click and select the wall. Go to the ‘Properties’ tab and copy the parent ID under the ‘Name’ box.
5. Click and select the material entity. Go to the ‘Properties’ tab and paste the copied parent ID in the ‘Parent’ box.
   You will see the material applied to the wall. In this step, you are parenting or applying a material to an entity.

Use the materialData Field

To add a material entity directly into the Create Tools app:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create a wall. Click the ‘Cube’ icon to add a box entity and change the dimensions to make it resemble a wall.
3. Go to the Create tab and click on the ‘Material’ icon to add a material entity.
4. Enter materialData when you’re prompted for a ‘Material URL’.
5. Click and select the wall. Go to the ‘Properties’ tab and copy the parent ID under the ‘Name’ box.
6. Click and select the material entity. Go to the ‘Properties’ tab and paste the copied parent ID in the ‘Parent’ box.
   In this step, you are parenting or applying a material to an entity.
7. Scroll down to the ‘Material Data’ field. Click ‘Clear Material Data’ and then paste the following JSON data:

```json
{
  "materialVersion": 1,
  "materials": [
    {
      "name": "CastleWall",
      "model": "hifi_pbr",
      "albedo": [1, 1, 1],
MaterialEntities/MatOne/CastleWall/CastleWall_Base_Color.png",
MaterialEntities/MatOne/CastleWall/CCastleWall_Roughness.png",
MaterialEntities/MatOne/CastleWall/CastleWall_Normal.png"
    }
  ]
}
```

See Also

- Create New Entities
- PBR Materials Guide
- Introduction to JSON

Add Sound to Entities

Entities have the ability to add a collision sound, so that the entity will emit a sound every time it comes in contact, or collides, with another entity.

To add a sound:

1. In Interface, pull up your Tablet or HUD and go to **Create**.
2. Select the entity.
3. In the **Create** app, click on ‘Properties’ and scroll down to the ‘Collision’ settings.
4. Check the box for ‘Collides’, then enter the URL of the audio file for ‘Collision Sound’.

Tutorial: Create a Bouncing Ball

In this example, we will walk through the steps to create a bouncing ball that emits a sound every time it hits a wall.

1. Create a wall to bounce your ball off of:
   1. Add a cube entity to your domain.
   2. Resize the entity to approximately 10m wide, 10m high and 1m deep (X:10, Y:10, Z:1).
2. Create a ball by adding a sphere entity to your domain. Optionally, change the color of your ball, so that it is different than your wall.
3. In the **Create** app, click on ‘Properties’ and scroll down to the ‘Collision’ settings. Check the box for ‘Collides’ and ‘Dynamic’.
4. For ‘Collision Sound’, enter the URL of your sound file. The sound must be a .wav file, uncompressed, 48Khz, 16 bit. The URL can be either a web address, or an ATP reference to the assets on this domain server.

5. Scroll down to the ‘Physics’ settings. Set the ‘Gravity’ for Y to -5. This will cause your ball to fall a little more slowly than things on earth (use -9.8 if you want that). Gravity is in units of m/s².

As soon as you click off the ball, the gravity will cause the ball to fall downwards. When it hits the floor, it should stop or bounce a little and the sound should play.

See Also

- Set Entity Behavior on Collision
- Set How an Entity Moves in a Gravitational Field

Define Interactions with Avatars

In real life, you interact with objects on a daily basis. In High Fidelity, your avatar can also interact with objects (entities) in the metaverse. There are a number of ways you can define the interactions you have with objects: you can write scripts to change the properties of an entity. You can create entities that are unique to your avatar (we call these “avatar entities”). And don’t forget that you can set an entity’s behavior and collision properties, so that objects are grabbable, triggerable, or dynamic.

On This Page:

- Control Interactions with Entities using Scripts
- Create Avatar Entities

Control Interactions with Entities using Scripts

When your avatar comes in contact with an entity, you can control its interactions with the entity using simple scripts. An interaction between an avatar and an entity occurs when the avatar comes in contact with an entity’s bounding box. The bounding box (or bounds) is the frame that is around the outside of the entity. In the case of a cube, the bounds are the exact size and shape as the entity. However, in the case of more complex objects, the bounds might be larger than the actual mesh model.

There are two methods you can use to script these interactions. Entities.enterEntity() occurs when the avatar contacts the bounding box, not the model itself. Similarly, Entities.leaveEntity() occurs when the avatar exits the bounding box.

Tutorial: Enter a Box to Change Its Color

The following example walks you through the process of creating a simple entity, and scripting an interaction between the entity and your avatar. When your avatar comes in contact with the box, the box will change color. When your avatar moves away, the box will return to its original color.
1. Create a cube entity.

2. The following script changes the color of the cube as you approach (yellow) or leave (pink) its bounding box. Save it to a file called `interactions-example.js`.

   ```javascript
   (function() {
       this.enterEntity = function(entityID) {
           Entities.editEntity(entityID, {
               color: { red: 255, green: 64, blue: 64 },
           });
       },
       this.leaveEntity = function(entityID) {
           Entities.editEntity(entityID, {
               color: { red: 255, green: 190, blue: 20 }
           });
       },
   });
   ```

3. In the **Create** app, click on ‘Properties’ and scroll down to the ‘Script’ settings. Enter the path and file name to `interactions-example.js` that you created above. Press ‘Enter’.

A full range of entity parameters are controllable with these functions. Entities can be used as invisible sensors or expanded to cover an entire building with the functions running while you are inside, and stop when you walk out.

**Create Avatar Entities**

Your avatar will also interact with avatar entities. Avatar entities are entities that are attached to your avatar, and unlike domain entities, they travel with your avatar when you go to other domains. Examples of avatar entities include wearables such as glasses or hats.

Avatar entities live on the **Avatar Mixer**, so they are connected to (and move with) your avatar. We’ve listed the ways you can create avatar entities with some examples:

1. **Create a wearable:** All wearables are avatar entities.

2. **Clone as an avatar entity:** When you clone an entity as an avatar entity, you make a copy of the entity and attach it to your avatar. Every copy of that entity will now leave with the avatar when they leave the domain. For example, if you have a coffee shop in your domain, you can set all coffee cups to be cloned as avatar entities. Any user who clones a coffee cup will take the avatar entity with them when they exit the domain. You can keep your domain free of clutter using this property.
3. Add an avatar entity using a script: You can add an avatar entity using scripts. For example, you can create a script to have a pet (avatar entity) follow you around as you explore High Fidelity.

This example script adds a cube as an avatar entity to your domain.

```javascript
var entityID = Entities.addEntity({
    type: "Box",
    position: Vec3.sum(MyAvatar.position, Quat.getFront(MyAvatar.orientation)),
    "avatar";
});
```

See Also

- Define an Entity’s Behavior
- Interact with Your Environment
- API Reference: Entities
- Get Started with Scripting

Prioritize Zones Within Your Domain

Whenever someone enters your domain, they must load all domain content, including avatars, models, textures and external resources that are used in the domain. Many of these resources are not optimized and can take a while to load, especially in large domains or events with lots of avatars. In these situations, you can prioritize a zone so that all avatars within that zone move smoothly to all observers. The avatars within the zone are known as “heroes”. Examples of heroes include a keynote speaker at an event or performers at a concert.

Avatar Hero Zones

All avatars in hero zones are allocated additional bandwidth, allowing them to load completely and move smoothly within the zone. You can set the amount of bandwidth that is dedicated to your hero zones in your domain settings. Go to http://localhost:40100/settings and scroll to Avatar Mixer > Advanced Settings > Hero Bandwidth to set the bandwidth.

To create an avatar hero zone:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Zone’ icon to create a zone entity. You’ll see a wireframe shape showing the zone’s bounding box.
3. If you are unable to view the zone’s bounding box, go to Edit > Show Zones in Create mode and select the option. Your zone should now be visible.
4. Go to the ‘Properties’ tab. Here, we recommend that you add a name to your zone.
5. For ‘Avatar Priority’, choose the desired setting:
   - Inherit: If the zone is nested within another zone, then it will inherit the priority of the parent zone. For single zones that are not nested, this option defaults to “Crowd”.
   - Crowd: The zone will not be prioritized in any way.
   - Hero: The zone will be given preferential treatment, and additional bandwidth will be given to heroes within the zone. This ensures that all avatars within the zone will load completely and move smoothly.
6. Like all entities, you can customize your zone by changing the lighting, size, or shape of the zone.

At any point, you can change the properties of your hero zone. These changes will take place immediately, and the zone will be updated to match your new settings.
Entity Tutorials

Walk through our online tutorials to get a deeper understanding of entities and their role in the metaverse.

Tutorial: Create a Gold Spotlight

In this tutorial, you will learn how light entities work by creating a gold spotlight that shines down a wall. A light entity behaves like a ball or a beam of light, and is used to add local lighting effects or spotlights to an area.

On This Page:
  • Prerequisites
  • Create a Wall to Shine the Light On
  • Create the Gold Spotlight

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:
  • Create New Entities
  • Change How Entities Look

Create a Wall to Shine the Light On

Your gold spotlight needs a surface to shine on. Create this surface or wall in the Create app:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Cube’ icon to create a cube entity.
3. Go to the ‘Properties’ tab and make the following changes:
   1. Change the color of the cube to teal (Red = ‘0’, Green = ‘128’, Blue = ‘128’).
   2. Change the dimensions of the cube to make it bigger and look more like a wall. We’ve used the following local dimensions:
      • X = ‘0.1300’
      • Y = ‘2.4000’
      • Z = ‘3.2000’
You’ve made your wall!

Create the Gold Spotlight

Create and edit the light entity to get a soft gold spotlight.

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. Click the ‘Light’ icon to create the light entity.
3. Grab and move the entity to the top and center of the wall. You’ll notice there is a white light shining on the wall. The light is small because of low brightness. This type of light is a point light and it emanates in all directions equally. It is meant for general area lighting as it has a bright point in the middle and fades as it radiates out.
4. Go to the ‘Properties’ tab and modify the light entity to make a gold spotlight:
   1. Change the color of the light to gold (**R** = ‘255’, **G** = ‘215’, **B** = ‘0’).
   2. You can make the light entity brighter by changing its intensity. Change the ‘Intensity’ to ‘100’. You’ll see that the light is now covering a larger area and is much brighter.
   3. You can modify the light entity’s ‘Fall-off Radius’ so that the it dims gradually towards the edges. The ‘Fall-off Radius’ defines the shape of the light curve of a light. A larger radius will simulate a larger light, which will “fall-off”, or dim, more gradually. It is the distance from the light at which the intensity is reduced by ‘25%’. Change this value to ‘0.5’.
   4. Select the ‘Spotlight’ checkbox to convert the light entity to a spotlight.
   5. Change the ‘Spotlight Cut-off’ to ‘50’. This property determines the radius of the spotlight. A higher cut-off value corresponds with a larger spotlight radius. You should see the beam tighten get smaller.
   6. Change the ‘Spotlight Exponent’ to ‘5’. This property affects the softness of the beam. You should see the edge of the beam soften.
   7. Rotate the spotlight so that it’s facing down the wall by changing the ‘Local Rotation’s’ **X** value to ‘-90.0000’. A spotlight positioned like this can be used for a soft lighting effect over paintings or wall hangings in your world.
Congratulations! You’ve created a soft gold spotlight! You can experiment with different spotlight exponents, cutoff values, and intensity combinations for varied effects.

See Also

- Create New Entities
- Change How Entities Look
- Creator Tools
Tutorial: Create a Smoke Fountain

In this tutorial, you will learn how particle entities work by creating a smoke fountain that emits multiple colors. Particle entities are used to create effects made up of many small particles, such as smoke, confetti, or falling leaves.

On This Page

• Tutorial: Create a Smoke Fountain
  – Prerequisites
  – Create a Smoke Fountain

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

• Create New Entities
• Change How Entities Look

Create a Smoke Fountain

Particle entities are used to create effects that are made up of smaller parts such as smoke, confetti, or falling leaves. The entity’s effect and appearance is defined by its texture. The default texture is a wispy smoke texture, but you can replace this texture with your own to create your desired effect.

To create your smoke fountain using a particle entity:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Particle’ icon to create the particle entity. By default, the particle entity emits smoke.
3. Go to the ‘Properties’ tab, and set the following values:
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifespan</td>
<td>1.20</td>
<td>This property defines how long each particle lives, in seconds.</td>
</tr>
<tr>
<td>Max Particles</td>
<td>356</td>
<td>Max particles defines how many particles are rendered at one time. Older particles, whose lifespans are completed, are swapped out for newer ones. Increase this value to increase the number of particles for your effect.</td>
</tr>
<tr>
<td>Emit Rate</td>
<td>524</td>
<td>The emit rate defines how many particles are emitted per second.</td>
</tr>
<tr>
<td>Speed Spread</td>
<td>2.10</td>
<td>This defines the range of speeds the particles will emit, which changes as per the Emit Speed. For example, if Emit Speed is 5 and Speed Spread is 1, the particles will have speeds that range from 4 to 6.</td>
</tr>
<tr>
<td>Emit Dimensions</td>
<td>x = 0.60, y = 10.00, z = 0.30</td>
<td>Particles are emitted from a radius. This property defines the outer limit of that radius.</td>
</tr>
<tr>
<td>Size</td>
<td>Start = 0.40, Middle = 0.80, Finish = 0.80</td>
<td>This property determines the size of each particle and how it changes from emission to end of life. Here, it starts at 0.4 and when its life is completed, the particle has increased in size to 0.8.</td>
</tr>
<tr>
<td>Color</td>
<td>Start = #000000, Middle = #FFFFFF, Finish = #000000</td>
<td>This determines the colors the particles will emit. Start, middle, and finish define the color spectrum to be emitted.</td>
</tr>
<tr>
<td>Color Spread</td>
<td>#FFFFFF</td>
<td>This determines the color spectrum of the particles.</td>
</tr>
<tr>
<td>Emit Acceleration</td>
<td>x = 0, y = -9.09, z = 0</td>
<td>This is the acceleration of each particle during its lifetime.</td>
</tr>
<tr>
<td>Spin Spread</td>
<td>147.40</td>
<td>The spread in spins for the particle entity. Changing this value results in a variety of spins for different particles.</td>
</tr>
<tr>
<td>Horizontal Angle</td>
<td>Start = 0, Finish = 180</td>
<td>This is the angle range in degrees at which the particles are emitted along the z axis and rotated around on the y axis.</td>
</tr>
<tr>
<td>Vertical angle</td>
<td>Start = 150, Finish = 180</td>
<td>This is the angle range in degrees at which the particles are emitted along the x axis and rotated around on the z axis.</td>
</tr>
</tbody>
</table>

Congratulations! You’ve created a multi-colored smoke fountain! You can experiment with different settings to simulate particle movement, such as a waterfall, confetti gun, or falling leaves.

**See Also**

- [Create New Entities](#)
- [Change How Entities Look](#)
- [Create Tools](#)
Tutorial: Modify a Zone Entity

A zone entity is a 3D area where you can create custom lighting environments. You can define zone boundaries using shapes and customize the zone’s light properties such as its intensity, direction, and color to create different effects.

The mini tutorials on this page show you how zone entities work and how you can edit their properties to create areas with different lighting effects.

On This Page:
- Prerequisites
- Create a Zone Entity
- Create Nested Zones with Different Lighting
- Change a Zone’s Shape
- Add a Skybox to a Zone
- Add Ambient Light to a Zone

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:
- Create New Entities
- Change How Entities Look

Create a Zone Entity

To create a zone entity:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Zone’ icon to create a zone entity. You’ll see a wireframe shape showing the zone’s bounding box.
3. If you are unable to view the zone’s bounding box, go to Edit > Show Zones in Create mode and select the option. Your zone should now be visible.
4. Go to the ‘Properties’ tab, and add a name ‘Zone-1’ for your zone.

Create Nested Zones with Different Lighting

Each zone entity you create can have different properties. When your avatar moves through different zones, you will experience each zone’s properties. In the case of nested or overlapping zones, you will experience the properties of the smallest zone you are currently inside.

You can understand how an avatar experiences lighting in a zone using this mini tutorial:

1. Create Two Zone Entities
2. Nest One Zone Inside the Other
3. Add Different Lighting Effects to Each Zone
Create Two Zone Entities

Follow the steps to create a zone entity to create two zone entities named ‘Zone-1’ and ‘Zone-2’.

Nest One Zone Inside the Other

To nest a zone, you have to resize one zone to make it smaller than the other, and change its position to bring it inside the larger zone.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select ‘Zone-1’ either from the Entity List or directly in Interface.
3. In the ‘Properties’ tab, change the dimensions of ‘Zone-1’ to $x=5$, $y=10$, $z=5$.
4. If you created both zones without moving your avatar, you don’t need to change the zone’s position. ‘Zone-1’ will already be inside ‘Zone-2’. If you moved while creating the zones, select ‘Zone-1’ and move it inside ‘Zone-2’.

Add Different Lighting Effects to Each Zone

When you create a zone, it inherits the properties of the zone your avatar was standing in. This means that both zones inherit the lighting properties of your domain. You won’t notice when you are entering or leaving a zone.

All lighting effects have three modes:

- Inherit: The property is inherited from the zone bounding the current zone.
- Off: The lighting effect is turned off.
- On: The lighting effect is turned on and can be changed to values of your choice.

The keylight represents a parallel source of light, such as the sun. Let’s change the keylight properties for each zone:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select ‘Zone-1’ either from the Entity List or directly in Interface.
3. In the ‘Properties’ tab, change the ‘Key Light’ property to ‘On’ from the drop-down.
4. Click on ‘Key Light’ color, and add these RGB $(255, 0, 0)$ values to add a red key light.
5. Select ‘Zone-2’ either from the Entity List or directly in Interface.
6. In the ‘Properties’ tab, change the ‘Key Light’ property by selecting ‘On’ from the drop-down.
7. Click on ‘Key Light’ color, and add these RGB $(0, 0, 255)$ values to add a blue key light.

When your avatar walks from Zone-1 to Zone-2, you will see the lighting around change from red to blue.

Change a Zone’s Shape

By default, a zone’s shape is a cube, like its bounding box. You can change a zone’s shape to the following:

- None: This will be the default shape (cube).
- Box: The zone’s shape will be a box.
- Sphere: The zone entity’s shape will be a stretched sphere.
- Ellipsoid: The zone entity will take the shape of an ellipsoid.
• Cylinder: The zone entity’s shape will be a cylinder.
• Compound: The zone entity’s shape will be a convex mesh that is an FBX or OBJ file. These complex convex shapes must be composed of multiple shapes. We can’t check against hollowed out interior volumes. For example, if you want a zone shaped like a bowl, you’ll have to build it up from other elements. You can include elements like sides and a floor, especially if you want a user to experience the right collision properties when in the center of the bowl. Upload your FBX or OBJ file to a cloud server, copy the URL, and paste it in ‘Compound Shape URL’.

All shapes will be stretched to fit the zone entity’s dimensions.

**Add a Skybox to a Zone**

A skybox determines the texture of the sky in your domain. The skybox can be just a color, or an image from a URL. For example, you can have a blue sky or use an image of the night sky with stars as a skybox.

To add a blue sky to your zone:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select ‘Zone-1’ either from the Entity List or directly in Interface.
3. In the ‘Properties’ tab, change the ‘Skybox’ property by selecting ‘On’ from the drop-down.
4. Click on ‘Skybox’ color, and add these RGB (0, 0, 255) values to add a blue key light.

To add an image of the night sky to your zone:

1. Host your image on a cloud service and copy the URL.
2. Create a JSON file that refers to the URL and other skybox properties.

```json
{
   "Entities": [ 
    {
     "skybox": { 
      "color": { 
       "blue": 255,
       "green": 255,
       "red": 255
      },
      "url": SKYBOX_IMG_URL
     },
     "skyboxMode": "enabled",
     "type": "Zone",
     "userData": "{\"grabbableKey\":{\"grabbable\":false}}"
    }
   ],
   "Id": ENTITY_ID,
}
```

3. Host the JSON file on a cloud service. Copy its URL.
4. Select ‘Zone-1’ either from the Entity List or directly in Interface.
5. In the ‘Properties’ tab, change the ‘Skybox’ property by selecting ‘On’ from the drop-down.
6. In ‘Skybox source’, add the JSON file’s URL.

You’ll see your zone’s lighting change to the image you specified in the skybox.
**Add Ambient Light to a Zone**

The ambient light in a zone is a source of light different from the key light and provides background lighting. For example, warm sunlight coming from a sunset in your domain is ambient light.

Similar to the skybox, your ambient light image can be added as a JSON file.

1. Select ‘Zone-2’ either from the Entity List or directly in Interface.
2. Change the ‘Ambient Intensity’ to 1.00.
3. In ‘Ambient Source’, add your ambient light JSON file, or click ‘Copy from Skybox’ to use the same image as the skybox.

Your zone’s ambient lighting will change to the image you’ve provided.

**See Also**

- Create New Entities
- Change How Entities Look

**Tutorial: Display a YouTube Channel**

In this tutorial, you will learn how web entities work by creating one displaying a YouTube channel. You can watch videos in High Fidelity using this web entity.

**On This Page:**

- Prerequisites
  - Create a Web Entity
  - Display High Fidelity’s YouTube Channel

**Prerequisites**

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
- Change How Entities Look

**Create a Web Entity**

A web entity is a flat object on which you can view any website of your choosing. A web entity lets you access the internet from inside your domain.

To create a web entity:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Web’ icon to create a web entity. By default, a web entity always displays High Fidelity’s home page.
Display High Fidelity’s YouTube Channel

You can make the web entity display High Fidelity’s YouTube channel.

1. In Interface, pull up your HUD or Tablet and go to **Create**.

2. Select your web entity and go to the ‘Properties’ tab.

3. Scroll down until you see the ‘Source URL’ option. Enter the High Fidelity YouTube channel URL: https://www.youtube.com/user/HighFidelityio. You should see the new page as soon as you hit ‘Enter’.

See Also

- Create New Entities
- Change How Entities Look

Tutorial: Create a Portal

Portals in High Fidelity transport you to the domain of your choice. You can use these portals to travel to a domain you visit frequently instead of using the **GoTo** app on your HUD or Tablet.

On This Page:

- Prerequisites
- Write a Script for the Portal
- Create and Edit an Entity to Use as a Portal

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
Write a Script for the Portal

A portal is an entity with a script attached (entity script). This attached script defines what happens when a user comes in contact with the portal. We’ve used portal.js, the script used to teleport in High Fidelity domains. You can also write your own script to suit your needs.

The portal.js script we’ve used:

- Uses High Fidelity’s JavaScript API to determine when a user walks into the entity and the teleport destination.
- Includes a sound that will played every time a user uses the portal.
- Teleports a user to the specified destination.

Create and Edit an Entity to Use as a Portal

Any entity you create to be used as a portal has to be collisionless so that the script can detect when you walk into the entity.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create an entity to be used as a portal. This can be a 3D model or a box or sphere entity.
3. Go to the ‘Properties’ tab and scroll down to ‘Behavior’.
4. Next to ‘Script’, paste the script URL. In this case, it is ‘portal.js’.
5. The script takes the location you want to teleport to from the ‘User Data’ field under ‘Behavior’.
6. Add hifi:// welcome (High Fidelity’s welcome domain) to the ‘User Data’ field.
7. Scroll down to ‘Collision’ and uncheck ‘Collides’. This is so that a user can walk into the entity and trigger the script.
8. Exit Create mode and walk into the entity.
You will be teleported to High Fidelity’s welcome domain.

See Also

- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts
- Interact with Your Environment

**Tutorial: Create an Avatar Scaling Button**

You can build content in High Fidelity that breaks the laws of physical boundaries by making them oversized or extremely small. To give any visiting users access to such an experience, you can add an avatar scaling button to your domain. This will help users fit into the spaces you design.

**On This Page:**

- Prerequisites
- Write an Avatar Scaling Script
- Create an Entity to Use as a Button
Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts

Write an Avatar Scaling Script

To define the behavior of your avatar and the button, you need to write a client entity script that:

- attaches to an entity (a button in your domain).
- shrinks or increases the size of an avatar.
- defines what happens when a user clicks on or triggers the entity.

In this tutorial, we’ve used shrink-avatar.js, an avatar scaling script used to shrink an avatar down to a tiny size. You can use this script, modify it, or write your own to suit your needs.

The shrink-avatar.js uses High Fidelity’s JavaScript API to determine when a user clicks with the mouse or triggers the entity with their hand controllers. It then scales the avatar to one-tenth its original size.

Create an Entity to Use as a Button

The entity you create for your button has to be triggerable so that the script can detect when you trigger or push the button with your hand controllers.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create an entity to be used as a button. This can be a 3D model, cube, or sphere entity.
3. Go to the ‘Properties’ tab and scroll down to ‘Behavior’.
4. Next to ‘Script’, paste the script URL. In this case, it is ‘shrink-avatar.js’.
5. Ensure that ‘Triggerable’ is selected.
6. After you exit the Create app, test your script by clicking or triggering the button to observe your avatar scale down.

See Also

- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
Tutorial: Open Web Page with Entities

Entities are often used to add objects to your environment. However, you can do so much more with this when you use scripting to define their behavior. In this tutorial, we will use an entity as a button to open a web page on your tablet. You can use this tutorial to do things like directing your visitors to specific Marketplace items for purchase.

On This Page:
- Prerequisites
- Write a Script to Open a Web Page
- Create an Entity to Use as a Button

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:
- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts

Write a Script to Open a Web Page

The script used here opens a web page on the Tablet when a user clicks or triggers an entity. In this example, we’ve written a client entity script that opens the Marketplace web page when an entity/item is triggered. The script looks for the URL in the ‘User data’ property of the entity and injects the Marketplace code into the link. This allows the user who triggered the script to purchase the item without having to go to the Market app on their Tablet or HUD.

You can get the script here.

Create an Entity to Use as a Button

The entity you create for your button has to be triggerable so that the script can detect when you trigger or push the button with your hand controllers.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create an entity to be used as a button. This can be a 3D model, cube, or sphere entity.
3. Go to the ‘Properties’ tab and scroll down to ‘Behavior’.
4. Paste the following JSON data into the ‘User data’ field for your entity:
5. Next to ‘Script’, paste the script URL. In this case, it is openTabletPageButton.js.
6. Scroll down and ensure that ‘Triggerable’ is selected.
7. After you exit the **Create** app, test your script by clicking or triggering the button to open the Marketplace web page for your item.

**See Also**
- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts
- Interact with Your Environment

**Tutorial: Build a Painting Set**

There are different ways to create games and experiences in High Fidelity. In this tutorial, we’ll cover how to make a pixel-like painting set to allow users to “paint” pictures on a canvas made out of “pixels” that are box entities.

**On This Page:**
- **Prerequisites**
- **Create a Painting Set**
- **Add a Paint Brush Script**

**Prerequisites**

Consider getting familiar with the following concepts before starting this tutorial:
- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts
Create a Painting Set

Our painting set comprises three elements:

- A brush, made of a cylinder and a sphere object.
- A palette, made of an octagon and sphere object.
- A canvas, made of box entities.

All of the logic for our painting set is contained in the brush head. The rest of the content is made by parenting entities to one another to make our brush, palette, and canvas.

Create a Paint Brush

We'll start by creating the paint brush. The brush is comprised of two parts, the handle and the brush head. The brush handle is the parent of the brush head, so we can control the movement and color of the brush head using only the handle.

To create the brush handle:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create an entity to be used as the handle. This can be a box or sphere entity.
3. Go to the ‘Properties’ tab and select the ‘Shape’ drop down. Change the shape of the entity to a ‘cylinder’.
4. Name your entity ‘Paint-Paintbrush-Tube’ by selecting the text box at the top of the ‘Properties’ tab.
5. Select your desired handle color from the ‘Color’ picker.
6. Scroll down to the ‘Spatial’ section. Change the local dimensions to: {x: 0.025, y: 0.5, z: 0.025}.

To create the brush head:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click on the ‘SPHERE’ icon to create a sphere entity to be used as the brush head.
3. Go to the ‘Properties’ tab and select your desired brush head color from the ‘Color’ picker.
4. Name your entity ‘Paint-Paintbrush-Head’ by selecting the text box at the top of the ‘Properties’ tab.
5. Scroll down to the ‘Spatial’ section. Change the local dimensions to {x: 0.05, y: 0.1, z: 0.05}.

Once you’ve created the brush head, you can parent the brush handle to it:

1. In Create Tools app, select your brush handle and go to the ‘Properties’ tab.
2. Copy the ‘entityID’.
3. From the Entity List window, select the brush head and go to the ‘Properties’ tab.
4. In the ‘Parent’ text box, paste the entityID of the brush handle (Paintbrush-Tube) entity.
5. Scroll down to the ‘Spatial’ section. Change the local position to {x: 0, y: 0.2, z: 0}.

We’ve detailed how you can add a script to use the brush to transfer color to other objects in Add a Paint Brush Script.

Create a Paint Palette

The second part of our painting set is the palette. This is where you can get creative, and add as many (or as few) paint colors as you’d like. Once you have the base of the palette created, you can repeat the process of adding paints until you are satisfied with the end result.
To create the palette base:

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. **Create an entity** to be used as the palette base. This can be a box or sphere entity.
3. Go to the ‘Properties’ tab and select the ‘Shape’ drop down. Change the shape to an ‘octagon’.
4. Name your entity ‘Paint-Palette-Base’ by selecting the text box at the top of the ‘Properties’ tab.
5. Select your desired palette color from the ‘Color’ picker.
6. Scroll down to the ‘Spatial’ section. Change the local dimensions to \( \{x: 0.55, y: 0.5, z: 0.55\} \).
7. Scroll down to the ‘Behavior’ section. Check ‘Grabbable’.

To create the paint colors:

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. **Create a sphere entity** to be used as your first paint color.
3. Go to the ‘Properties’ tab and name your entity ‘Paint-Color’ by selecting the text box at the top of the tab.
4. Select your desired paint color from the ‘Color’ picker.
5. Scroll down to the ‘Spatial’ section. Change the local dimensions to \( \{x: 0.1, y: 0.05, z: 0.1\} \)
6. In the **Create** Tools app, select your palette base and go to the ‘Properties’ tab.
7. Copy the ‘entityID’.
8. Select your paint color and go to the ‘Properties’ tab.
9. In the ‘Parent’ text box, paste the palette base entityID.
10. Use the grab handles to adjust the position and size of the paint on the palette.
11. Scroll down to the ‘Behavior’ section and uncheck ‘Grabbable’.

Repeat the above steps to create additional paint colors for your palette.

**Create a “Pixel” Canvas**

The last component that makes up our painting set is the canvas we’ll use for our “pixel” style painting. We’ve provided a JSON file for you to import a canvas so you don’t need to go through each step individually, but you can import the grid multiple times to make a larger painting space, if desired.

1. In Interface, go to **Menu > Edit** and select ‘Import Entities from URL’.
2. Paste this URL into the dialog window and select ‘OK’.

The canvas is made up of box entities parented to a single backplate, but you could use any entities to create a scene that could be painted this way.

**Add a Paint Brush Script**

To start painting on the canvas, you have to write a paint brush script that serves as our ‘painting’ logic. This script will:

- Check if the paint brush head has collided with another entity.
- Check if this entity is a paint color, and change the color of the brush head.
• Check if the entity is a different entity. If it is not a paint color, the script will “transfer” it’s color over to the other entity.

To add the paint brush script:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select the Paint-Brush-Head entity.
3. Go to the ‘Properties’ tab and scroll down to ‘Behavior’.
4. Next to ‘Script’, paste the script URL. In this case, it is ‘brushScript.js’.
5. After you close the Create app, test it out by painting on the canvas in your domain!

See Also

• Create New Entities
• Change How Entities Look
• Define an Entity’s Behavior
• Get Started with Scripting
• Client Entity Scripts
• Write Your Own Scripts
• Interact with Your Environment

Tutorial: Create a Purchase Button for Marketplace Sales

If you are selling items on the Marketplace, you can have a space in your domain where users can easily buy your items. This could be done with a display copy of your items, an image of your item, or a purchase button for users to click.

On This Page:

• Prerequisites
• Get an Item’s Marketplace ID
• Write a Purchase Item Script
• Create an Entity to Use as a Button

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

• Create New Entities
• Change How Entities Look
• Define an Entity’s Behavior
• Get Started with Scripting
• Client Entity Scripts
• Write Your Own Scripts
• Purchase Marketplace Items from Script
Get an Item’s Marketplace ID

If you are putting up an item (in your domain) you have on sale in the Marketplace, you need to get the item’s ‘marketplaceID’. This unique identifier can be found when you view your Marketplace item in a browser window.

1. In your browser of choice, navigate to www.highfidelity.com/marketplace.
2. Locate the item.
3. Copy the multi-number identifier at the end of the URL in the address bar of your browser. This is your item’s ‘marketplaceID’.

Write a Purchase Item Script

To make our button work and actually sell an item, we need to attach a client entity script to it. In this example, buy-item.js, the script opens a specified Marketplace page. You can use it as-is, modify it, or write your own script to suit your needs.

The buy-item.js script we’ve used:

- Uses High Fidelity’s JavaScript API to determine when a user clicks with the mouse or triggers the entity using hand controllers.
- Opens the user’s Tablet to the purchase page if there is a valid ‘marketplaceID’ specified in the ‘userdata’ field of the entity that is clicked or triggered.

Create an Entity to Use as a Button

The entity you create for your button has to be triggerable so that the script can detect when you trigger or push the button with your hand controllers.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Create an entity to be used as a button. This can be a 3D model, cube, or sphere entity.
3. Go to the ‘Properties’ tab and scroll down to ‘Behavior’.
4. Select ‘Code’ from the drop down menu in the ‘User Data’ property. Add the following line, replacing ‘YOUR_MARKETPLACE_ID_HERE’ with the ‘marketplaceID’ that you copied in the first step of the tutorial:

   ```javascript
   {'marketplaceID' : 'YOUR_MARKETPLACEID_HERE'}
   ```

5. Next to ‘Script’, paste the script URL. In this case, it is ‘buy-item.js’.
6. Scroll down to ‘Triggerable’ and ensure that ‘Triggerable’ is selected.
7. When you close the Create app, test your script by clicking or triggering the button to open the item’s purchase page. If a user has already purchased the item, they will be shown a page that allows them to re-purchase another copy or view it in their Inventory.

See Also

- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
• Write Your Own Scripts
• Interact with Your Environment

Tutorial: Create a Boombox

You can create a music player that plays all your favorite tracks and also syncs the audio for other users in your domain.

On This Page

• Tutorial: Create a Boombox
  – Prerequisites
  – Create a Boombox Entity
  – Add User Data to Your Boombox
  – Write Music Player Scripts

Prerequisites

• Create New Entities
• Change How Entities Look
• Define an Entity’s Behavior
• Get Started with Scripting
• Client Entity Scripts
• Write Your Own Scripts

Create a Boombox Entity

Your BoomBox will consist of:

• A boombox base model: A model entity that runs an entity server script.
• An ‘ON/OFF’ button: A child entity runs a client entity script to allow users to interact with the boombox.

The boombox will start playing when users click or trigger the ON/OFF button.

To create a boombox:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Use the Create app to import the 3D model entity. You can create your own 3D model for the boombox base or use one we’ve created.
3. Next, create the button entity that users will interact with. This can be a cube entity.
4. Go to ‘Properties’ tab for the button entity. Change the ‘Shape’ property from ‘Box’ to ‘Octagon’ or ‘Cylinder’ depending on your aesthetic preferences.
5. Scroll down to the ‘Behavior’ section and ensure that ‘Grabbable’ and ‘Triggerable’ are checked.
6. Scale, rotate, and move your button to align it to the desired position on the model.
7. With the Create app open, select the 3D model of the boombox. Go to the ‘Properties’ tab and copy the ‘ID’ under ‘Name’.

8. Select the cube entity you created, go to the ‘Properties’ tab, and paste the copied entity ID in the ‘Parent’ field. This makes your boombox model entity the parent of your button entity.

**Add User Data to Your Boombox**

The User Data property is a JSON object that can be customized to fit the needs of a script. User Data also helps in synchronizing and keeping variables the same for all users in a domain. In this case, User Data will contain:

- **Song List**: All URLs of the songs you want played on your boombox. You can also use MP3 or WAV files stored on your local machine.
- **Music player volume information**: You can change this as per your preference.

**Note**: User Data can store information only up to a certain size. We recommend keeping the limit of 10 songs. We support the following formats:

- WAV: 16-bit uncompressed WAV at any sample rate, with 1 (mono), 2(stereo), or 4 (ambisonic) channels.
- MP3: Mono or stereo, at any sample rate.
- RAW: 48khz 16-bit mono or stereo. Filename must include “.stereo” to be interpreted as stereo.

To add User Data to your boombox:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select your boombox entity, not the button.
3. Go to the ‘Properties’ tab. Scroll down to ‘User Data’ and paste the following JSON information. This JSON data consists of 10 songs and the volume setting. You can use your own songs and change the volume setting:

```json
{
    "grabbableKey": {
        "grabbable": false
    },
    "music": {
        ...
    }
}
```

(continues on next page)
The boombox system contains the following scripts and files that allows a user to control audio playback:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity Server Script</td>
<td>This server script handles the state of the music player and plays audio back so that it is synchronized across all users. Actions and behaviors of entities that need to be in the same state for all users, should run on the server. The client script that runs on the button relays the requests for each remotely callable function to execute on the server, and the server script handles the audio playback accordingly.</td>
<td>boom-Box-Entity-Server-Script.js</td>
</tr>
<tr>
<td>Client Entity Script</td>
<td>This client script handles the interactions between users and displays the UI for controlling the boombox via an HTML page using the Tablet Scripting Interface. It listens for mouse clicks and controller triggers, displays the controls, and serves as a relay mechanic between the HTML page and the boombox entity server script.</td>
<td>boom-Box-EntityScript.js</td>
</tr>
<tr>
<td>HTML and CSS</td>
<td>The HTML page displays the controller UI for the boombox through the Tablet Scripting Interface and is styled with CSS. It uses the EventBridge to send the user input from the HTML elements to the boombox entity script, which in turns calls entity server methods depending on the EventBridge message contents.</td>
<td>boom-Box-Controller.html styles.css</td>
</tr>
</tbody>
</table>

You can use the existing versions of our scripts, modify them, or write your own scripts.

If you’re using the existing versions of our scripts:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Select your boombox model and go to the ‘Properties’ tab.
3. Scroll down to the ‘Behavior’ section and paste the entity server script’s URL into the ‘Server Script’ field.
4. Select your button entity and go to the ‘Properties’ tab.
5. Scroll down to the ‘Behavior’ section and paste the client entity script’s URL into the ‘Script’ field. If you’re writing your own scripts or modifying the existing ones, and want to host these in the ‘Asset Browser’:

1. On your computer, create a folder called ‘BoomBox’. You’ll save your files here with the following structure.

![File Structure]

2. Save the entity server script, client entity script, HTML file, and CSS file to the folder in your computer.

3. In Interface, pull up your HUD or Tablet and go to Create.

4. In the Create app, click ‘Open This Domain’s Asset Server’ to view the Asset Browser.

5. Create the same boombox directory in your ‘Asset Browser’ and upload your files.

6. Use the Create app and select your boombox model and go to the ‘Properties’ tab.

7. Scroll down to the ‘Behavior’ section and paste the entity server script’s URL into the ‘Server Script’ field.

8. Use the Create app and select your button entity and go to the ‘Properties’ tab.

9. Scroll down to the ‘Behavior’ section and paste the client entity script’s URL into the ‘Script’ field.

Note: Some additional notes:

- The scripts linked above use relative paths to link to one another, so it’s important to preserve the folder structure given. If you want to move things around, make sure you also update the links in the scripts themselves to reference the new file location.

- If you want to make modifications to your script files, you will need to re-upload them to the asset browser. Reload all content and reload your entity server scripts to see changes take effect after modifying files.

- Entity server scripts do not have access to mouse press or controller events, since these are all handled on the client side.

- HTML pages will not render in the Oculus Quest, so only desktop users will be able to interact with the boombox controls.

See Also

- Create New Entities
- Change How Entities Look
- Define an Entity’s Behavior
- Get Started with Scripting
- Client Entity Scripts
- Write Your Own Scripts
- Interact with Your Environment
1.3.5 3D Models

3D models are *entities* that represent objects you’d like to see in-world such as a water fountain, a castle, or a coffee mug. You can get your 3D model by creating your own, sourcing it externally, or purchasing them from our [Marketplace](#). We only support 3D models in the OBJ and FBX formats.

The appearance of a 3D model is controlled by its materials. The materials supported in High Fidelity are Physically-based Rendering (PBR) materials. This means that a 3D model’s materials will reflect or absorb light as they would (approximately) in real life.

In This Section:

- Get Your 3D Model
  - Get Your 3D Model from 3D Content Stores
  - Create Your Own 3D Model

Get Your 3D Model

Many 3D models are available for purchase on our [Marketplace](#). If the Marketplace doesn’t have the model you’re looking for, you can create and customize your own. All 3D models should be in the OBJ or FBX format and have materials supported by High Fidelity.

On This Page

- Get Your 3D Model
  - Get Your 3D Model from 3D Content Stores
  - Create Your Own 3D Model

Get Your 3D Model from 3D Content Stores

There are many online 3D content websites that contain models that you can purchase or get for free. Keep the following in mind when sourcing 3D models from such sites:

- **Check Licensing Terms:** Make sure you check a model’s licensing terms before you use it. It is your responsibility to ensure that you have sufficient rights to upload the content. When you make a 3D model available on your High Fidelity server, visitors are getting the links to those files in the same way as they would when viewing an image on a website. You should be comfortable and have the rights to re-distribute the 3D content. High Fidelity offers proof-of-purchase certificates for 3D models (uploaded to our Marketplace) that certify that they have been legitimately purchased.

- **Check Materials:** You might find that the model may be missing its textures. If that happens, first check to see if the textures are included. If a model loads into High Fidelity and doesn’t look right, you may also find error information in the Interface logs.

Create Your Own 3D Model

You can create your own 3D model using 3D modeling software such as Blender or Maya. Use any software of your choice as long as:

- The 3D model is in the OBJ or FBX format.
- The 3D model materials are supported by High Fidelity. Use our [materials guide](#) to make sure that your materials load correctly.
Best Practices

Making 3D models for High Fidelity (and VR) is different than making models for films, videos, and games.

- 3D models for VR are rendered twice (for both right and left eyes): This means that the number of polygons on your model and the number of materials you use will affect your performance.
- All VR headsets run at 90Hz: You'll have to keep your framerate at 90fps and be cautious about your model’s size. Models that are too big or very complex can slow down the framerate and make people feel nauseous.

We’ve listed the best practices for creating 3D models for High Fidelity (and VR).

<table>
<thead>
<tr>
<th>Property</th>
<th>Best Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polycount</td>
<td>Your count should resemble that of a model for a tablet game, not too high, but not too low either.</td>
</tr>
<tr>
<td>Edge Loops</td>
<td>Remove edge loops that are not needed.</td>
</tr>
<tr>
<td>Mesh</td>
<td>Clean the mesh to make sure there are no N-gons and no coplanar faces.</td>
</tr>
<tr>
<td>Materials</td>
<td>Always try to create Atlas maps. When every piece of your content shares the same material and UV space, it is an Atlas map. For example, if you create a robot, all its pieces should share one UV map, instead of giving its hands, feet, or face separate materials and UV maps.</td>
</tr>
<tr>
<td>Materials</td>
<td>High Fidelity's engine only supports one UV mapping per material.</td>
</tr>
<tr>
<td>Textures</td>
<td>PNG, JPEG and JPG files are recommended, but we also support TGA, TIF and TIFF formats.</td>
</tr>
<tr>
<td>Textures</td>
<td>Choose the color types wisely to minimize the size of the final file.</td>
</tr>
<tr>
<td>Textures</td>
<td>PNG-8 has only ON/OFF transparency, has a palette of colors (256 colors, like GIF), and can be used to mask transparency.</td>
</tr>
<tr>
<td>Textures</td>
<td>For more color resolution, you can use PNG-24. For translucent mask or transparent textures, use PNG-32.</td>
</tr>
<tr>
<td>Textures</td>
<td>Do not use PNG-48 or PNG-64, as neither are supported by High Fidelity.</td>
</tr>
<tr>
<td>Textures</td>
<td>When loaded in the engine, textures are automatically resized to a grid of 128x128. Pick sizes which are multiples of 128.</td>
</tr>
</tbody>
</table>
| Draw Calls | Draw calls happen before something gets rendered on screen.  
1 model w/ 1 material = 1 draw call  
There are no definitive measures for a desirable polycount. You need to balance between draw calls and polys. Fewer draw calls means more room for polys. Smaller textures means more room for higher poly models. |

See Also

- Shopping the Marketplace

Import Your 3D Model

You can import a 3D model into High Fidelity by hosting it online or hosting it on your Asset Server. When you host your 3D model on your Asset Server, it will not be available in other domains. If you wish to make your models available in other domains, it’s best to host it online on cloud services.
On This Page:

- Import 3D Models Using a Hosting Service
- Import 3D Models Using the Asset Server

Import 3D Models Using a Hosting Service

If you wish to make your model available for purchase on our Marketplace, or make it available to users in other domains, you can host it on a cloud service of your choice and then import it.

1. Upload your 3D model’s files to the hosting service of your choice.
2. Copy the OBJ or FBX file’s URL.
3. In Interface, pull up your HUD or Tablet and go to Create.
4. Select the ‘Model’ icon.
5. Paste the model URL you copied in step 2 and click ‘Add’.

Import 3D Models Using the Asset Server

You can import your 3D model through your Asset Server. The Asset Server hosts files or assets that can either be added as-is to a domain or that are referenced by existing entities and scripts already in a domain. This is accessible only in a domain where you have the permission to create and edit an entity.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click ‘Open this Domain’s Asset Server’.
3. The window that opens is called the Asset Browser. In the Asset Browser, click ‘Choose File’ to select your 3D model’s files.
4. Enter a folder name to specify a path for your 3D model. This path should be added before the model’s file name /model_filename. By default, there is no folder created for your asset. However, we recommend that you add folders for each of your models to help keep them organized.
5. In the Asset Browser, select the OBJ or FBX file and click ‘Add to World’. Then, click ‘Add’ on the confirmation window that opens.

See Also

- Manage Your Domain Assets
- Add Your 3D Model to the Marketplace
- Creator Tools
- Create New Entities

PBR Materials Guide

The appearance of a 3D model is controlled by its materials. The materials supported in High Fidelity are physically-based rendering (PBR) materials. This means that a 3D model’s materials will reflect or absorb light like how they would (approximately) in real life.
Introduction to Materials, Textures, and Shading

A 3D model’s appearance is controlled by its materials. For example, a 3D model of a key will use a material that determines its color, how metallic it looks, and if its surface is bumpy or smooth. A 3D model of a brick wall will have material that determines its roughness and color.

High Fidelity supports physically-based rendering (PBR) materials. This means that your model will behave like a real world object when exposed to light. For example, the same 3D model of a key will shine and reflect any light that falls on it. The 3D model of a brick wall will not shine, but will reflect enough light for you to observe its colors and surface.

A material contains texture and shading information.

Textures

Textures are flat images that are applied to 3D models. These add detail on how a 3D model’s material looks. For example, a 3D model of a tree trunk will use a texture of bark to show what the surface looks like.

High Fidelity supports the use of the following texture formats:

- PNG (recommended)
- JPEG, JPG (recommended)
- TGA
- TIFF, TIF

For best performance, we recommend baking all 3D models (including textures) before uploading it to High Fidelity.

Shading

Since High Fidelity supports PBR materials, the shading used depicts an accurate representation of how light interacts with different material surfaces. This means that your 3D model will not look the same under different light settings.

The PBR shader has a set of material parameters or channels that can be modified to create different types of materials.

Material Channels

Material channels determine various parameters such as the roughness or color of a material. You can determine the value of each channel in two ways:

- **Setting a Value:** The value of a channel is a value set on a slider. Setting a value is like turning off or turning on a switch. For example, if you look at your phone, some parts of it are shiny and some are matte. When you use a value, the entire object reflects that value. If you want different parts of an object to reflect varied roughness, you’ll need to use a map.
• **Using a Map:** The map is an image which you can import to define a property. You will use a map to apply a texture to your 3D model. For instance, your phone may have a case that is matte, but the rest of your phone is shiny. You can use a map to set the case as matte and the phone as shiny.

All materials in High Fidelity have the following channels that determine how they look:

<table>
<thead>
<tr>
<th>Channel Type</th>
<th>Description</th>
<th>Value</th>
<th>Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albedo</td>
<td>This channel defines the material’s color. You can pick any color value of your choice.</td>
<td>sRGB</td>
<td>sRGB</td>
</tr>
<tr>
<td>Metallic</td>
<td>This channel determines if the material is metallic or not. You cannot have a material that is half metallic, it is either metallic or it isn’t.</td>
<td>[0,1]</td>
<td>[0,1]</td>
</tr>
<tr>
<td>Roughness</td>
<td>This determines how rough/matte or glossy/shiny an object is, using brightness levels.</td>
<td>[0,1]</td>
<td>[0,1]</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal is a channel that renders an object like there is actual geometry. For instance, normal would add bumps and other irregularities to a stone or ridges to a sea-shell.</td>
<td>xyz</td>
<td>bump</td>
</tr>
<tr>
<td>Opacity</td>
<td>Opacity determines if an object is transparent or opaque.</td>
<td>[0,1]</td>
<td>mask, alpha</td>
</tr>
<tr>
<td>Occlusion</td>
<td>This property approximates the shading to be as natural as possible. This means that it will reproduce how objects interact with light.</td>
<td>—</td>
<td>[0,1]</td>
</tr>
<tr>
<td>Emissive</td>
<td>This channel controls the amount of light being reflected from an object.</td>
<td>sRGB</td>
<td>sRGB</td>
</tr>
<tr>
<td>Scattering</td>
<td>Scattering determines how light will behave when it hits human skin. This channel details how light is reflected or absorbed by human bodies.</td>
<td>[0,1]</td>
<td>[0,1]</td>
</tr>
<tr>
<td>Material Type</td>
<td>This channel decides if an object is lit or unlit.</td>
<td>[lit, unlit]</td>
<td>—</td>
</tr>
</tbody>
</table>

Notes:

1. If you set transparency with a texture, the transparency (alpha) should be in the material’s albedo texture, as a PNG file with transparency and not as a separate transparency texture.

2. High Fidelity’s renderer can draw two different kinds of transparency: “alpha” (255 graduates steps of transparency, no shading on surface, casts no shadows,) and “mask” (binary transparency, full shading of opaque surface, whole surface casts shadow.)

3. To determine whether a texture is treated as a mask or as alpha, the engine looks for alpha values between 2% and 98%. An easy way to create a mask texture is to save your image as a PNG-8 since it only supports binary transparency, while PNG-24 supports a range of transparency levels.

4. We support using a second UV set with the following texture channels only: Light Map and Ambient Occlusion.

**Sample Materials and Their Textures and Shading**

High Fidelity supports different types of materials. We’ve created sample objects with each material type. You can download each object from a repository on GitHub, or run a script in High Fidelity to upload all sample objects in your domain.

We’ve listed all material information (including textures, shading, and channel values and maps) for these sample objects here.
Set Material Values in Blender

When you create a model in Blender, you have to export it in the FBX format to use in High Fidelity. Additionally, you have to modify material properties and textures in Blender to match the PBR material textures in High Fidelity. Doing so ensures that your model appears like how you want it to.

By default, any material property set with a texture will override a property set with a value. The only exception to this is in the case of the albedo color set with an RGB color value and a texture, in which case the albedo value and texture will multiplied together.

We’ve included images where the fields corresponding to each supported PBR channel in Blender are highlighted, along with details about which values and colors correspond to the range corresponding with that channel. It should be noted that all of the Blender material properties below only relate to exported FBX format models. Models exported as OBJ or other formats do not have full PBR material support in High Fidelity yet.
Material textures

Selected Texture
JPG, PNG-8, or PNG-24

Influences:
(check all that apply)

Albedo
Opacity

Metallic
Roughness*

Occlusion/Lightmap**
Emissive
Normal

* Any roughness texture's color must be inverted to work correctly as a normal texture.

** To specify whether a texture is a lightmap or an occlusion map, in Material > Shading, set Ambient to 1.0 for the texture to be treated as an occlusion map and 0.0 for the texture to be treated as a lightmap.
Set Material Values in Maya

Use the graphics below to set the right material values and textures in Maya.
### Material values

- **Albedo**: Any RGB value
- **Metallic**: Any grayscale value (white = metallic, black = non-metallic)
- **Roughness**: Any value 0 - 1 (0 = perfectly rough, 1 = perfectly smooth)
- **Emissive**: Any value RGB value (black = non-emissive)
- **Opacity**: Any value 0.0 - 1.0 (0 = opaque, 1 = transparent)

Note: To set a material value for opacity, you must change the Parent Material above to `presets/Standard_Transparent`.

Note: To improve the size and performance of your material, break these connections:

Right-click on label and choose "Break Connection"
Enrich your High Fidelity experience by having 3D models in your domain with animations. For example, you can import the 3D model of a flag that appears to flutter with the wind using this feature.
Prerequisites

You need to be familiar with creating animations in 3D modeling tools such as Blender and Maya before importing an animation into High Fidelity.

Prepare a 3D Model Animation

Before you import an animation into High Fidelity, adjust some settings in the 3D modeling tool of your choice to ensure that it plays smoothly.

1. Set the framerate to 30 fps for the best results (our recommendation).
2. Bake your animation channels, key frames, and inbetweens to ensure that High Fidelity reads everything. This is to ensure that your animation doesn’t stop and start, but appears smooth and flows through each movement.
3. Prepare to export the skeleton and frames that are being used in the animation.
4. Export your animation as an FBX file.
5. Upload this FBX file to a cloud server. Copy the URL.

Import an Animation

Once you complete uploading your animation’s FBX file, you can import the 3D model and it’s animation into High Fidelity.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click on the ‘MODEL’ icon and enter your 3D model’s URL. If you have saved your 3D model’s FBX file with the animation, the model’s URL and the animation’s URL will be the same. Otherwise, your animation is saved as a separate FBX file.
3. In the ‘Properties’ tab, scroll down to ‘Animation’ and paste the animation’s URL.
4. You can edit the following animation properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Play Automatically</td>
<td>Enable this to play your animation automatically when a user loads a domain.</td>
</tr>
<tr>
<td>Loop</td>
<td>Select this property to play your animation in a continuous loop.</td>
</tr>
<tr>
<td>Allow Transition</td>
<td>Enable this to let the animation move through space. This means that the joints will not only rotate, but translate through three dimensions.</td>
</tr>
<tr>
<td>Hold</td>
<td>Select to pause your animation at a particular frame.</td>
</tr>
<tr>
<td>Animation Frame</td>
<td>Enter the frame at which you want to pause or hold your animation.</td>
</tr>
<tr>
<td>First Frame</td>
<td>This is the first frame from when your animation to start.</td>
</tr>
<tr>
<td>Last Frame</td>
<td>This is the last frame where you want your animation to stop. It will not play beyond this specified frame.</td>
</tr>
<tr>
<td>Animation FPS</td>
<td>This is the animation’s framerate.</td>
</tr>
</tbody>
</table>

5. You can also control an animation’s properties and when it starts playing with an entity script.

See Also

- Import Your 3D Model

Tutorial: Create Buildings

You can create large-scale worlds in High Fidelity. A big part of such worlds are buildings. Perhaps you have an idea for a meeting space, a virtual office, an apartment complex, or even personal mansion. We’ll walk through some of the options available to you to create buildings that can be explored for your own domain.

On This Page:

- Create or Find Buildings
  - Build a 3D Building in Maya or Blender
  - Create a 2D Floor Plan
  - Find Buildings
- Add Your Building to Your Domain
- Customize Your Space
Create or Find Buildings

There are many options for creating your own custom buildings to bring into your High Fidelity domains, ranging from primitive building to ground-up 3D modeling and photogrammetry capture. You can use any of the following methods to make a building:

- Build your own custom 3D models in a modeling program
- Create a 2D floorplan that is converted to 3D
- Make a building with primitives in High Fidelity

Build a 3D Building in Maya or Blender

If you're already familiar with a 3D modeling program like Maya or Blender, you can use these tools to create your own custom buildings. Models that are used as interior spaces should be carefully constructed so that collisions align with the mesh and not around the entire model, so that users can easily move throughout the inside. If you have closed off spaces and doors, either inside the building or to the outside of the model, you can find an example of a scripted door with open/close behavior on Marketplace.

Create a 2D Floor Plan

Another option for creating a custom floor plan (no 3D modeling required!) is to generate a 2D floor plan and use a service like Archilogic to generate a 3D mesh of your floor plan. While you can use any art tool to sketch out a 2D layout, you can also pick a program designed for floor plans, including:

- Draw.io, a browser-based flowchart program that exports drawings and can be used to create basic floor plans
- RoomSketcher, another browser-based app for layouts and building designs
- MagicPlan, a mobile app that generates floor plans based on a physical location

Once you have a 2D floor plan, you can upload it to Archilogic or another service for generating 3D models based on 2D designs. Archilogic also allows you to include reference images of what you’d like your model to include to influence the textures or appearance of the generated model, which you can then customize further before importing to High Fidelity.

Find Buildings

If you’d like to use an existing 3D model, you can find options for buildings in High Fidelity under the ‘Architecture’ section of Marketplace. You can also browse sites like Clara.io from within High Fidelity or in your desktop browser. Look for existing FBX models to use as buildings, and if necessary, specify a version of the mesh stored as an OBJ to set up collisions properly in your domain.

Add Your Building to Your Domain

Set up your own domain that you can host content in. If you’re planning on using your building in a single domain, you can host your building by:

1. Opening the Asset Browser and uploading your model to your domain
2. Placing your building by clicking ‘Add to World’
3. Using the Edit tools to change the scale and location of your building, if necessary
Customize Your Space

Once you have your building in High Fidelity, you’ll likely want to customize it! You can add in various decorations, including posters, house plants, furniture, and games through the various Marketplace objects that are available in High Fidelity. You can also follow the steps that you used to add your building into your domain on external 3D models that you want to use in your building. Keep in mind that any objects added to the Asset Server are only available for that specific domain - if you want to use your decorations or furniture in additional domains, you’ll need to host them somewhere that is publicly accessible to each different computer that the domains are hosted on.

You can swap out textures to your model directly in High Fidelity by specifying new files in the ‘Textures’ section of the Entity Properties window that align with the names of the original texture files, viewable under ‘Original textures’.

See Also

- Set Up Your Domain
- Manage Your Assets
- Environments

1.3.6 Environments

A High Fidelity domain can have an environment to express a theme. Environments include all the domain content, such as entities, skyboxes, assets, and more. You could have a domain with a deserted island environment or a cyberpunk apartment environment.

Environments are available for purchase on our Marketplace. If you don’t want to use any of the available environments, you can create your own.

Create an Environment

An environment consists different types of 3D models and a skybox following the theme of your choice. You can create an environment using:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marketplace Items</td>
<td>You can purchase various models like buildings and skyboxes from the Marketplace. For example, you can change the skybox in your domain by purchasing a different one from the Marketplace.</td>
</tr>
<tr>
<td>3D Models Available</td>
<td>You can purchase or get 3D models from online 3D content websites. Then, you can import these models using the Asset Server.</td>
</tr>
<tr>
<td>Your 3D Models</td>
<td>You can use 3D modeling software like Blender or Maya to create your own 3D models and import them into your High Fidelity domain.</td>
</tr>
<tr>
<td>A Content Archive File</td>
<td>High Fidelity domain settings have downloadable content archives. These archives are zip files containing all domain content information. You can use a backup file of your own or one sent to you by a user.</td>
</tr>
</tbody>
</table>

See Also

- Shopping the Marketplace
- Get Your 3D Model
- Import 3D Models Using the Asset Server
- Restore Your Domain
1.3.7 Tablet Apps

Tablet apps (or simply “apps”) in High Fidelity are customizable programs that expose functionality in an easy-to-use user interface. Apps let you take complex code from our JavaScript API and simplify it into a window with controls for others to use.

**Note:** To create custom apps, you must have a basic knowledge of web development (HTML, CSS and JavaScript) and be able to navigate our API.

The steps involved in creating a tablet app are:

1. **Create icons to show up on the tablet and HUD**
2. **Design your app’s UI in HTML and CSS**
3. **Add event handlers to your HTML file**
4. **Write a JavaScript file that:**
   - Adds a button to the tablet and HUD
   - Loads your app
   - Closes your app
   - Listens for events
   - Runs your code (in this case, *create some gemstones*)

**Tutorial: Create Gemstone Switching Tablet App**

In this tutorial, we will walk through the above steps to create an app called “Gemstone Magic Maker”. This simple app lets you spawn colorful little gemstones in VR that you can share with your friends.

**Create icons to show up on the tablet and HUD**

You need two icons to show up on the tablet and HUD: an SVG or PNG image to display on the app button when the app is active, usually named `<appName>-a.svg` and another to display when the app is inactive, usually named `<appName>-i.svg`.

We recommend the following specs for your icons:

- Size: 50px by 50px
- Color: White on a transparent background (for inactive icons) and black on a transparent background (for active icons)
- File format: SVG or PNG

You can create your own icon using graphic design software or any other online resources.
Design your app’s UI in HTML and CSS

Your app’s UI should provide text on how the app works and use familiar UI elements that a user knows how to interact with (such as buttons, scroll bars, and links). Keep in mind that the tablet screen dimensions are 480 x 720, so all of your UI should be confined to this space.

To help you get started, we’ve put together a quick start HTML template that you can reuse. It contains the same layout, styling and font as the main menu screen, and has a header bar for your app title. With just a few simple modifications, you can create have a simple app UI within minutes.

![Image of HIF Tablet Sample App and Gemstone Magic Maker]

Add event handlers to your HTML file

The Tablet UI framework provides a communication channel called EventBridge. It allows you to send and receive events between the client script (gemstoneApp.js) and JavaScript in your web app (gemstoneMagicMaker.html). Use the following EventBridge code inside of `<script> </script>` tags in the body of your HTML file to handle the button clicks:

```javascript
function main() {
    // Send an event to gemstoneApp.js when the page loads
    // and is ready to get things rolling
    console.log("document ready");
    var readyEvent = {
        "type": "ready",
    };

    // The event bridge handles events represented as a string the best.
    // So we first create a JavaScript object, then convert to string
    EventBridge.emitWebEvent(JSON.stringify(readyEvent));

    // Send an event when user click on each of the gemstone buttons
    $('.gemstone-button').click(function(){
        console.log(this.value + " button click");
        var clickEvent = {
            "type": "click",
        };
        EventBridge.emitWebEvent(JSON.stringify(clickEvent));
    });
}

(continues on next page)
Write a JavaScript file

Your JavaScript file will contain all of the core functionality of your app. At a minimum, we require that you have code that adds a button to the tablet and HUD, loads your app, closes your app gracefully, and listens for events. Below, you will find code samples to do each of these things.

Add buttons to the tablet and HUD

Use the AppUI module to automatically add your app’s button to the tablet and HUD, and to wire button click handlers:

```javascript
(function () { // BEGIN LOCAL_SCOPE
    var AppUi = Script.require('appUi');

    var ui;
    function startup() {
        ui = new AppUi({
            buttonName: "APP-NAME", // The name of your app
            home: Script.resolvePath("app.html"), // The path to your app’s UI
            graphicsDirectory: Script.resolvePath("./") // The path to your button icons
        });
        startup();
    }

    startup();

    // END LOCAL_SCOPE

    (function () { // BEGIN LOCAL_SCOPE
        var AppUi = Script.require('appUi');

        function onOpened() {
            console.log("hello world!");
        }

        var ui;
        function startup() {
            ui = new AppUi({
                buttonName: "APP-NAME", // The name of your app
                home: Script.resolvePath("app.html"), // The home screen of your app that appears when clicking the app button
                graphicsDirectory: Script.resolvePath("./") // Where your button icons are located
            });
            startup();
        }

        // END LOCAL_SCOPE
    })(); // END LOCAL_SCOPE
```

Determine the app’s startup behavior

If you want your app to do something specific when it is opened, use the AppUI module’s onOpened functionality. For example, you could:

- Query a server to get a response and determine what to show on the UI
- Start displaying a 3D interface separate from the tablet
- Determine the display mode (VR/Desktop) and change things to show on the UI

```javascript
(function () { // BEGIN LOCAL_SCOPE
    var AppUi = Script.require('appUi');

    function onOpened() {
        console.log("hello world!");
    }

    var ui;
    function startup() {
        ui = new AppUi({
            buttonName: "APP-NAME", // The name of your app
            home: Script.resolvePath("app.html"), // The home screen of your app that appears when clicking the app button
            graphicsDirectory: Script.resolvePath("./") // Where your button icons are located
        });
        startup();
    }

    // END LOCAL_SCOPE
```

(continues on next page)
Close the app gracefully

The AppUI module ensures that your app closes gracefully. However, if you want to do something else when you close the app, you can with the `onClosed` functionality built into the AppUI module. For example, you could:

- Remove 3D interfaces
- Stop secondary scripts

```javascript
(function() { // BEGIN LOCAL_SCOPE
var AppUi = Script.require('appUi');

function onOpened() {
    console.log("hello world!");
}

function onClosed() {
    console.log("hello world!");
}

var ui;
function startup() {
    ui = new AppUi({
        buttonName: "APP-NAME", // The name of your app
        home: Script.resolvePath("app.html"), // The home screen of your app that appears when clicking the app button
        graphicsDirectory: Script.resolvePath(".\"), // Where your button icons are located
        onOpened: onOpened // See the simple function above
        onClosed: onClosed // See the simple function above
    });
    startup();
}(); // END LOCAL_SCOPE
```

Listen for events

In step 3 above, we added event handlers to your HTML file. Now, you need to add code to your JavaScript file to listen for the events:

```javascript
function onWebEventReceived(event) {
    print("gemstoneApp.js received a web event: " + event);
}
tablet.webEventReceived.connect(onWebEventReceived);
```

Create gemstones

The final step is to code the behavior of your JavaScript file. In this case, we’ll create gemstones using High Fidelity’s JavaScript API. Each gemstone will be created as an entity, and we can change the gemstone’s properties using the Entity namespace.
Calculate the position of each new gemstone

The following code gives us a position right in front of the user:

```javascript
// Helper function that gives us a position right in front of the user
function getPositionToCreateEntity() {
  var direction = Quat.getFront(MyAvatar.orientation);
  var distance = 0.3;
  var position = Vec3.sum(MyAvatar.position, Vec3.multiply(direction, distance));
  position.y += 0.5;
  return position;
}
```

Set the gemstone’s properties and add it

The gemstone will be created when gemstoneApp.js receives click events from each of the buttons:

```javascript
// Handle the events we’re receiving from the web UI
function onWebEventReceived(event) {
  print("gemstoneApp.js received a web event:" + event);

  // Converts the event to a JavaScript Object
  if (typeof event === "string") {
    event = JSON.parse(event);
  }

  if (event.type === "click") {
    // Define the entity properties of for each of the gemstone, then add it to the scene
    var properties = {
      "type": "Shape",
      "position": getPositionToCreateEntity()
    ,
      "userData": "{\"grabbableKey\":\{"\"grabbable\":true\\}}"
    ,
    properties.name = "Emerald"
    ,
    properties.shape = "Dodecahedron"
    ,
    properties.color = {
      "blue": 122,
      "green": 179,
      "red": 16
    }
    ,
    properties.dimensions = {
      "x": 0.20000000298023224,
      "y": 0.26258927583694458,
      "z": 0.20000000298023224
    }
    ,
    Entities.addEntity(properties);
  } else if (event.data === "Ruby") {
    properties.name = "Ruby"
    ,
    properties.shape = "Octagon"
    ,
    properties.color = {
      "blue": 160,
      "green": 52,
      "red": 237
    }
    ,
    properties.dimensions = {
      "x": 0.20000000298023224,
      "y": 0.24431547522544861,
```

(continues on next page)
Congratulations, you have successfully created an app in High Fidelity! To use your app, upload it to a cloud platform, such as Amazon S3, Google Cloud Storage, Microsoft Azure, etc. Once hosted, you can install it and use it:

1. In Interface, go to Edit > Running Scripts.
2. Under Load Scripts, click ‘From URL’ and enter the URL to your hosted JavaScript file.
3. Click the app icon on the tablet or HUD to open the app.

See Also

- Write Your Own Scripts
- API Reference: Entities
- API Reference: Script
- API Reference: Quat
- API Reference: Vec3
1.4 Script

High Fidelity uses scripts (written in JavaScript) for a number of different things: creating content, moving your avatar, playing audio at a specific location, wearing an avatar attachment, and much more.

Throughout this chapter, learn about the different types of scripts and how you can use them to create new experiences:

1.4.1 Get Started with Scripting

Many of the scripts in High Fidelity run behind the scenes, so that you don’t even know they’re running. However, if you want to create some advanced behavior, you may need to write your own scripts to make sure everything works correctly.

This page ensures that you know what type of script to use and helps you get started running your own simple scripts.

On This Page

- Get Started with Scripting
  - JavaScript Basics in High Fidelity
  - Types of Scripts
  - Scripting Permissions
  - Running Scripts Window
  - Sample Scripts
  - Scripting Console
  - Debug Window

JavaScript Basics in High Fidelity

High Fidelity scripting runs on a JavaScript engine that is provided with Qt.

Note: Note that any functionality that runs around web pages (such as cookies, local storages, or databases) does not work with 3D environments such as High Fidelity. For this reason, you cannot use JavaScript frameworks such as Angular, React, Express, jQuery, Vue, etc.

You are likely to interface most with these High Fidelity APIs:
### API(s)

| Entities | Lets you manipulate the entities around you, as long as you have permissions to do so. This means you can add, remove, and edit entities. Everyone has access to get properties of an entity, and can be used to find Entities in range, direction, collision, or raytrace. |
| AvatarList | Lets you get information on an Avatar, or manipulate your own client-only MyAvatar. The information here will always be the avatar information of the client running the script. AvatarList and AvatarManager are basically the same. |
| Script | Lets you to connect callbacks from your client to script, such as functionality that is dependent on time (Script.update, Script.setTime, Script.setInterval etc), connect paths relatively to Assets (Script.relativePath), refer to other scripts (Script.include), or create events which occur when the script is turned off (Script.scriptEnding). |

There are many other APIs available, and we encourage you to make sure use of them as you become more comfortable scripting in High Fidelity.

### Types of Scripts

<table>
<thead>
<tr>
<th>Script Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Script</td>
<td>Interface scripts run as long as you have Interface running. With these scripts, you can perform one-time creation tasks or modify the user experience with new menus, overlays, tweaks, plugins, and extensions.</td>
</tr>
<tr>
<td>Assignment Client Script</td>
<td>Assignment client scripts coordinate the actions between entities and avatars in your domain. These scripts continue to run even when you shut down Interface.</td>
</tr>
<tr>
<td>Avatar Script</td>
<td>Avatar scripts run on an avatar and can give it unique effects, such as flowing hair.</td>
</tr>
<tr>
<td>Client Entity Script</td>
<td>Client entity scripts are scripts attached to entities that run locally in response to a user in a domain. With these scripts, you can customize what happens when a user encounters an entity.</td>
</tr>
<tr>
<td>Server Entity Script</td>
<td>Server entity scripts are scripts attached to entities that do not require a user to trigger. These scripts control entities so that their behavior is seen and heard by everyone in the domain.</td>
</tr>
</tbody>
</table>

### Scripting Permissions

Each domain owner has the ability to restrict create and edit permissions. If the script you want to run adds or edits entities and you don’t have the permission to do so, you won’t see any objects created or changed. However, you will still see the script listed in the Running Scripts window.

### Running Scripts Window

The Running Scripts window can be used to load, run and stop scripts from a URL or from a disk drive. High Fidelity also provides a number of sample scripts for you to try out.

To open the Running Scripts window, go to Edit > Running Scripts or press Ctrl + J on your keyboard.
Sample Scripts

High Fidelity comes with a collection of scripts designed to improve your experience as a user and provide tools for developing your own content. We encourage you to look at these scripts as a resource to learn how to code your own.

Note: Loading (or running) a script lets you test the functionality and see how it behaves. If you want to view the actual code, you will need to open the file in the text editor of your choice. In the ‘Running Scripts’ window, click the ‘Reveal Scripts’ folder and browse to the JavaScript file that you want to view.

These are the scripts we have available:

<table>
<thead>
<tr>
<th>Scripts Folder</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>android</td>
<td>These scripts were built to run on Android devices.</td>
</tr>
<tr>
<td>developer</td>
<td>These scripts were created for internal use and debugging, but are available as advanced developers may find them useful when creating content. These scripts are not “entry-level” and are not guaranteed to work or be documented.</td>
</tr>
<tr>
<td>modules</td>
<td>These scripts create external tools that simplify scripting in High Fidelity. For example, the AppUI module helps you create a tablet app, while the Request module processes HTTP requests.</td>
</tr>
<tr>
<td>system</td>
<td>These scripts are critical to the stability and usability of High Fidelity. Making changes to these scripts is not recommended, nor is it easy, as you may need ‘administrative’ privileges.</td>
</tr>
<tr>
<td>tutorials</td>
<td>These scripts provide examples of what you can do using scripts in High Fidelity. Examples include: creating butterflies, making your avatar clap, and adding ambient sound to your domain.</td>
</tr>
</tbody>
</table>

Load and Run a Script

To run a script:

1. Open the ‘Running Scripts’ window.
2. For scripts hosted in the cloud, click ‘From URL’. Enter the URL of your script file and click ‘OK’.
3. For scripts on your local computer, click ‘From Disk’. Browse to your script file and click ‘Open’.
4. To load a sample script, browse to the script at the bottom of the ‘Running Scripts’ window.

Reload or Stop a Script

To reload or stop a script, open the ‘Running Scripts’ window and do one of the following:

- To reload all running scripts, click the ‘Reload All’ button at the top of the ‘Running Scripts’ window.
- To reload a specific script, click the circular arrow next to the script.
- To stop all running scripts, click the ‘Stop All’ button at the top of the ‘Running Scripts’ window.
- To stop a specific script, click the ‘X’ next to the script.

Add a Script to the Default Scripts

You can add a script to the default scripts to run every time you start Interface.

- In Interface, pull up your Tablet or HUD and go to Menu > Edit > Running Scripts.
• Click ‘Reveal Scripts Folder’ at the bottom.
• In the file explorer window, open the ‘defaultScripts.js’ file.
• Add your script to this file to make it run with other default scripts. Ensure the folder path to your script is correct.

**Note:** The ‘defaultScripts.js’ file is updated every time you update Interface to the latest release version. This means that any changes you make to the file will be overwritten. You can avoid this by writing and running a ‘loader’ script to load scripts on start up.

**Scripting Console**

The Scripting Console lets you test and run short script snippets quickly in High Fidelity to see how they work. To open the console, go to the ‘Developer menu’, then **Scripting > Console**. If the ‘Developer’ menu is not visible, first go to the ‘Settings’ menu and click ‘Developer’ Menu.

![Scripting Console](image)

**Debug Window**

The Debug Window shows the output generated by your running scripts. This lets you watch the script(s) in action and make sure that it is running as you intended. If the script fails, the debugger can help you identify what went wrong, and point you to specific lines of code where the error occurred. To open the Debug Window, go to the ‘Developer’ menu, then **Scripting > Script Log** (HMD Friendly). If the Developer menu is not visible, first go to the ‘Settings’ menu and click ‘Developer’ Menu.
1.4.2 Write Your Own Scripts

High Fidelity’s robust JavaScript API provides the tools for you to build great content and user experiences in VR. In this section, you can find simple code samples to do common tasks in High Fidelity. To see these code samples in action, copy the code to a file, `testScripts.js`, saved somewhere on your computer.

On This Page:

- Write to the Debug Window
- Create an Entity
- Edit an Entity

Write to the Debug Window

This is an example of an interface script and cannot be attached to an entity. It shows you how to print something to the debug window. In this example, we’ll start with a simple “Hello, World” script.

```javascript
print("Hello, World");
```

1. Copy and paste this in a file `testScript.js` and save it on your computer.
2. When you load and run this script, it will write the words “Hello, World” to the ‘Debug Window’ in High Fidelity.

```javascript
var position = Vec3.sum(MyAvatar.position, Quat.getFront(MyAvatar.orientation));
var properties = {
  type: "Box",
  name: "ScriptBox",
  position: position,
  color: { red: 255, green: 0, blue: 0 }
};
var entityID = Entities.addEntity(properties);
print("Entity added");
```

**Create an Entity**

Instead of using the Create app to add an entity, you can create one using an interface script.

1. Copy and paste this in a file `testScript.js` and save it on your computer.
2. When you load and run this script, it will locate your avatar in the domain, create a new entity based on the customized properties that you set, then print a line to the ‘Debug Window’. In this case, the entity will be a red box.

**Edit an Entity**

To manipulate an entity’s properties, you can use `Entities.editEntity` in an interface script.

```javascript
var entityID = Entities.addEntity({
  type: "Box",
  position: Vec3.sum(MyAvatar.position, Quat.getFront(MyAvatar.orientation)),
});
```
var properties = Entities.getEntityProperties(entityID, ["color"]);
print("Entity color: " + JSON.stringify(properties.color));

Entities.editEntity(entityID, {
    color: { red: 255, green: 0, blue: 0 }
});
properties = Entities.getEntityProperties(entityID, ["color"]);
print("Entity color: " + JSON.stringify(properties.color));

1. Copy and paste this in a file testScript.js and save it on your computer.
2. When you load and run this script, it will locate your avatar in the domain, create a new entity based on the customized properties that you set, then print the color of that entity to the ‘Debug Window’. Then, the script changes the color of the entity to red, and prints the new color in the ‘Debug Window’.

See Also

- Get Started with Scripting
- Load and Run a Script
- Interface Scripts
- API Reference

1.4.3 JavaScript Tips & Tricks

High Fidelity’s robust JavaScript API provides the tools for you to build great content and user experiences in VR. We’ve compiled some advanced JavaScript tips you can use while scripting for High Fidelity.

You can use the Scripting Console in Interface to try out the examples on this page. The output will be visible in the console itself.

On This Page:

- Compute 3D Math Operations
Compute 3D Math Operations

When you script for VR worlds like High Fidelity, you need 3D math operations to compute the position and orientation of 3D objects and avatars in-world. We cannot simply add two vectors. To script 3D math operations and to determine position and orientation information of avatars, you can use the following namespaces in our JavaScript API:

- **Vec3**: The Vec3 API has facilities for generating and manipulating 3-dimensional vectors.
- **Quat**: The Quat API provides facilities for generating and manipulating quaternions.
- **MyAvatar**: The MyAvatar API provides facilities for manipulating avatars.

Get Your Avatar’s Position

When creating objects in world, it’s often very helpful to know where your avatar currently is. High Fidelity uses a 3D Cartesian coordinate system where the position vector of an entity or avatar looks like this:

```javascript
{ x: 0, y: 0, z: 0 }
```

To get your avatar’s current position, use the `MyAvatar` namespace. `MyAvatar` contains properties with information related to your avatar. Use the position property, `MyAvatar.position`, which returns an object.

In the following example, we are using the `JSON.stringify` method to convert the JavaScript object (returned by `MyAvatar.position`) to a data string that can be sent over the server.

Open your Scripting Console and find your avatar’s position.

```javascript
JSON.stringify(MyAvatar.position);
// {"x": -10.349810600280762,"y": -9.55379867553711,"z": 11.861204147338867}
```

Get Your Avatar’s Orientation

If you want an object to appear in front of you, you need to know how your avatar is currently oriented in-world. Rotations are handled by a number-system called Quaternions. Quaternions help simplify calculations in three dimensional space. They add an extra dimension of ‘w’ and the values are normalized (-1,1).

Quaternions are represented in the form:

```javascript
{ x: 0, y: 0, z: 0, w: 1 }
```

Get your avatar’s orientation in the Scripting Console by using the `MyAvatar.orientation` property:
JSON.stringify(MyAvatar.orientation);  
// {"x":0,"y":-0.4974651634693146,"z":0,"w":0.8674839735031128}

### Get the Direction Your Avatar is Facing

You can use the Quat namespace to get the direction which your avatar is facing. Pass your avatar’s orientation to `Quat.getForward` to get a vector describing which direction you are facing on the world axis.

```javascript
{ x: 0, y: 0, z: 1 } // Backward
{ x: 0, y: 0, z: -1 } // Forward
{ x: -1, y: 0, z: 0 } // Left
{ x: 1, y: 0, z: 0 } // Right
```

### Make an Entity Appear Before Your Avatar

You can make an entity appear before your avatar and also control the distance at which it appears.

Use the Vec3 namespace and the direction your avatar is facing to return the position at which you can make your entity appear. This position is 1m away from your avatar.

```javascript
Vec3.sum(MyAvatar.position, Quat.getForward(MyAvatar.orientation)); // This will add your position vector to the direction vector returned from Quat.getForward. This will represent a position that is 1 meter in front of your avatar.
```

You can also control the distance at which an entity appears rather than using the default distance of 1m. First, multiply the vector representing the direction your avatar is facing and the distance.

```javascript
Vec3.multiply(Quat.getForward(MyAvatar.orientation), 2.0); // if we are facing forward, that means our vector { x: 0, y: 0, z: -1 }, get's multiplied by 2.0, giving us a vector of { x: 0, y: 0, z: -2 }
```

Use `Vec3.sum` to return a new vector representing how far away an entity will appear from your avatar.

```javascript
Vec3.sum(MyAvatar.position, Vec3.multiply(Quat.getForward(MyAvatar.orientation, 2.0))); // this will give us a final vector representing where in the world a point 2 meters directly in front of our avatar is
```

We’ve included the above operations in a function below for you to save and run as a script.

```javascript
var myPosition = MyAvatar.position;
var myOrientation = MyAvatar.orientation;
var myDirection = Quat.getForward(myOrientation);
var distanceInFrontOfMe = 2.0;
var distanceVectorOfObjectInFrontOfMe = Vec3.multiply(myDirection, distanceInFrontOfMe);
var positionOfObjectInFrontOfMe = Vec3.sum(myPosition, distanceVectorOfObjectInFrontOfMe);

// we can even wrap this all up in a function to help simplify this any time we want.

function getPositionInFrontOfMe(distanceInFrontOfMe) {
  var myPosition = MyAvatar.position;
  var myOrientation = MyAvatar.orientation;
  var myDirection = Quat.getForward(myOrientation);

  var positionOfObjectInFrontOfMe = Vec3.sum(myPosition, distanceInFrontOfMe);
```
Include External JS and JSON Files

When writing a script in High Fidelity, you might need to access the methods or objects in an external JS file or get information from a JSON file. For example, if you’re writing a script to make your avatar wave, you might need to use some methods that already exist in an external JS file. You can do this using the `require` method in the Scripts namespace of our API.

Any script that you try to retrieve using this method must export either a function or an object. Let’s try this using an example.

Create a JS script that you want to access from your main script.

`example.js`

```javascript
module.exports = {
  sayHello: function () {
    console.log("Hello!");
  },
  sayGoodbye: function () {
    console.log("Goodbye!");
  }
};
```

In `example.js`, you are exporting two functions that print either Hello or Goodbye, depending on which function you call, to the console. Now, let’s use `require` in your main script.

Create a JS script called `main.js`.

```javascript
var greet = Script.require(Script.resolvePath("example.js"));
greet.sayHello(); // prints Hello to the console
```

When you use the `require` method, you are making any function or object exported from `example.js` available to `main.js`. This means that `main.js` now has access to functions that will print either Hello or Goodbye to the console. In the above example, we are printing Hello to the console when we run `main.js`.

Equip an Item

You can equip an item by grabbing and holding an entity without pressing the grab button or trigger continuously. For example, you could equip a paint brush to your avatar’s hand and drop it only when you’re done painting.

You can equip an item using a script:

```javascript
Messages.sendLocalMessage('Hifi-Hand-Grab', JSON.stringify({hand: 'XXX', entityId: 'YYY'})); \ where XXX is either the left or right hand and YYY is entityId to equip
```

To drop the entity from your avatar’s hand:
Connect a Signal to a Function

Signals can be connected to functions. This means that every time a signal is triggered, a function is executed. For example, if your avatar changes when collisions are enabled or disabled, you can connect a function to react to this specific event such as:

```javascript
function collisionChanged(enabled) {
    if (enabled) {
        console.log("avatar collision is enabled");
    } else {
        console.log("avatar collision id disabled")
    }
}
MyAvatar.collisionsEnabledChanged.connect(collisionChanged);
```

Each signal usually gets passed in arguments, and you can refer to the API documentation to see what a signal will provide you, such as the enabled property passed into collision changed.

It’s good practice to disconnect from signals, but you can only do that if you name your function.

```javascript
MyAvatar.collsionEnabledChanged.disconnect(collisionChanged);
```

See Also

- Write Your Own Scripts
- API Reference

### 1.4.4 Interface Scripts

Interface scripts run on your local machine, as long as you have Interface up and running. Each user has full control over when interface scripts are started and stopped. The results (such as when your script changes the color of a box) can be seen by everyone in your domain, but the script itself is running on your local machine. Your Interface will communicate that information to the Entity Server, which will communicate it to whoever is visiting the domain.

With interface scripts, you can do things like add new menus to the Interface, add plug-ins, or add 3D overlays that you have control over.

On This Page

- Load an Interface Script
- Example of an Interface Script

**Load an Interface Script**

To load and run an interface script:

1. In Interface, go to **Edit > Running Scripts** or press Ctrl + J on your keyboard.
2. For scripts hosted in the cloud, click ‘From URL’. Enter the URL of your script file and click ‘OK’.
3. For scripts on your local computer, click ‘From Disk’. Browse to your script file and click ‘Open’.
Example of an Interface Script

The following script automatically enters a first person perspective when you enter VR mode with a HMD.

```javascript
(function() {

  // Automatically enter first person mode when entering HMD mode
  HMD.displayModeChanged.connect(function(isHMDMode) {
    if (isHMDMode) {
      Camera.setModeString("first person");
    }
  });

}());
```

See Also

- Get Started with Scripting
- Write Your Own Scripts

1.4.5 Assignment Client Scripts

Assignment Client (AC) scripts (also known as “persistent scripts”) run persistently in a domain and aren’t affected by other scripts. These scripts run on an assignment client separate from the Interface, so the script will continue to run until you either remove the script from the domain or you shut down the domain entirely.

With AC scripts, you can do things like coordinate actions between entities and avatars, and add virtual pets to greet visitors to your domain.

On This Page

- Add an AC Script
- Example of an AC Script

Add an AC Script

Once you’ve written and hosted your script, you need to add it to a domain, either your own or one where you have permissions to run an AC script.

1. Open your ‘Domain Administration Panel’. If you are on a local sandbox, open it by clicking on the High Fidelity icon in the taskbar notifications and ‘click Settings’.

2. From the menu, go to Content > Scripts.

3. In the Persistent Scripts section, click + and paste the URL to your script under ‘Script URL’.
4. At the top of the page, click ‘Save and Restart’. Now, every time you enter that domain, the AC script will be running.

Example of an AC Script

The following script counts the number of entities found in a domain using High Fidelity’s EntityViewer.

```javascript
var SEARCH_CENTER = {x: 0, y: -10, z: 0};
var SEARCH_RADIUS = 100;
var isInitialized = false;
var timeout = 1000;

var update = function(deltaTime) {
    if (!isInitialized) {
        if (Entities.serversExist() && Entities.canRez()) {
            EntityViewer.setPosition(SEARCH_CENTER);
            EntityViewer.setCenterRadius(SEARCH_RADIUS);
            EntityViewer.queryOctree();

            Script.setTimeout(function() {
                var foundEntities = Entities.findEntities(SEARCH_CENTER, SEARCH_RADIUS).length;

                print("AC Script found: " + foundEntities + " entities within " + SEARCH_RADIUS + "m of " + JSON.stringify(SEARCH_CENTER));

                }, timeout);

                isInitialized = true;
                Script.update.disconnect(update);
            }
        }
    }

Script.update.connect(update);
```

See Also

- Configure Your Domain Settings
- Get Started with Scripting
- Write Your Own Scripts
1.4.6 Avatar Scripts

Avatar scripts are bound to an avatar. This means that they run when a user puts on a specific avatar. Likewise, avatar scripts stop running when the avatar is removed or changed. Other users in the domain will be able to see the script in action, but they will not be able to run the script themselves.

With avatar scripts, you can do things like make your hair flow or create particle clouds around your avatar.

On This Page

- Add an Avatar Script
- Example of an Avatar Script

Add an Avatar Script

There are two different ways you can add an avatar script to your FST file: either by using our Package Model tool or by manually adding the script.

To add an avatar script using the Package Model tools:

1. Create a folder called `scripts` in the same location as your FBX file.
2. Copy your avatar script into this new folder.
3. In Interface, go to Edit > Package Model as .fst
4. For ‘Script Directory’, enter the path to the `scripts` folder you created above.

To add an avatar script manually:

1. Open the FST file for your avatar in the text editor of your choice.
2. Add a line telling the avatar where to find the script file using the syntax `script = [SCRIPT URL]`.
   
   You can add multiple scripts to your avatar by adding multiple `script = url` lines.

Example of an Avatar Script

The following script makes your avatar throw balls when its right hand moves.

```plaintext
name = mannequinHairTest8
type = body+head
scale = 1
filename = mannequinHairTest8/mannequinHairTest8.fbx
texdir = mannequinHairTest8/textures
script = mannequinHairTest8/scripts/flow.js
joint = jointRoot = Hips
joint = jointLeftHand = LeftHand
joint = jointHead = Head
joint = jointLean = Spine
joint = jointEyeLeft = LeftEye
joint = jointEyeRight = RightEye
joint = jointRightHand = RightHand
joint = jointNeck = Neck

URL].
```
function() {
  var triggerDistance = 0.0;
  var TRIGGER_THRESHOLD = 0.9;
  var LOAD_THRESHOLD = 0.6
  var init = false;
  var rightHandIndex = MyAvatar.getJointIndex("RightHand");
  var rightArmIndex = MyAvatar.getJointIndex("RightArm");
  var distance = 0.0;
  var triggered = false;

  function fireBall(position, speed) {
    var baseID = Entities.addEntity({
      type: "Sphere",
      color: { blue: 128, green: 128, red: 20 },
      dimensions: { x: 0.1, y: 0.1, z: 0.1 },
      position: position,
      dynamic: true,
      collisionless: false,
      lifetime: 10,
      gravity: speed,
      userData: "{ "grabbableKey": { "grabbable": true, "kinematic": false } }"
    });
    Entities.editEntity(baseID, { velocity: speed });
  }

  Script.update.connect(function() {
    rightHandPos = MyAvatar.getJointPosition(rightHandIndex);
    rightArmPos = MyAvatar.getJointPosition(rightArmIndex);
    fireDir = Vec3.subtract(rightHandPos, rightArmPos);
    var distance = Vec3.length(fireDir);
    triggerDistance = distance > triggerDistance ? distance : triggerDistance;
    if (!triggered) {
      if (distance < LOAD_THRESHOLD * triggerDistance) {
        triggered = true;
      }
    } else if (distance > TRIGGER_THRESHOLD * triggerDistance) {
      triggered = false;
      fireBall(rightHandPos, Vec3.normalize(fireDir));
    }
  });

  MyAvatar.scaleChanged.connect(function () {
    triggerDistance = 0.0;
  });
}

This example script uses the MyAvatar namespace to determine if your avatar’s hand moves. Upon detecting movement, the script makes your avatar launch balls. It also uses some other namespaces such as Entities (to create the ball you will launch) and Vec3 (to determine the right positions and distances). Add it to your avatar to see how it works.

See Also
• Get Started with Scripting
• Write Your Own Scripts
• API Reference
1.4.7 Client Entity Scripts

You can make content in High Fidelity interactive by attaching scripts to entities. *Client entity scripts* are entity scripts that run locally on each user’s computer. When a user comes into contact with the entity, it will “preload” (or run) the script, then “unload” (or stop) the script when the user leaves.

There can be (and typically are) multiple entities in a domain, and each one can have a different client entity script associated with it.

**On This Page**

- Attach a Client Entity Script to an Entity
- Example of a Client Entity Script

**Attach a Client Entity Script to an Entity**

To attach a client entity script to an entity:

1. In Interface, pull up your tablet or HUD and go to Create.
2. Select the entity you’d like to script by either clicking on it in Interface or finding it in the ‘Entity List’.
3. In the Create app, go to the ‘Properties’ tab and scroll down to the ‘Behavior’ section.
4. For Script, enter the URL to your client entity script.

**Example of a Client Entity Script**

The following script changes the color of a non-model entity (such as a box or a sphere) when you click on it:

```javascript
(function () {
  var clicked = false;
  this.clickDownOnEntity = function (entityID, mouseEvent) {
    if (clicked){
      Entities.editEntity(entityID, { color: { red: 0, green: 255, blue: 255} }
    } else {
      Entities.editEntity(entityID, { color: { red: 255, green: 255, blue: 0} }
    };
    clicked = !clicked;
  };
});
```

This example is written as a JavaScript class prototype function, and it uses the mouse event `clickDownOnEntity()`. When the user clicks on an entity, `clickDownOnEntity()` triggers the function associated with that click event. In this case, it changes the entity’s color back and forth between yellow and magenta.

**See Also**

- Get Started with Scripting
- Write Your Own Scripts
- API Reference
1.4.8 Server Entity Scripts

You can make content in High Fidelity interactive by attaching scripts to entities. *Server entity scripts* are entity scripts that run on the server (or domain) that hosts the entity. These scripts run persistently in a domain, even if there are no users present. This means that there is only one instance of the script running at a time, and it is running on the server. Any behavior that is controlled by your script will be seen and heard by everyone in the domain.

**On This Page**

- Attach a Server Entity Script to an Entity
- Example of a Server Entity Script
- Script API

**Attach a Server Entity Script to an Entity**

To attach a server entity script to an entity:

1. In Interface, pull up your tablet or HUD and go to Create.
2. Select the entity you’d like to script by either clicking on it in Interface or finding it in the ‘Entity List’.
3. In the Create app, go to the ‘Properties’ tab and scroll down to the ‘Behavior’ section.
4. For ‘Server Script’, enter the URL to your server entity script.

**Example of a Server Entity Script**

The following script modifies the intensity of a light entity, so that it flickers tea lights.

```javascript
(function() {
  var MINIMUM_LIGHT_INTENSITY = 100.0;
  var MAXIMUM_LIGHT_INTENSITY = 125.0;

  // Return a random number between 'low' (inclusive) and 'high' (exclusive)
  function randFloat(low, high) {
    return low + Math.random() * (high - low);
  }

  var self = this;
  this.preload = function(entityID) {
    self.intervalID = Script.setInterval(function() {
      Entities.editEntity(entityID, {
        intensity: randFloat(MINIMUM_LIGHT_INTENSITY, MAXIMUM_LIGHT_INTENSITY)
      });
    }, 100);
  }

  this.unload = function() {
    Script.clearInterval(self.intervalID);
  }

}());
```

This script is a good example of a server entity script because it only needs one actor to update the intensity of the light. The same script could be attached as an entity client script, but all clients who could see the tea light would be editing the entity to vary the intensity of the light to flicker it.
Script API

The Entity Script Server does not have access to all of the listed components of the API. APIs for avatars, controllers, recording, overlays, and mouse and keyboard events are not available in the Entity Script Server.

Learn more about what APIs are available to server entity scripts here.

See Also

- Get Started with Scripting
- Write Your Own Scripts
- API Reference

1.4.9 Tutorial: Transfer Money and Items

While you can transfer money and items using the Inventory app, sometimes you may want to use scripting to help automate your gifting process. This page will walk you through some examples that programmatically transfers money and items using the High Fidelity Commerce APIs.

On This Page:

- Transfer Money to Someone
- Transfer Marketplace Items
- Purchase Marketplace Items
- Verify Your Inventory
- Add a Tip Jar
- Create a VIP Access Zone
- Add a Slot Machine Game

Transfer Money to Someone

To transfer money to someone, you need to use an entity script or client script to open an end-user’s tablet to the “Send Money” screen. The script must specify a recipient and an amount of HFC. It can specify a message to the user if desired.

```javascript
var tablet = Tablet.getTablet("com.highfidelity.interface.tablet.system");
tablet.loadQMLSource("hifi/commerce/common/sendAsset/SendAsset.qml");
tablet.sendToQml({method: 'updateSendAssetQML',
  assetCertID: "]”,
  amount: "10",
  username: "steve",
  message: ”Pay me 10 HFC and I will unlock this door!”
});
```
Transfer Marketplace Items

To transfer Marketplace items to someone, you need to use an entity script or client script to open an end-user’s tablet to the “Send Item” screen. The script must specify a recipient and an item’s Certificate ID. The user running the script must own the specified item Certificate ID. Optionally, the script can specify a message to the user if desired.

```javascript
var tablet = Tablet.getTablet("com.highfidelity.interface.tablet.system");
tablet.loadQMLSource("hifi/commerce/common/sendAsset/SendAsset.qml");
tablet.sendToQml({method: 'updateSendAssetQML',
    assetCertID: "ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890", // This is a fake Certificate ID!
    amount: "1", // Amount will always be "1" regardless of what is specified here
    username: "steve",
    message: "Send me the item you've previously bought!")
});
```

Purchase Marketplace Items

To purchase a Marketplace item for yourself, you need to use an entity script or client script to open an end-user’s tablet to the “Marketplace Checkout” screen. The script must specify the Marketplace Item ID.

```javascript
var tablet = Tablet.getTablet("com.highfidelity.interface.tablet.system");
tablet.loadQMLSource("hifi/commerce/checkout/Checkout.qml");
tablet.sendToQml({method: 'updateCheckoutQMLItemID', params: {itemId: "399921f6-bf26-4bba-8654-75d1b30f9442"}});
```
Verify Your Inventory

If a script has the proper credentials, it can check a user’s Recent Activity and Inventory on their behalf. You can use this to verify that another user has sent you money, sent you an item, or purchased your item from the Marketplace. This feature is used in the tutorial to create a VIP zone.

Add a Tip Jar

In this example, we walk through creating an entity that, when clicked, will prompt a user to send you 10 HFC. This tutorial uses a client entity script to transfer money.

Here are the steps for this tutorial:

- Create a “Tip Jar” in Your Domain
- Write an Entity Script for the Tip Jar
- Add the Entity Script to the Tip Jar

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
- Get Started with Scripting
- Asset Server

Create a Tip Jar in Your Domain

You need a jar that accepts money. In this simple example, we will use a cube entity as our tip jar. Create your tip jar in the Create app:

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the cube icon to create a cube entity.
3. Move the Tip Jar to where you want it.
Write an Entity Script for the Tip Jar

We need to write a script to put on the Tip Jar entity or cube. When a user clicks the cube, a message will prompt them to pay a specified username (you) 10 HFC. The message will be, “Here’s a 10 HFC tip for doing a cool thing!”.

Click here to download a pre-made “Tip Jar” entity script. Follow along with the comments in the code to understand what it’s doing!

Add the Entity Script to the Tip Jar

You’ll have to add the entity script to the Tip Jar cube entity.

1. Change the DESTINATION_USERNAME variable in tipJar.js to match your username.
2. Upload the tipJar.js script to your domain’s ATP server.
   - In Interface, go to Edit > Asset Browser > Choose File.
   - Right-click the script file, then select ‘Copy URL’.
3. In Interface, pull up your HUD or Tablet and go to Create.
4. In the ‘Entity List’, select the Tip Jar cube entity.
5. In the entity’s ‘Properties’, scroll down to ‘Script’ and paste the URL you copied, into the text box. Press Enter.
6. Lock the entity so nobody can change its attributes.

Close the Create app and click on the Tip Jar cube entity. A window pops up, prompting you to pay 10 HFC to the username specified in the script.

Create a VIP Access Zone

In this example, we walk through creating an exclusive VIP zone that is accessible only to paid VIPs. This tutorial uses a client entity script to transfer money and an assignment client script to verify payments.

Here are the steps for this tutorial:

- Place a VIP Zone in Your Domain
- Write an Entity Script for the VIP Zone
- Add the Entity Script to the VIP Zone
- Obtain the Auth Token
- Write an Authenticated AC Script
- Run the AC Script in Your Domain
- Optional: Create a Box to Accept Payments

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
- Get Started with Scripting
- Asset Server
Place a VIP Zone in Your Domain

You need an area (or zone) in your domain that will be designated the “VIP Zone”. Only users who have paid you 10 HFC in the current server session have access to enter this zone. If the server restarts, users will have to pay for VIP status again.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click the ‘Zone’ icon to create a zone entity.
3. Move the zone to where you want it.

Write an Entity Script for the VIP Zone

We need to write an entity script to put on the VIP Zone. This script will check to see whether the user entering the zone is a VIP or a domain admin. If they aren’t, the script will remove them from the zone. In addition, the script will change a user’s status to VIP if they pay 10 HFC.

Click here to download a pre-made “VIP Zone” entity script. Follow along with the comments in the code to understand what it’s doing!

Add the Entity Script to the VIP Zone

To add the entity script to the VIP zone:

1. Upload the vipZoneEntityScript.js script to your domain’s ATP server.
   - In Interface, go to Edit > Asset Browser > Choose File.
   - Right-click the script file, then click ‘Copy URL’.
2. In Interface, pull up your HUD or Tablet and go to Create.
3. In the ‘Entity List’, select the ‘VIP Zone’ entity.
4. In the entity’s ‘Properties’, scroll down to ‘Script’ and paste the URL you copied into the text box. Press Enter.
5. Lock the zone entity so nobody can change its attributes.

Obtain the Auth Token

You need to get a High Fidelity authentication token that has the commerce scope. You will use this token while writing an Assignment Client (AC) script to check your Recent Activity for recent transactions of 10 HFC made in your domain.

To obtain this auth token:

2. Name the token something memorable.
3. Select the commerce scope.
4. Click ‘Create Token’.
5. Copy and save the token.
Write an Authenticated AC Script

Now, write an Assignment Client (AC) script containing the authentication token you copied and saved. This AC script checks your Recent Activity for recent transactions of 10 HFC made in your domain.

Click here to download a pre-made “VIP Zone” AC script. Follow along with the comments in the code to understand what it’s doing!

Run the AC Script in Your Domain

To run the above AC script in your domain from ATP:

1. Set HIFI_COMMERCE_TOKEN to the token you saved in the vipZoneACScript.js script.
2. Upload your vipZoneACScript.js script to your domain’s ATP server. In Interface, go to Edit > Asset Browser > Choose File. Right-click and select ‘Copy URL’.
3. Navigate to the Domain Settings page of your domain (for a local sandbox, this is http://localhost:40100/).
4. Click ‘Content’ at the top of the page, then scroll to the ‘Scripts’ section.
5. Under ‘Persistent Scripts’, click the + button on the right column.
6. Under ‘Script URL’, paste the ATP URL you copied.
7. Click ‘Save and restart’ at the top right of the page.

(Optional) Create a Box to Accept Payments

This step is optional as it doesn’t matter how a user sends you 10 HFC to earn VIP status. For example, if a user (in Interface) went to Inventory > Send Money > Nearby to send you 10 HFC while you were in your domain, they would still get “VIP status”.

However, to make it easier to people to pay you, you can create an box that collects payment. Simply follow the directions to create a tip jar above. You should now have a working “VIP Zone” in your domain. Only users with VIP status can enter this “VIP Zone”. You should also have a “Tip Cube” in your domain that helps users pay you HFC to become VIPs.

Add a Slot Machine Game

In this example, we walk through creating a slot machine game that pays out HFC. Players will pay you (the domain owner) 1 HFC to start playing, and the slot machine will pay out 25 HFC if the payer wins. This tutorial uses coupons to reserve HFC ahead of time, client entity scripts to control the mechanics of the slot machine, and an assignment client script to handle the slot machine game logic.

Here are the steps for this tutorial:

- Create a Slot Machine in Your Domain
- Create a Coupon to Authorize Winnings
- Add the Coupon Credentials to a Database
- Allow Users to Start the Slot Machine
- Add the Entity Script to the Slot Machine
- Allow Users to Pull the Reels
Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:

- Create New Entities
- Get Started with Scripting

Create a Slot Machine in Your Domain

You need a slot machine that your users can play. You can create your own or use the one that we created for you.

To use the one that already exists:

1. Download the following JSON: basicSlotMachine_noScripts.json
2. In Interface, pull up your tablet or HUD and go to Create.
3. In the Create Tools app, click ‘Import Entities’. Browse to and select basicSlotMachine_noScripts.json.

You should now see a slot machine entity in your domain. This example entity consists of:

- Three “reels” (red, green, and blue cubes).
- A “spin arm” that players will use to start the game.
- A “play text” text entity that will display the game status to players.
- A “pay-in text” text entity that will instruct users to add credits to the slot machine.

Create a Coupon to Authorize Winnings

By creating a coupon, you are authorizing High Fidelity to take out funds from your account even if you are not present in the domain. Follow these instructions to create a coupon. Copy the Authorization ID and the Coupon ID.

Add the Coupon Credentials to a Database

Later, you will write an Assignment Client (AC) script for the Slot Machine game logic, including payout logic. When the Slot Machine pays out, it needs to know the Authorization IDs and Coupon IDs associated with your pre-authorized payout funds.

In this step, you will put the “Authorization ID” and “Coupon ID” into some sort of database. Here, we use a Google Sheet to store the data, and a Google Script to access the data in the Sheet.

To create the Google sheet:

1. Log into Google Sheets, and create a new spreadsheet. Give it a filename you want, such as “Slot Machine Payouts”.
2. Name the current sheet “Authorizations” using the arrow on the tab at the bottom left of the screen.
3. Give the header row (the first row) the following labels in this order:
• Used
• HFC
• Authorization ID
• Coupon ID

4. In the second row, under the HFC column, put 25.

5. In the second row, under the ‘Authorization ID’ column, paste your saved Authorization ID.

6. In the second row, under the ‘Coupon ID’ column, paste your saved Coupon ID.

To create the Google script

1. At the top of the Google Sheets window, click Tools > Script editor.

2. Name your currently untitled project “Slot Machine Authorization Handler”.

3. Copy and paste the contents of this example GS script into the Script Editor.

4. Change var SPREADSHEET_ID to match the Spreadsheet ID of your spreadsheet above.

5. Save the script, using whatever filename you wish.

6. Click ‘Publish’, then ‘Deploy as Web App…’

7. Follow Google’s instructions to deploy your script as a web app. Ensure you set ‘Who has access to the app’ to ‘Anyone, even anonymous’. When finished, copy the URL you’re given at the end of the process and save it somewhere you’ll remember for later. The web app URL will look something like https://script.google.com/macros/s/ABCDEFGHIJKLMNOP_QRSTUVWXYZ1984373/exec

Allow Users to Start the Slot Machine

You need to provide your users with a way to add slot machine play credits. You can do this by adding a client entity script to the Slot Machine entity.

This script will display a message “1 Slot Machine Play Credit” when they click the text or border around the text on the slot machine. They can pay the specified user (you) 1 HFC to play.

Click here to download a pre-made “Add Credits” entity script. Follow along with the comments in the code to understand what it’s doing!

Add the Entity Script to the Slot Machine

To add the entity script to the slot machine:

1. Change the DESTINATION_USERNAME to your username in addCreditsButton.js.

2. Upload the addCreditsButton.js script to your domain’s ATP server. In Interface, go to Edit > Asset Browser > Choose File. Right-click the script file, then click ‘Copy URL’.

3. In Interface, use the Create app to select the ‘Click Here to Add Credits’ text entity on the Slot Machine entity.

4. In the entity’s ‘Properties’ tab, scroll down to ‘Script’ and paste the URL you copied in step 2 into the text box. Press Enter.
5. Lock the entity so nobody can change its attributes.

6. In Interface, use the Create app to select the border entity around the ‘Click Here to Add Credits’ button on the Slot Machine entity.

7. In the entity’s ‘Properties’ tab, scroll down to ‘Script’ and paste the URL you copied in step 2 into the text box. Press Enter.

8. Lock the entity so nobody can change its attributes.

Allow Users to Pull the Reels

Next, you need to provide your users with a way to start the slot machine’s reels. Here, we will write an entity script to put on the slot machine’s Spin Lever. This script will send a message to an Assignment Client (AC) script to kick off the rest of the game logic.

Click here to download a pre-made “Spin Lever” entity script. Follow along with the comments in the code to understand what it’s doing!

To add the entity script to the reels:

1. Upload the slotMachineSpinLever.js script to your domain’s ATP server. In Interface, go to Edit > Asset Browser > Choose File. Right-click the script file, then click ‘Copy URL’.

2. In Interface, use the Create app to select the red Spin Lever sphere entity on the Slot Machine entity.

3. In the entity’s ‘Properties’ tab, scroll down to ‘Script’ and paste the URL from step 1 into the text box. Press Enter.

4. Lock the entity so nobody can change its attributes.

Obtain Auth Token

You’ll have to get a High Fidelity authentication token that has the commerce_ro scope. You will use this token while writing an Assignment Client (AC) script to check your Recent Activity for recent transactions of 1 HFC made in your domain with a specific memo (“1 Slot Machine Play Credit”).

To get this auth token:


2. Name the token something memorable.

3. Select the commerce_ro scope.

4. Click ‘Create Token’.

5. Copy and save the token.

Write a Game Logic AC Script

Now, write an AC Script that will handle the slot machine game logic, including:

- Knowing when to start a new spin.
- Knowing whether a user who attempted to spin has paid.
- Changing the slot machine reel colors during a spin.
- Checking the end state of the reels to determine win/loss.
• Paying out pre-authorized funds.

Click here to download a pre-made “Slot Machine” entity server script. This script is quite long and is arguably the most important element of this project! Follow along with the comments in the code to understand what it’s doing.

Run the AC Script on Your Domain

To run the AC script on your domain from ATP:

1. Change your slotMachineACScript.js as follows:
   1. Set HIFI_COMMERCE_TOKEN to your HiFi commerce_ro token.
   2. Set SLOT_MACHINE_REEL_1_ID to the Entity ID of the leftmost slot machine reel.
   3. Set SLOT_MACHINE_REEL_2_ID to the Entity ID of the middle slot machine reel.
   4. Set SLOT_MACHINE_REEL_3_ID to the Entity ID of the rightmost slot machine reel.
   5. Set SLOT_MACHINE_PLAY_TEXT_ID to the Entity ID of the “Play Text” text entity right below the slot machine reels.
   7. Set SLOT_MACHINE_AREA to the coordinates around which the slot machine entity will be placed. See the comments in the code for more details about why this is necessary.

2. Upload your slotMachineACScript.js script to your domain’s ATP server. In Interface, go to Edit > Asset Browser > Choose File. Right-click the script file, then click ‘Copy URL’.

3. Navigate to the Domain Settings page of your domain (for a local sandbox, this is http://localhost:40100/).

4. Click ‘Content’ at the top of the page, then scroll to the ‘Scripts’ section.

5. Under ‘Persistent Scripts’, click the + button on the right column


7. Click ‘Save and restart’ at the top right of the page

You should now have a basic but fully working slot machine in your domain, that you and anyone else in your domain can play.

See Also

• Get Started with Scripting
• Write Your Own Scripts
• API Reference
• Create a Coupon

1.4.10 Tutorial: Use MIDI to Control Your Environment

MIDI (Musical Instrument Digital Interface) is a protocol (with electrical connectors and a digital interface) that allows digital tools and electronic devices (virtual and physical) to communicate with each other. MIDI is usually used as a music sequencer. Originally, this format was created in the 80s as a way for instruments to communicate with each other, but over the last 30 years, it has evolved into a highly organized specification that is heavily tested and adopted for a multitude of purposes.

We created a MIDI class (with the help of one of our community members, Brainstormer) that can be used to control your environment in High Fidelity. For example, we used MIDI to control lighting in a domain for a music show.
Note: Currently, we support the MIDI class only on Windows.

On This Page

- Tutorial: Use MIDI to Control Your Environment
  - MIDI Class Basics
  - Connect Your Controller Device
  - Configure Your MIDI Device
  - Example: Change the Color of a Cube using MIDI
  - Other Ways to Use MIDI in High Fidelity

MIDI Class Basics

Our MIDI class works by passing a DWORD (double word), a data type specific to Microsoft Windows, as a message. It is an unsigned 32-bit unit of data and can contain an integer value ranging from 0 to 4,294,967,295.

Every time you move a lever, rotate a knob, press/release a key, or push down a pad, you are creating a MIDI message that says what channel, what note, what velocity, and what is the status/command to run.

Each byte in this message describes a different type’s value.

```
00000000 0vvvvvv 0nnnnnn 1ss  ccc
```

Where:
- \( v \) = velocity
- \( n \) = notes
- \( s \) = status
- \( c \) = channel

The number in the higher order bit (the first number) denotes whether it is a command (1) or data (0). The rest of the numbers determine the value of the type. This means that the velocity and note can represent 128 unique values \((1+2+4+8+16+32+64)\), status can represent 8 unique values, and channel can represent 16 values.

The different status types we support are:

<table>
<thead>
<tr>
<th>Status</th>
<th>Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>08</td>
<td>note off</td>
</tr>
<tr>
<td>09</td>
<td>note on</td>
</tr>
<tr>
<td>10</td>
<td>polyphonic key pressure</td>
</tr>
<tr>
<td>11</td>
<td>control change</td>
</tr>
<tr>
<td>12</td>
<td>program change</td>
</tr>
<tr>
<td>13</td>
<td>channel pressure</td>
</tr>
<tr>
<td>14</td>
<td>pitch bend</td>
</tr>
<tr>
<td>15</td>
<td>system message</td>
</tr>
</tbody>
</table>
Connect Your Controller Device

You can either connect a real controller device that you use to control your environment in High Fidelity, or create a virtual one that will help you connect to other virtual devices.

Connect Ableton Live to Interface

To connect Ableton Live directly to High Fidelity’s Interface client, we recommend the following virtual tools:

- **loopMIDI**: This will create a virtual in/out port to send information into and out of HiFi
- **VMP**: You can use this to simulate keys being pressed or sliders/knobs being manipulated if you do not have a controller.

Connect an iPad or iPhone to Interface

You can use your iPad or iPhone as a touch screen controller with buttons, knobs, and sliders. Read the sections *Configure Your MIDI Device* and *Example: Change the Color of a Cube using MIDI* before reading these instructions.

- Download our recommended app touchosc.
- Download the Windows bridge.
- Set up either through USB or through your local WIFI network in the settings menu.
- If you setup your `onEventReceived` to log the messages coming in, you can see which knobs send what information that you can use to call custom functions.
- There are some interesting components like the accelerometer which you can use as well!

Configure Your MIDI Device

Once you’ve set up your MIDI Controller Device, you need to configure it.

1. Here is a general recommended MIDI config function you can run in a script. With it, you can see a list of MIDI devices that are currently connected:

```javascript
// Some helpful constants
const INPUT = false;
const OUTPUT = true;
const ENABLE = true;
const DISABLE = false;
function midiConfig(){
    Midi.thruModeEnable(DISABLE );
    Midi.broadcastEnable(DISABLE );
    Midi.typeNoteOffEnable(ENABLE );
    Midi.typeNoteOnEnable(ENABLE );
    Midi.typePolyKeyPressureEnable(DISABLE);
    Midi.typeControlChangeEnable(ENABLE);
    Midi.typeProgramChangeEnable(ENABLE);
    Midi.typeChanPressureEnable(DISABLE);
    Midi.typePitchBendEnable(ENABLE);
    Midi.typeSystemMessageEnable(DISABLE);

    // get a list of the available in and out device IDs
    midiInDeviceList = Midi.listDevices(INPUT);
}
```

(continues on next page)
2. After you run the configuration function, you will want to connect to `midiMessages`:

```javascript
Midi.midiMessage.connect(onEventReceived);
// Your message handler will look like the following:
// @param {int} device: device number
// @param {int} channel: channel number
// @param {int} type: 0x8 is noteoff, 0x9 is noteon (if velocity=0, noteoff), etc
// @param {int} note: MIDI note number
// @param {int} velocity: note velocity (0 means noteoff)

function onEventReceived(eventData){
  // functions you run in response to different MIDI events
}
```

**Example: Change the Color of a Cube using MIDI**

Let’s change the color of a cube entity in High Fidelity using MIDI.

1. Use this method to figure out the MIDI range of 0 to 127 to be any other output range you want using linear interpolation:

```javascript
function lerp(InputLow, InputHigh, OutputLow, OutputHigh, Input) {
  return ((Input - InputLow) / (InputHigh - InputLow)) * (OutputHigh - OutputLow) + OutputLow;
}
lerp (0,127,0,360,eventData.velocity); // the 0 would be 0, and the 127 would be 360.
```

2. Since colors go from 0 to 255, we could do the following:

```javascript
var red = 0;
function changeCubeColor(redValue){
  var entityColorProps = Entities.getEntityProps(cubeID, ["color"]).color;
  entityColorProps.red = redValue;
  Entities.editEntity(cubeID, entityColorProps);
}
```

3. Then use `onEventReceived` to change the color of the cube:

```javascript
// eventData.device, eventData.channel, eventData.type, eventData.note, eventData.velocity

function onEventReceived(eventData){
  changeCubeColor( lerp(0,127,0,255,eventData.velocity) );
}
```

Print the `eventData` in your `onEventReceived` function to see each controller and its output. This will tell you everything you need to know about how to route the right key, slider, knob, or button to your intended JavaScript functions.

If you want to use to control something outside of High Fidelity, or to directly call a MIDI event to control something in Hifi, you can use the function:
// event similar to the above
Midi.playNote(Status, Note, Velocity);

Other Ways to Use MIDI in High Fidelity

- Use Ableton to sequence out entire animations of your domain.
- Control real world devices by the movements things make in Hifi and vice versa (think update loop)
- Setup your iPad to be a whole group of buttons that you can press at any time to trigger events in your domain at will.

See Also
- API Reference: MIDI
- MIDI-API
- MIDI-Test
- MIDI-Examples

1.5 Host

High Fidelity enables you to host your own domain where you can create, add, and edit content and have visitors. We invite you to create your own High Fidelity experience by creating a domain tailored to suit your needs. You can create a domain to host an event, as a place to hangout with friends, or even as an office space.

Throughout this chapter, learn how to set up a domain, secure and host it, and make it accessible to other users.

1.5.1 Set Up Your Domain

Your domain is your own personal virtual world. In it, you can express your creativity and host your own content, with the option to share it with the world. Before you start creating content, you will need to install, configure, and secure your domain to control how other users will interact with your environment.

In This Section:

Install Your Domain

High Fidelity has a local sandbox that you can download and start using as your home domain. This sandbox is hosted on your local machine’s server.

On This Page:
- Install Your Sandbox
- Visit Your Sandbox

Install Your Sandbox

You can install the sandbox if your system meets the minimum requirements. To install the sandbox:

1. Go to High Fidelity’s website and download the ‘Client + Sandbox’ installer.
2. Run the High Fidelity installer.
3. Follow the prompts and complete the installation.

Visit Your Sandbox

Once you download and install the sandbox, you can access it in your system tray (Windows) or in the top menu bar (OS X).

To visit your sandbox:

1. Right-click the High Fidelity icon in your system tray or top menu bar.
2. On the menu that opens, click ‘Visit Sandbox’.

See Also

- Install High Fidelity

Configure Your Domain Settings

While you can use the sandbox as is, we recommend that you configure your domain settings to ensure that it is secure and serves your needs.

On This Page

- Configure Your Domain Settings
  - Configure Basic Server Settings
Configure Basic Server Settings

You can configure your domain’s basic server settings to control how your domain is accessed and the permissions granted to different types of users. To open your settings, do one of the following:

• Right-click the High Fidelity icon on your system tray (Windows) or top menu bar (OS X), then click ‘Settings’.
• With your Sandbox running, open http://localhost:40100/settings in a browser.

Setup Wizard

The first time you open your settings, your browser will open a Setup Wizard to help you configure your basic server settings. Click ‘Skip Wizard’ to close the wizard and manually configure your settings.

1. In Step 1 of the wizard, you can connect your domain to your High Fidelity account. You do not need to connect your domain to your High Fidelity account and can click ‘Skip’ and proceed to the next step. If you wish to connect to your account, click ‘Connect your High Fidelity account’.

   1. Log in to High Fidelity in the tab that opens.
   2. Click ‘Create Token’.
   3. Copy your new token.
   4. In the pop-up window on the settings page, paste the token and click ‘OK’ to connect your account.
2. In Step 2 of the wizard, you can change your domain’s place name, which is name people will use to get to your domain. Place names are randomly generated. For a fee, you can choose to customize your place name so that it more accurately describes your domain.

3. In Step 3 of the wizard, you can set user permissions for your domain. In this step, you can add yourself and other users as administrators of your domain. You can also grant or deny specific users the ability to connect to your domain or rez content. All of these permissions can be changed later, as you secure your domain.

4. Click ‘Finish’. You will be redirected to the ‘Domain Settings’ page.

Domain Server Settings

If this is not the first time you’ve opened your domain settings, you will not see the Setup Wizard. Instead, you can manually configure all of your domain settings.

For example, the ‘Metaverse/Networking’ section defines all of the network settings for your domain. Here, you can disconnect your High Fidelity account, view your access token, create a new domain ID, change your UDP port, and Enable Packet Verification.
If you modify your domain settings, you need to save and restart for the new settings to be reflected. Click ‘Save’, and then ‘Restart’ once you’re done with your modifications. This saves your settings and restarts your domain server.

Firewall Settings

Ensure that your firewall settings allow you to run High Fidelity and let other users visit your domain.

1. In your domain server settings, set the Local UDP port value. ‘Local UDP Port’ is the port that needs to be opened for High Fidelity to let users into your local sandbox. The default value for this port is ‘40102’.

2. Configure your firewall settings:
   - For Windows: In your firewall settings, open the port 40102, and add ‘interface.exe’ and ‘domain-server.exe’ to the list of allowed apps.
   - For Mac: In your firewall settings, add ‘interface.app’ (Library > Application Support > Launcher > interface.app) and ‘domain-server.app’ and allow incoming connections for those applications.

View Your Logs

Your logs lists the different events relevant to your High Fidelity domain. These can help you troubleshoot any issues with your domain and understand how the different parts of your domain communicate with each other. To view these:

1. Right-click the High Fidelity icon on your system tray or top menu bar.
2. Select ‘View Logs’.
A window pops up showing your domain server’s logs.

Add a Description For Your Domain

1. Open your domain settings.
2. Under the Description section, click ‘Advanced Settings’ and add the following:
   1. Description: A description of your domain in less than 256 characters.
   2. Maturity: A maturity rating for your domain. You can choose Everyone, Teen(13+), Mature(17+), and Adult(18+).
   3. Hosts: Add users who have permission to show your domain to other users.
   4. Tags: Common categories under which your domain falls.

Change Your Content Settings

Your content settings define the behavior of the content (or entities) in your domain. To get to the content settings, open your domain settings, then click on the ‘Content’ menu. This section describes the types of content settings you can define:

- **Content Archives**
- **Upload Content**
- **Paths**
- **Scripts**
- **Audio Environment**

Content Archives

Your domain server regularly backs up the content in your domain, with archives that include all the entities in your domain. The ‘Content Archives’ section lets you restore your domain to a previous version or to share your archive and settings with other users.

Each archive has a menu that you can access by clicking the three dots to the right of the archive name. Here, you can:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restore from here</td>
<td>Reset the domain to a saved state or archive.</td>
</tr>
<tr>
<td>Download</td>
<td>Compress all of the content in your domain to a downloadable ZIP file.</td>
</tr>
<tr>
<td>Delete</td>
<td>Remove the archive from the saved archives list.</td>
</tr>
</tbody>
</table>

To create a content archive manually, click ‘Generate New Archive’ under the list of automatic content archives.

Upload Content

You can upload content stored on your computer to your domain. Typically, you will use this option to load a content archive into your own domain, replacing all of its existing content. To do this:

1. Click ‘Choose File’ and locate the ZIP file that contains your content archive.
2. Click ‘Upload Content’ to upload a content archive.

Paths

A path is an exact location within your domain. By default, you have one path automatically assigned: the path “/” is the entry point where a user appears when they enter your domain. From the ‘Paths’ section, you can change the location of this default path, or add new ones to set multiple viewpoints within your domain.

Each path that you define has a set of coordinates, alongside a rotation to define the direction that clients will look when they go to that path.

To set a new path:

1. In your domain, move your avatar to the point where you wish to have users appear.
2. Go to Navigate > Copy Path to Clipboard. The path will be represented using 3D coordinates with orientation information, for example /-8.10251e-07,-11.1818,1.79641e-05/0,-0.708837,0,0.705372. This type of path is tedious to approximate and enter manually, making ‘Copy Path to Clipboard’ the more practical method of defining a path.
3. Open your cloud domain settings if you’re hosting it on a cloud service. Or, use http://localhost:40100/settings for a local sandbox. Click ‘Content’ and go to ‘Paths’ on the drop down menu.
4. Click the plus sign to add a new path. Paste your copied path under ‘Viewpoint’.
5. Click ‘Save’ at the top of the page.

To appear at a specific viewpoint in your domain, clients will enter the domain_name/path. For example, if a user is going to example-domain and enters it by typing in example-domain/taco, they will appear with the direction and rotation of the /taco path.

Scripts

The ‘Scripts’ section loads all of the assignment client scripts that you’ve written for your domain.

To add a script:

1. Click the plus sign next to an empty row in the ‘Persistent Scripts’ table.
2. Enter the URL to your script, the number of instances, and the pool.
3. Click ‘Save’ at the top of the page.
Note: Scripts are cached. If the content on the URL host changes, you may want to use some cache-defeating trick, such as specifying a URL with query parameter such as http://whatever.com/path/file.js?version=17

Audio Environment

Your audio settings are perhaps one of the most important things that define someone’s experience in your domain. Because of this, you will need to configure exactly how sound in transmitted and received throughout your domain. Here are some of the audio settings you can configure:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attenuation</td>
<td>This determines how much quieter sounds get over a distance. The default domain attenuation is the amount of noise reduction that is enabled across the domain environment. High Fidelity domains default to a distance attenuation curve roughly like the real world. If you see two avatars talking in the distance, you can hear them, but not very well. If you approach them, they become more audible in a manner that approximates what you’re likely to be used to. If the default attenuation is 0, no matter how far away a sound source is, it still plays at full volume. Likewise, the default attenuation for a domain can be set very high (to a max value of 1), making only things very near to you audible.</td>
</tr>
<tr>
<td>Zones</td>
<td>By setting a zone, you can specify 3D boundaries for audio environments that aren’t applied to the whole domain.</td>
</tr>
<tr>
<td>Attenuation Coefficients</td>
<td>These determine how sound is transmitted between zones. When the attenuation coefficient is 0, there is no volume attenuation with someone in another zone; at 1, you won’t be able to hear someone in another zone.</td>
</tr>
<tr>
<td>Reverb Settings</td>
<td>These settings enable echo-like effects in your domain. It can give the effect of sounding like you are in a large empty room, deep inside a large cave, or inside a tiny room like a tiled shower. The ‘Reverb Decay Time’ defines how long you can hear an echo after the initial sound. ‘Wet/Dry Mix’ sets the percentage mix of the reverb tail relative to the original “dry” signal. Levels between 5-25% will generally give you useful results. For a very thick reverb, you might try a value as high as 50% where the reverb is nearly as loud as the original signal.</td>
</tr>
</tbody>
</table>

Example: Building a Stage

You can add audio settings for a stage in your domain. This stage is used for performances and events.

1. Create two new zones, call them “Stage” and “Audience”.
2. Set the ‘Attenuation Coefficients’ to zero with Stage as the source and Audience as the listener. That way, anyone in Audience will hear anyone on Stage at maximum volume.
3. If you want to minimize people in the audience hearing one another in the Audience zone, set up another ‘Attenuation Coefficients’ pair with Audience as both source and listener, and set the distance attenuation very high—e.g., to 1.0. That way, you’ll only hear people very close to you like your friend on your right, but won’t hear the guy coughing 10 rows back.

See Also

- Secure Your Domain
- Broadcast to Other Domains
1.5.2 Secure Your Domain

You can protect your domain from being modified by other users using the domain server settings. By setting the right permissions and filters, you can ensure that your domain is secure.

Secure Your Domain Settings

Add a username and password to protect your domain settings from being modified by unauthorized users. Anyone who tries to modify your domain settings will need the username and password to make changes.

1. If you’re hosting your domain on a cloud service, open the cloud settings. If you’re using your local sandbox, open your local sandbox settings.
2. Go to Settings > Security and add a username and password under the ‘HTTP Username’ and ‘HTTP Password’ fields.

Every time someone tries to access your domain settings, they’ll be prompted to enter this username and password.

Set User Permissions

You can control the types of users that have access to your domain, and the permissions granted to them. For example, you can set a connection to have access to your domain, but not have permissions to edit your domain content.

1. If you’re hosting your domain on a cloud service, open the cloud settings. If you’re using your local sandbox, open your local sandbox settings.
2. Go to Settings > Security and scroll down to the ‘Domain-Wide User Permissions’. The ‘Standard Permissions’ table allows you to set the type of users and their permissions.

<table>
<thead>
<tr>
<th>User Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anonymous</td>
<td>A person who is not logged in and is using an instance of High Fidelity’s Interface. This is the default user type for someone who has downloaded Interface for the first time.</td>
</tr>
<tr>
<td>Friends</td>
<td>A user that you have connected with by shaking hands with their avatar and clicking on the ‘Friends’ checkbox in your People app. Connections are different from Friends. By default, Friends have more permissions in your domains than your Connections. Connections are treated like anonymous users.</td>
</tr>
<tr>
<td>Localhost</td>
<td>A user who is running Interface on the same machine where the server is hosted. Localhost users do not need to be logged in and have permissions that override non-user-specific permissions.</td>
</tr>
<tr>
<td>Logged In</td>
<td>A user that is logged into their High Fidelity account while using Interface. They do not need to be a Friend or a Connection to have server rights. Permissions that are granted to a specific user override all other permissions.</td>
</tr>
</tbody>
</table>
3. If you want to grant permissions only to specific users, you can do so under the ‘Permissions for Specific Users’ table. Add their usernames and check the required permissions.

4. You can set permissions for user groups and lists as well. Groups and lists can be created through the groups registration page.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Features</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>This is a collection of users that voluntarily join. Users join groups by invitation.</td>
<td>You can assign ranks to different users in a group and grant them permissions such as editing the group, editing group members, or editing the rank of another lower ranked member.</td>
<td>Granting/taking away permissions to a group or list will copy those permission rights to the users in the group even if you are not friends with them.</td>
</tr>
<tr>
<td>List</td>
<td>This is a collection of users who have been added to a list for the purpose of whitelisting or blacklisting them.</td>
<td>The list details are visible to and can be edited only by users with the right permissions.</td>
<td>Granting/taking away permissions to a group or list will copy those permission rights to the users in the group even if you are not friends with them.</td>
</tr>
</tbody>
</table>

**Protect Your Domain’s Content**

Entity-specific permissions prevent users from changing or removing your domain content. You can do this by locking individual entities in your domain, or adding entity filters in your domain settings.

**Lock an Entity in Your Domain**

When you lock an entity, you are preventing it from being modified by anyone who doesn’t have lock/unlock permissions in your domain. This means that if a user cannot unlock an entity, they cannot modify a locked entity.

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. Add a box entity by clicking on the ‘Cube’ icon.
3. Switch to the “List” submenu to search for your newly created entity using the filter bar.
4. You can disable editing, deleting, or moving the entity by choosing the lock icon from this menu or the “Properties” submenu.
5. Click the ‘Lock’ icon again to unlock the entity and make it editable.

**Note:** Anyone with connect permissions can make edits to your domain if they are not locked. Using filters is the only way to prevent unwanted edits.

**Set an Entity Filter**

Entity filters are specialized JavaScript functions that prevent unwanted modifications to entities in your domain. They can be applied to a whole server or to specific zones within a domain.

Entity filters apply to all users who do not have lock/unlock permissions. You can use filters to request the original properties of an entity to compare them to new values. You can then approve the new values.

**Protect All Entities in a Domain**

1. Write a script for an entity filter. Host the script on a cloud service.
2. If you’re hosting your domain on a cloud service, open the cloud settings. If you’re using your local sandbox, open your local sandbox settings.
3. Go to **Content > Entities** and add the URL for your script.
Protect All Entities in a Zone

You can add these scripts to a specific zone in your domain as well.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Find your zone entity, and click ‘Properties’.
3. Add the script’s URL in the ‘Filter’ field.

Examples of Entity Filters

1. Prevent all entities from being deleted from your domain:

```javascript
// prevent-all-deletes.js by Brad Hefta-Gaub
(function() {
    function filter() {
        return false; // all deletes are blocked
    }
    filter.wantsToFilterAdd = false; // don't run on adds
    filter.wantsToFilterEdit = false; // don't run on edits
})();
```
2. Prevent specific entities from being modified or deleted from your domain:

```javascript
// prevent-add-delete-or-edit-of-entities-with-name-of-zone.js by Brad
˓→Hefta-Gaub
(function() {
  function filter(properties, type) {
    var ENTITY_ID = "{the ID of the entity that you want to protect}";
    if (type === Entities.DELETE_FILTER_TYPE) {
      if (properties.id === ENTITY_ID) {
        return false;
      }
    }
    return properties;
  }
  filter.wantsToFilterDelete = true; // do run on deletes
  filter.wantsToFilterAdd = false; // don’t run on adds
  filter.wantsToFilterEdit = true; // do run on edits
  filter.wantsToFilterPhysics = false; // don’t run on physics
  filter.wantsToFilterDelete = true; // do run on deletes
  filter.wantsToFilterAdd = false; // don’t run on adds
  filter.wantsToFilterEdit = true; // do run on edits
  filter.wantsToFilterPhysics = false; // don’t run on physics
});
```

3. Allow changes only to entities’ basic physics properties:

```javascript
// allow physics, reject all other changes including adds and deletes
(function() {
  function filter() {
    return false;
  }
  filter.wantsToFilterAdd = true; // run on adds
  filter.wantsToFilterEdit = true; // run on edits
  filter.wantsToFilterPhysics = false; // don’t run on physics
  filter.wantsToFilterDelete = true; // do run on deletes
});
```

4. Reject any type of change to your domain and protect all entities:

```javascript
function filter(properties, filterType, originalProperties) {
  // doesn’t matter here if rejectAll is set to true
}
// If reject all is true. Any of the filterType changes won’t go through
filter.rejectAll = true; // default false
```

See Also

- [*Configure Your Domain Settings*](#)
- [*Backup and Restore Your Domain*](#)

### 1.5.3 Manage Your Assets

Once you’ve completed [*setting up your domain*](#), you can start building content for it. You can add content to your domain using [*our creator tools*](#) or the Asset Server.

The Asset Server hosts files or assets that can either be added as-is to your domain or that are referenced by existing entities and scripts already in your domain. Your assets are 3D models, audio files for various sound effects, image files for textures and skyboxes, scripts, and more.
You can host your content on the Asset Server, or use a cloud service such as DigitalOcean Spaces, Amazon S3, or Google Cloud Storage.

**On This Page**

- Manage Your Assets
  - Add an Asset to the Asset Server
  - Bake an Asset
  - Add an Asset to Your Domain
  - Edit Assets in the Asset Server
  - Host Assets on DigitalOcean Spaces

**Note:** Files hosted on the Asset Server are only available in your domain, and cannot be seen in other domains. Therefore, we recommend that you do not upload avatars to the Asset Server since they will not be available in domains other than your own.

**Add an Asset to the Asset Server**

You can add, edit, and remove files from your Asset Server through the Asset Browser in your domain. To add a file from your computer to the Asset Server:

1. In Interface, pull up your HUD or Tablet and go to **Create**.
2. In the **Create** app, click ‘Open This Domain’s Asset Server’ to view the Asset Browser.
3. Click ‘Choose File’ in the Asset Browser window.
4. Browse to your asset file, then click ‘Open’.
5. Enter a folder name to specify a path for your asset. This path should be added before the asset file name `/asset_filename`. By default, there is no folder created for your asset. However, we recommend that you add folders for each of your assets to help keep your assets organized.

![Specify Asset Path](image)

6. Click ‘OK’. Your assets will be added to the Asset Browser under the folder that you specified.
Once files have been uploaded to your Asset Server, they can be added directly to your domain or referenced by any entities or scripts in your domain using their URLs. To get an asset’s URL:

1. Right-click on the asset in the Asset Browser.
2. Click ‘Copy URL’.

**Bake an Asset**

Whenever you upload an asset to the Asset Server, it will be baked if it is any of the following file types:

- A 3D Model (with embedded textures)
- A Skybox

Baking is the process of optimizing assets to make them easier to transmit and render, reducing load time significantly.

**Note:** You can use the Oven tool to bake entire domains, 3D models, and skyboxes hosted externally. The Oven is a standalone application that is packaged with High Fidelity. If you’ve got any content that is hosted on external cloud services or websites, we recommend using the Oven tool. You cannot bake content using both the Oven and the Asset Server.

A baked file will have a check mark under the ‘Use Baked?’ column on the Asset Browser. A file will not bake when:

- It is a 3D model with external textures. Only models with embedded textures can be used. This error will show up as a triangular sign under the ‘Use Baked?’ column.
- It is a file type that cannot be baked. It is usually depicted by two dashes under the ‘Use Baked?’ column.

To bake a file again, just uncheck and check ‘Use baked version’.

For example, if you upload a 3D model asset to your Asset Server, you are rendering your model’s assets, which include the materials, textures, and meshes.

- Geometric meshes: A geometric mesh (more commonly known as a polygon mesh) is a collection of vertices, edges, and faces that together define the shape of a model in 3D graphics and modeling.
- Textures: Textures help in defining high frequency detail, color, or surface textures.

A user who visits your domain will download and render your 3D model assets. Rendering a 3D model in High Fidelity involves producing and analyzing images in real time. Baking optimizes the download and rendering process by compressing the poly meshes and textures in the 3D model.

**Add an Asset to Your Domain**

You can add an asset to your domain through the Asset Browser.

1. In Interface, pull up your HUD or Tablet and go to Create.
2. Click ‘Open This Domain’s Asset Server’.
3. Select the asset you wish to add from the list displayed.
4. Click ‘Add to World’.

5. In the window that pops up, you can view the asset’s URL (Asset Server URL) in the format atp:/\$FILENAME, and modify its collision and physics properties. Once you are done with these changes, click ‘Add’.

**Edit Assets in the Asset Server**

You can use the Asset Browser to edit the assets in the following ways:

- Remove assets from your Asset Server by selecting the asset files and clicking ‘Delete’.
- Rename assets by selecting the asset file and clicking ‘Rename’.
- Organize your assets into folders to help you find and reference them quickly.

**Host Assets on DigitalOcean Spaces**

You can host your assets using DigitalOcean Spaces.

1. Log in to your DigitalOcean account.
2. Once your account is open, click on ‘Spaces’ in the menu bar on the left.
3. Click ‘Create a Space’.
4. Choose your region, a unique name for your Space, and manage the public or private access to your Space. Click ‘Create a Space’.
5. You will be redirected to a page where you can upload files to your Space.
6. Click ‘Upload Files’ to start uploading your digital assets.
7. Select the files you want to upload and choose if you’d like them to be private or public. Click ‘Upload Files’.
8. Your files have now been uploaded to your Space. Hover over a file in the list to copy its URL. You can use this URL to upload assets to your High Fidelity domain.
1.5.4 Bake Your Content Using the Oven

You can reduce your load times by optimizing your domain content, models, or avatar using Oven, a standalone application created by High Fidelity. This process of optimizing content is called baking.

Why You Should Bake Your Content

Most content (avatars, entities, etc) in High Fidelity references external resources such as textures, models, scripts, and materials. When a user encounters any content in the domain, they need to download the content’s resources. Many of these resources are not optimized and can take a while to download. Baking is the process of optimizing these resources to make them easier to transmit, store, and render, reducing load time significantly.

The Oven is a tool that will help you bake any of the following types of content:
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Textures</td>
<td>When you bake a texture, the resulting folder will contain a .texmeta.json file, various .ktx images, and the original texture. Baking might introduce some compression artifacts, but these are usually minor. Baking a texture produces mipmap maps, which allow you to progressively load textures, and compresses the results. Baking large textures like skyboxes can take a while, but the benefits at runtime for everyone loading the skybox image will be significant. For the best experience, we recommend using PNG or JPEG (JPG) textures. However, we also support textures in following formats: TGA, TIF, and TIFF.</td>
</tr>
<tr>
<td>Materials</td>
<td>Baking a material will produce a .baked.json file and will also bake all of the textures in the material. Currently, we only support baking a material entity JSON file.</td>
</tr>
<tr>
<td>3D Models</td>
<td>Baking a model will produce a Draco compressed geometric mesh and will also bake all of the materials in the model. The mesh compression can slightly alter the geometry of the original model, but it is usually minor. Your output folder will contain a .baked.fst file, a .baked.fbx file (the original format of your model can be different), and a .baked.json file (materials). Because the baked materials JSON file references the baked texture files, all textures are removed from the .baked.fbx file. This JSON file is referenced in the materialMap field in the .baked.fst file. We support the following formats: FBX, OBJ, and FST (that points to a supported type).</td>
</tr>
<tr>
<td>Avatars</td>
<td>As avatars are 3D models, you can bake avatars with the same results as above. Use the resulting .baked.fst file to host and wear your baked avatar.</td>
</tr>
</tbody>
</table>

Note: You can also bake content using the Asset Server. The process for this is different than what we describe here. If you use this method, you will need to store your content on the Asset Server. Using the Asset Server will only bake items that remain in single domain (such as regular entities). It does not bake any items that can travel between domains such as avatars and avatar entities.

Get the Oven

The Oven is a standalone application that is packaged with High Fidelity and is available in your High Fidelity folder on your local or virtual machine. To use the Oven:

1. Download and install the Client + Sandbox for High Fidelity.
2. After installing, go to the High Fidelity installation folder. Run ‘oven.exe’.

The Oven has three baking options:

- Domain: You can bake all assets present in a domain.
- Models: Use this option to bake 3D Models and avatars.
- Skybox: Bake skyboxes to improve loading time for all users in a domain.
**Bake Your Domain**

When you bake an entire domain, the Oven will bake as many assets as it can, swapping out the entity properties for the baked versions of the resources.

To bake a domain:

1. Run `oven.exe`.
2. Click ‘Bake Domain’.
3. Enter the following information in the fields provided and then click ‘Bake’.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Name</td>
<td>Enter your domain’s name.</td>
</tr>
<tr>
<td>Entities File</td>
<td>This field will accept a .json or .json.gz file that contains information on all the assets in the domain. You can download these from the content archives in domain settings.</td>
</tr>
<tr>
<td>Output Directory</td>
<td>This is where the baked domain and its content will be placed.</td>
</tr>
<tr>
<td>Destination URL Path</td>
<td>This is where all the absolute resource paths will point to after baking. These can be cloud services where you store your content.</td>
</tr>
<tr>
<td>Re-bake originals</td>
<td>This checkbox lets you control whether or not to try to re-bake any baked content encountered.</td>
</tr>
</tbody>
</table>

4. In your ‘Output Directory’, upload everything in the ‘content’ folder to the ‘Destination URL Path’. This is the step where you upload all baked content to your hosting site to make it available for download.

5. Upload the resulting baked JSON file to the domain server.

**Example**

Let’s bake a domain with the following values:

- Domain Name: example_domain_name
- Entities File (.json or .json.gz):

```json
{
    "Entities": [
        {
            "type": "Model",
            "dimensions": {
                "x": 1,
                "y": 1,
                "z": 1
            }
        }
    ]
}
```

(continues on next page)
"position": {
    "x": 0,
    "y": 0,
    "z": 0
},
"modelURL": "https://hifi-content.s3.amazonaws.com/samuel/models/Block-Display-FTUE.fbx"
},
"Version": 119
}

- Output Directory: Choose a folder directory to store your baked domain files.
- Destination URL Path: “http://mywebsite.com/baked-domain/” (This is where we will host our baked content).

The resulting .baked.json file should look like this:

```
{
    "Entities": [
        {
            "dimensions": {
                "x": 1,
                "y": 1,
                "z": 1
            },
            "modelURL": "http://mywebsite.com/baked-domain/Block-Display-FTUE/baked/Block-Display-FTUE.baked.fst",
            "position": {
                "x": 0,
                "y": 0,
                "z": 0
            },
            "type": "Model"
        }
    ],
    "Version": 119
}
```

- Upload your baked content in ‘Output Directory > content’ to your hosting site.
- Upload the resulting models.json.gz file to your domain server.

**Bake Your 3D Models**

To bake a 3D model:
1. Run ‘oven.exe’.
2. Click ‘Bake Models’.
3. Enter the following information in the fields provided and then click ‘Bake’.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model File(s)</td>
<td>Enter the file or URL path for your model file(s).</td>
</tr>
<tr>
<td>Output Directory</td>
<td>This is where the baked and original content will be placed.</td>
</tr>
</tbody>
</table>

4. Upload the baked model file(s) to your hosting site.
Bake Your Skybox

To bake a skybox:

1. Run ‘oven.exe’.
2. Click ‘Bake Skyboxes’.
3. Enter the following information in the fields provided and then click ‘Bake’.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skybox File(s)</td>
<td>Enter the file or URL path for your skybox file(s).</td>
</tr>
<tr>
<td>Output Directory</td>
<td>This is where the baked and original content will be placed.</td>
</tr>
</tbody>
</table>

4. Upload the baked skybox file(s) to your hosting site.

Oven Command Line Interface

You can also use a command line interface instead of the GUI to bake single assets only (not domains). We support the following attributes:

- i: Path to file that you would like to bake.
- o: Path to folder that will be used as the output directory.
- t: Type of asset. The value can be “model” (for any model type) and “material” (for a material JSON description). For textures, the values differ based on the type of texture you want to bake, such as default, strict, albedo, normal, bump, specular, metallic, roughness, gloss, emissive, cube (same as skybox), skybox, ambient, occlusion, scattering, and lightmap.
- disable-texture-compression: Disables texture compression for any type. Use this only if the texture compression is introducing too many artifacts.

Examples

To bake a 3D model through the Oven’s command line interface:

```
>> oven.exe -i %path to model% -o %output directory% -t model
```

To bake a material through the Oven’s command line interface:

```
>> oven.exe -i %path to material json% -o %output directory% -t material
```

See Also

- Bake Your Assets Using the Asset Server
- Change Your Content Settings
- PBR Materials Guide
- Add a Material Entity

1.5.5 Host Your Domain

Depending on your requirements, you can host your domain on your local machine or on cloud servers, such as Amazon EC2. We have a partnership with DigitalOcean so that we can provide you with hosting services if you want to open your domain to a lot of visitors.
Host Your Domain with DigitalOcean

1. Log in or sign up with your High Fidelity username and password on https://highfidelity.com.
2. Hover over your username to see the various menu options. Click Cloud Domains. You can also go to https://highfidelity.com/user/cloud_domains.
3. You’ll be redirected to the Cloud Domains. Click ‘Create New’ to create a new cloud domain.
4. Before you set up a cloud domain, you need to connect your DigitalOcean account. If you don’t have one, you can sign up now.
5. You will be redirected to the DigitalOcean authorization page. Once you’ve signed in, choose the virtual machine to host your domain. Select the configuration that best suits your needs, keeping in mind that our recommendations are based on the number of users that will be visiting your domain.
6. Next, select the geographical location where you would like to host your server.
7. Click ‘Launch your cloud domain’, and then ‘Confirm’.

You will be redirected to your Cloud Domains page, where you will see the domain server you just created being configured.
The table below outlines the recommended configuration based on numbers of users.

<table>
<thead>
<tr>
<th>Users</th>
<th>DigitalOcean Droplet</th>
<th>Estimated Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-15</td>
<td>8 GB Memory, 4 vCPUs, 80 GB SSD, 5 TB Transfer</td>
<td>$40/mo</td>
</tr>
<tr>
<td>25-30</td>
<td>16 GB Memory, 6 vCPUs, 320 GB SSD, 6 TB Transfer</td>
<td>$80/mo</td>
</tr>
<tr>
<td>40-50</td>
<td>32 GB Memory, 8 vCPUs, 640 GB SSD, 7 TB Transfer</td>
<td>$160/mo</td>
</tr>
<tr>
<td>160</td>
<td>64 GB Memory, 32 vCPUs, 400 GB SSD, 9 TB Transfer</td>
<td>$640/mo</td>
</tr>
</tbody>
</table>

**Configure Your Domain Settings**

You can configure your domain settings by clicking on the three dots on the right corner and selecting ‘Domain Server settings’ from the drop down. You will be prompted to configure the basic server settings.

![Server Configuration](image)

**FAQ**

<table>
<thead>
<tr>
<th>Question</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why isn’t my domain updating? Why is it stuck on ‘updating’?</td>
<td>Check if your DigitalOcean account contains the right payment information. If the issue still persists, please contact High Fidelity support.</td>
</tr>
<tr>
<td>Is there a maximum file size for content archives for my domain?</td>
<td>Yes. The maximum file size depends on the size of your Droplet. If your file is bigger than your Droplet, you can resize your Droplet. Go to your DigitalOcean account, find your Droplet, and select resize from the menu on the left.</td>
</tr>
<tr>
<td>How can I get into my domain settings if I forgot my username and password?</td>
<td>Go to <a href="https://highfidelity.com/user/cloud_domains">https://highfidelity.com/user/cloud_domains</a>. For your domain, click on the three dots on the right corner and select ‘Reset Domain Server password’ from the drop-down. Once your server restarts, click ‘Domain Server settings’ from the same drop-down. You will be prompted to set a new username and password.</td>
</tr>
<tr>
<td>Can I host assets on my DigitalOcean domain like I can host assets on my sandbox’s Asset Browser?</td>
<td>Yes. You can use the Asset Browser to host your assets while hosting your domain on DigitalOcean.</td>
</tr>
<tr>
<td>How do you power cycle your Droplet?</td>
<td>You can do this through your DigitalOcean account. Log in and select your Droplet. Click ‘Power’ on the menu to the left and then select ‘Power Cycle’.</td>
</tr>
</tbody>
</table>

**See Also**

- Configure Your Domain Settings
1.5.6 Run Your Domain Using Docker

You can run your High Fidelity domain using Docker containers. Containers are more flexible and lightweight than a virtual machine. When you launch a container, you’re running an image of the file system at a particular instance. So you will have to update the image with every latest version of High Fidelity.

On This Page

• Run Your Domain Using Docker
  – Requirements
  – Run Your Domain in a Container

Requirements

Before you create a container for your domain, you will need to:

• Install and configure Docker.
• Install docker compose.

Run Your Domain in a Container

A Docker container is isolated from the rest of your system. This means that you will have to copy and communicate specific data to the container.

1. Create an empty folder with two files: ‘.env’ and ‘hifi.yml’.
2. The .env file contains the version of the image you want to use. To it, add the following line:

   HIFI_VERSION=0.82.0

   This is the tag and needs to be updated with the release of the latest version of High Fidelity.
3. Next, you’ll need to add information about your domain in the ‘hifi.yml’:

   ```yaml
   version: "3.6"
   services:
     domaina:
       hostname: hifi-domain-a
       image: highfidelity/hifi:${HIFI_VERSION}
       ports:
         - "40100:40100"
         - "40102:40102/udp"
         - "48000-48006:48000-48006/udp"
       volumes:
         - "hifiVol1:/root/.local/share/High Fidelity"
   volumes:
     hifiVol1: {}
   ```

   Here, you’re passing High Fidelity’s version number. You are also exposing the required ports for your domain to communicate with the network and let other users visit.
4. Go to the directory in the command line tool and run the image:
6. To stop running your container, run the following in the command line tool:

```
docker-compose stop
```

### Update Your Container to the Latest Version

You can update your container when a new version is released by changing the version number in the .env file.

See Also

- [Host Your Domain](#)
- [Set Up Your Domain](#)

#### 1.5.7 Add a Place Name

All domains have a place name, which is an easy way for visitors to enter a domain. When you enter the place name into the GoTo app, you will teleport to that domain.

By default, every domain is assigned a random place name (such as “blue-bandwidth-86201”). While you are welcome to continue using this place name, purchasing a custom place name makes it easier for visitors to come to your virtual world.

On This Page:

- [Purchase a Place Name](#)
- [Add a Custom Place Name to Your Domain](#)
- [Connect New and Existing Place Names](#)
- [Add Your Place to the GoTo Directory](#)

**Purchase a Place Name**

You will need to have a High Fidelity account to purchase a Place Name. Place Names are US $20/year, and you need to renew the purchase every year to continue using it.

1. Go to [https://highfidelity.com/user/places](https://highfidelity.com/user/places) to purchase a place name. If you’re not logged in, you will be prompted to do so.
2. Click ‘Create a Place Name’.
3. Type in your new Place Name. Place Names must be 4-64 characters in length. Letters, numbers and hyphens are allowed. You can use a Place Name that is currently not purchased.
4. Click ‘Checkout Now’.
5. Enter your credit card number, PayPal account information or a coupon code to complete the purchase.
Add a Custom Place Name to Your Domain

After you purchase a Place Name, you need to connect it to your domain. If you do not already have a domain, make sure you set one up.

1. Go to your domain settings.
2. In the ‘Metaverse/Networking’ section, click ‘Advanced Settings’.
3. Copy the Domain ID.
4. Go to https://highfidelity.com/user/places and edit the Place Name you purchased.
5. Paste the domain ID you copied in the ‘Points To’ field.
7. Click ‘Update Place’.
8. Next, configure your domain server to use the domain of your choice (in case it already isn’t configured). Right-click the High Fidelity icon on your system tray or top menu bar, and click ‘Settings’.
9. In your domain server settings, under the ‘Metaverse/Networking’ section, click ‘Advanced Settings’.
10. Under ‘Domain ID’ click ‘Choose from my domains’ and select the domain of your choice.
11. You and other users will now be able to access your domain using Interface by typing the Place Name into the Address Bar. You may also use the URL hifi://MyPlaceName.

Connect New and Existing Place Names

When you purchase a new Place Name, if you connect it with your domain ID, users will not be able to access your domain with the old Place Name. You can avoid this by connecting the newly purchased Place Name to the existing one.

1. Go to https://highfidelity.com/user/places. Edit the Place Name you purchased.
2. Add the existing Place Name under the ‘Points To’ field.
3. Enter a ‘Path’, ‘Description’, and ‘Preview Image’ (optional).
4. Click ‘Update Place’.
5. You and other users will now be able to access your domain using Interface by typing the Place Name into the Address Bar. You may also use the URL hifi://MyPlaceName.

Add Your Place to the GoTo Directory

To see your Place listed in the GoTo directory in Interface, you will need to modify some settings and permissions.

1. If you’re hosting your domain on a cloud service, open the settings here.
2. Go to Settings > Security and scroll down to the ‘Domain-Wide User Permissions’. The ‘Standard Permissions’ table allows you to set the type of users and their permissions.
3. Under ‘Connect’, select the user type you want to allow to enter your domain: anonymous, friends, localhost, and logged in. Your Place will be visible in the GoTo directory to the user types selected. For example, if you checked ‘logged-in’ under ‘Connect’, all users logged in to High Fidelity can view your Place in the GoTo Directory. But a user who hasn’t logged-in will not see your Place in the directory.

See Also

• Set an Entry Path for Users
• Configure Basic Server Settings

1.5.8 Broadcast to Other Domains

You can broadcast what’s happening in your domain to other domains in High Fidelity. This feature is useful when you want to host an event in your domain, but cannot support the number of users interested in attending it.

On This Page:

• Change Audio and Avatar Mixers’ Ports
• Add Broadcasting Server Settings

Change Audio and Avatar Mixers’ Ports

When you broadcast events from your domain, you are transmitting audio and avatar information. For the talk show you want to host, you’ll be broadcasting the audio and avatar information of yourself and the users who will be your guests. The users in the receiving domains will be able to see and hear you and your guests.

Your audio mixer conveys all the audio information in your domain. The avatar mixer contains information about how your avatar is moving in a domain. The ports for these mixers are usually randomly assigned and change every time you modify your domain’s server settings. To ensure your broadcasting settings will work, you need to assign specific ports to both mixers. Or they will change every time you save any new domain server settings.

You can assign specific ports to your mixers using the command prompt (in Windows) or the terminal (in OS X).

```
// Launch 1 domain server:
./domain-server

// Launch 1 assignment client for each type:
// List of the types:
// AudioMixerType = 0,
// AvatarMixerType = 1,
// AgentType = 2,
// AssetServerType = 3,
// MessagesMixerType = 4,
// EntityScriptServerType = 5,
// EntityServerType = 6,
// replace $TYPE with the type number of the mixer and $PORT with the port
./assignment-client -t $TYPE -p $PORT
```

Add Broadcasting Server Settings

Once you assign ports to the mixers, you can add the required broadcasting settings.

1. Right-click the High Fidelity icon on your system tray, and go to ‘Settings’.
2. In the server settings, scroll down to ‘Broadcasting’.
3. Add the server address and ports for the broadcasting server’s (your domain’s) audio and avatar mixers.
4. Add the server address and ports for the receiving server’s (your domain’s) audio and avatar mixers.

5. Add usernames to broadcast specific users (for example: when hosting a music night, you can broadcast only the band and not the audience). This acts as a whitelist: if you don’t add any users, you will broadcast nothing.
1.5.9 Stream Content

You can stream content to your High Fidelity domain, which lets you host events such as watching a football game or a movie with your friends. Or, you can go the other way, and live stream what's happening in your domain to external users.

On This Page

- Stream Audio to Your Domain
- Stream Video to Your Domain
- Live Stream from a Domain

Stream Audio to Your Domain

When you want to stream audio from your computer to your domain, you can use the Stereo Mix recording device in Windows. Any users visiting your domain will hear your computer’s audio.

1. Go to the Sound settings on your computer. Click the ‘Recording’ tab.
2. Right-click and select ‘Show Disabled Devices’. Certain new releases of Realtek drivers do not have Stereo Mix as a device. If Stereo Mix does not show up after turning on Show Disabled Devices, you can install an older version of the drivers.

3. Select and enable ‘Stereo Mix’. This will route any audio coming from your computer’s sound system into High Fidelity, as if it were a microphone.

4. In Interface, pull up your HUD or tablet and go to Audio and disable noise reduction to prevent the audio cutting out. If you are streaming from a stereo input, enable stereo input here.

5. Under ‘Choose Input Devices’, select ‘Stereo Mix’.

6. Prepare to broadcast audio. Adjust the volume on your own computer to change the volume in High Fidelity. To prevent feedback, mute High Fidelity in the Volume Mixer. Remember, you will not be able to talk while you are broadcasting audio with Stereo Mix.

7. Position the avatar within the domain. This can be tricky, so we recommend giving yourself extra time to prepare. You will now be able to hear the audio from your computer in High Fidelity.

You can change the natural attenuation and reverb of an audio source by creating Audio Zones. Audio Zones will allow you to modify the audio effects of different parts of the domain to better fit your style.

Troubleshooting

• If you cannot find Stereo Mix in your audio devices, check out this document.
• If your computer does not support Stereo Mix, check out some alternatives here.
• If your domain visitors hear an echo, ensure that your High Fidelity audio output is muted.

Stream Video to Your Domain

Screenleap is an external service for Windows that allows you to share your whole screen, an application window, or a screen section to the rest of the world. You can use Screenleap to stream video through a web entity in your domain.

We use Screenleap because it works with the current web entity system for High Fidelity. With Screenleap, everyone sees the exact same video image in sync, no matter how late they come into the domain. Latency is typically less than milliseconds.

1. Make an account on screenleap.com and login.
2. Click on Broadcast Screen and follow the instructions to download and install the Screenleap app.
3. Right click the green Screenleap icon on your system tray (Windows) or top menu bar (OS X) and choose START BROADCAST (not start screen share). You will be given the option to stream your entire screen, a window, or a box on your screen that you can define by dragging and stretching. Paste the URL into the web entity on which you wish to stream.
Troubleshooting

- If your Screenleap video rate is too low, it may be that the bandwidth is unable to keep up with the video. To fix this, you can scale down your video player to be a small window, and then make the Broadcast Screen Area box small to only accommodate that area. This will cut down reduce the necessary bandwidth and increase the frame rate. The tradeoff is lower resolution video.

- If your Screenleap video resolution is too low, increase the size of your video player window. This will require more bandwidth, and may cause your video to have lower frame rate.

Live Stream from a Domain

If you want to reach your audience through YouTube or Twitch, or your friends through Facebook Live, you can do so with the help of our live streaming guide for High Fidelity.

You can download and install OBS, our recommended streaming software, here. Open Broadcaster Software (OBS) Studio is a free and open source software for video recording and live streaming. Because of the ease of availability and valuable features, we recommend using OBS for all your streaming and recording needs.

Ensure you have the permission to live stream through the channel of your choice.

Set Up Your Domain for Streaming

Before you start recording and streaming, determine if this is a one-time event, or if it is a show or event, and the type of VR personality you’d like to have.

- Depending on your requirements, you can use your desktop, an HMD with or without controllers, or a motion capture system.

- Download High Fidelity’s Spectator Camera Marketplace app.

- Set up OBS to stream your Spectator Camera’s view.

Set Up OBS for Streaming from High Fidelity

1. Under Sources, select **Game Capture**. This is not necessary, but we recommend this setting for streaming and recording.

2. Ensure that your display is showing your Spectator Camera’s view.
3. If you’re using an HMD, uncheck disable preview in High Fidelity. In Interface, pull up your HUD or Tablet, and go to **Menu > Display > Disable Preview**. This is important, as OBS captures what is being displayed in a window and will not capture anything apart from the preview disabled window if you don’t uncheck it.

4. Set your application’s audio gain as -3dB so that its volume is lower than your audio. This allows you to be heard over any of the application’s audio.

5. To see your avatar moving properly, without any jarring experiences, you have to set your fps (frames per second) to 60. You can do this in OBS by going to **File > Settings > Video**. In the window that opens, change the Common FPS Values to 60.

6. If you have to switch between Desktop and HMD during the show, check and change your audio inputs to allow your streaming to continue without any disruptions.

7. Once you’ve set up OBS with these High Fidelity specific settings, you can add your streaming specific settings.

8. You’re ready! Click on ‘Start Streaming’.

**See Also**

- Spectator Camera
- Edit Your Audio Environment
- Official OBS Guide for Beginners
- Official OBS Guide for Advanced Users

### 1.5.10 Backup and Restore Your Domain

You can create domain backups for just the content in your domain, or your entire domain itself. These backup files can be used to restore your domain on any computer.

**On This Page:**

- Backup Your Domain Content
- Restore Your Domain Content
- Backup Your Domain
- Restore Your Domain

**Backup Your Domain Content**

You can use the content archives in your domain settings, to backup and restore your domain content. Content archives contain information about the 3D models, content settings, and assets in your Asset Server.

By default, automatic content archives settings exist and create backups in intervals. You can access these archives or generate a new one to create a backup.

1. Right-click the High Fidelity icon on the system tray or top menu bar and click ‘Settings’.

2. Go to ‘Content’ on the top menu bar and click ‘Content Archives’ from the drop-down menu.
3. You can use one of the content archives or scroll down and generate a new one by clicking ‘Generate New Archive’.

4. To download the backup folders, click the three dots under ‘Actions’ for the archive you chose.

5. Click ‘Download’. You will be prompted to save a folder with all content information on your computer.

**Restore Your Domain Content**

To restore your content backup on your computer or any other computer:

1. Right-click the High Fidelity icon on the system tray or top menu bar and click ‘Settings’.
2. Go to ‘Content’ on the top menu bar and click ‘Upload Content’ from the drop-down menu.
3. Choose your archive zip file from your computer.
4. Select ‘Upload Content’ to upload and restore your domain content.

**Backup Your Domain**

You can backup your entire domain, including your content, using the domain settings page.

1. Right-click the High Fidelity icon on the system tray or top menu bar and click ‘Settings’.
2. Go to ‘Settings’ on the top menu bar and click ‘Settings Backup/Restore’.
3. Click ‘Download Domain Settings’ to download a JSON file containing your domain’s information.

**Restore Your Domain**

You can restore your entire domain on your computer or any other computer using the domain’s JSON backup file.

1. Right-click the High Fidelity icon on the system tray or top menu bar and click ‘Settings’.
2. Go to ‘Settings’ on the top menu bar and click ‘Settings Backup/Restore’.
3. Click ‘Choose File’ under Upload a Settings Configuration to choose your domain’s JSON file.
4. Click ‘Upload Settings’ to restore your domain settings.

**1.5.11 Maintain Your Domain**

It is important that you keep your domain clean, and free of cluttered entities. You can do this with regular domain maintenance, where you clean your domain in intervals with an assignment client (AC) script. For example, you can write a script that deletes all entities that were created in a specified time (this is similar to “reverting” your domain to a backup at a specific time). Another maintenance method would be assigning the right permissions to users, so you can restrict who can edit your domain.

**On This Page**

- Set Temporary Create Permissions
- Use an AC Script to Clean Your Domain
Set Temporary Create Permissions

You can maintain your domain by giving visiting users temporary create permissions. A user can create and edit entities which will be automatically deleted after a specific interval of time (such as an hour). This keeps your domain free of clutter and maintains your domain as is.

To give a user temporary create permissions:

1. Set user permissions to ‘rez temp’. This grants users create permissions for the interval of time specified in the domain settings.
2. Set entity specific permissions (lock entities or set entity filters) to make sure that any visiting user cannot edit the existing entities in your domain.

For example, if you want to host a virtual class on adding and editing a cube’s properties, you can:

1. Set user permissions to ‘rez temp’ and set an interval of an hour. Any user attending your class can create and edit a cube entity in your domain. This cube entity will be deleted in an hour.
2. Lock all existing entities or set entity filters in your domain. This ensures that any visiting users cannot change or delete existing entities like your virtual blackboard or other classroom settings.
3. Once your class is completed, any entities created by other users will be automatically deleted. You won’t have to find these entities and delete them yourself to bring your domain back to its previous state.

Use an AC Script to Clean Your Domain

You can also maintain your domain using an assignment client script. This script can contain the details of the entities that you want to keep unedited in your domain and delete the rest. You can configure your domain settings to run this script.

We’ve written an example assignment script that contains the list of entities in a domain that you don’t want edited, and delete everything else.

```
var SEARCH_CENTER = {x: 0, y: 0, z: 0};
var SEARCH_AREA = 60000; // search area (sphere) in meters radius
var CLEANUP_INTERVAL_MIN = 1;
var MINUTES_TO_MILLISECONDS = 60000;
var ENTITIES_TO_KEEP = ["{ENTITY_ID1}", "{ENTITY_ID2}"]
var initialized = false;
var interval;
Agent.isAvatar = true;
Avatar.skeletonModelURL = "AVATAR_FST_URL";
Script.update.connect(initialization);

function cleanup() {
    EntityViewer.queryOctree();
    var foundEntities = Entities.findEntities(SEARCH_CENTER, SEARCH_AREA);
    print("Found: " + foundEntities.length + " entities");
    foundEntities.forEach(function(entityID){
        if(ENTITIES_TO_KEEP.indexOf(entityID) === -1) {
            print("Need to delete: " + entityID);
            Entities.deleteEntity(entityID);
        }
    });
}
```
function initialization(deltaTime) {
  if (!initialized) {
    if (Entities.serversExist() && Entities.canRez()) {
      Entities.setPacketsPerSecond(60000);
      EntityViewer.setPosition(SEARCH_CENTER);
      EntityViewer.setCenterRadius(SEARCH_AREA);
      Script.setInterval(function() {
        EntityViewer.queryOctree();
      }, 1000);
      initialized = true;
      interval = Script.setInterval(cleanup, CLEANUP_INTERVAL_MIN * MINUTES_TO_MILLISECONDS);
      Script.update.disconnect(initialization);
    }
  }
}

Script.scriptEnding.connect(function() {
  Script.clearInterval(interval);
});

See Also

- Set User Permissions
- Set Entity Specific Permissions
- Assignment Client Script
- Backup and Restore Your Domain

1.5.12 Best Practices For Hosting Events

If you want to host an event in High Fidelity, you will need to modify your domain settings to ensure you can host a large number of people and load all your domain content. You will also have to consider securing your domain so that your content is safe and other users have a pleasant and memorable experience.

Depending on the type of event you’re hosting, determine the following:

1. The number of users you are expecting in your domain.
2. The content you will be adding to the domain.

We’ve listed some guidelines you can follow when you set up your domain for an event.

Note: Keep in mind that domain settings will vary depending on your requirements. Here, we’ve listed the best practices, and you can tailor these to suit your needs.

1. **Host your domain**: If you’re expecting a large number of users, we recommend hosting your domain on a virtual machine. You’ll need to determine the size of the virtual machine you want to host your domain on. This varies depending on the number of users you’re expecting and the content in your domain.

2. **Set Up User Permissions**: It is good practice to set up user permissions and create entity filters so that users cannot create or edit entities in your domain. You can keep your domain clutter-free and keep users from modifying the existing domain content.
3. **Create audio attenuation zones:** Depending on the type of event you’re hosting, you will need to create audio attenuation zones. These zones control how users hear audio. For example, if you’re having an event with two hosts on stage, the hosts and audience members should hear the hosts louder than everyone else. You can set up two attenuation zones, one for the stage and the other for the audience.

4. **Whitelist scripts:** This step ensures that only whitelisted scripts can run in the domain. You are avoiding risking running any unauthorized scripts that could affect your domain and any users present.

5. **Use only server scripts:** Local scripts can slow down load times for a domain. Use server scripts instead.

6. **Bake your content:** Most content (avatars, entities, etc) in High Fidelity references external resources such as textures, models, scripts, and materials. When a user encounters any content in the domain, they need to download the content’s resources. Many of these resources are not optimized and can take a while to download. Baking is the process of optimizing these resources to make them easier to transmit, store, and render, reducing load time significantly.

**See Also**

- *Host Your Domain*
- *Set Up User Permissions*
- *Configure Your Domain Settings*
- *Manage Your Assets*
- *Bake Your Content Using the Oven*

### 1.5.13 Tutorial: Set Up A Theater Domain

This tutorial will walk you through creating and setting up a domain with a customizable theater that meets your needs. At the end of this tutorial, you’ll be able to:

- Set up a server and connect it to a domain.
- Add a theater environment.
- Configure audio server settings for a large audience.
- Manage server access and edit privileges.

**On This Page:**

- *Zaru: High Fidelity’s Customizable Theater*
- *Add the Theater to Your Domain*
- *Modify Your Server Settings*
- *Additional Server Settings*

**Zaru: High Fidelity’s Customizable Theater**

Zaru is your personal theater for any of your virtual reality event needs. While you’re free to create your own theater or environment for hosting events, we created Zaru as an easy starting point for you to host events. Zaru has a wide variety of features geared towards hosting an event:

- Seating for up to 100 avatars.
- Customizable environment tools.
- Audio attenuation settings that keep the audience quiet and focus on your performers at center stage.
Theaters to Your Domain

You can import the entire Zaru theater, as well as the ground, zoning lights, and models, to your domain. Once you have a copy of the objects in your domain, you can customize them to personalize your theater.

Import Zaru to your domain:

1. Enable both the Advanced and Developer menus.
2. Copy and paste the contents of ZaruTheater.json and save it to your computer.
3. Go to Edit > Import Entities and select the JSON file that you just saved.

When you import the JSON file, you’ll get a zone for the theater, all of the necessary models for the theater itself, lights, and a builder grid base. These are all separate, so take care if you relocate objects to align everything the way you want.

The Zaru theater file contains a specific zone called SKY/Inside, which has a set of properties for the interior of the theater. You can change settings in here to specify some permissions for users, including whether or not they are given the ability to fly.

Modify Your Server Settings

Attenuation Settings

When you host events in the theater environment, you’ll probably have performers or presenters on stage and you’ll want to update the audio in the environment accordingly. High Fidelity supports audio settings in the form of attenuation zones. These zones can be used to create different volume levels for people around your domain. For our theater, we want audio from the stage to be louder and prioritized over the audience audio, so that people in the back of the theater can hear, and so that the audience doesn’t talk over whoever is on stage.

Find the location of where your stage, audience, and microphones are in your domain. Make a note of the coordinates on the corners of each area where you want to change the attenuation settings - you’ll need to update these in the server settings of your domain.

Find your avatar’s coordinates by:

1. Opening up the console (CTRL + ALT + J).
2. Typing `JSON.stringify(MyAvatar.position)`.

3. Or by hitting the “/” button and reading the “position” in the middle column.

Audio Zones are defined by a start and end coordinate value for each of the three axes in-world. An example of how you might choose to define your zones is illustrated in the diagram below. In the sample, the X and Z values are defined along their respective axes, and the Y start and end would be the value of the floor and ceiling. To add audio zones to the Microphone stands in Zaru, create a smaller region within the audience.

We’ve made a handy tool for audio zone bounding boxes you can find here.

The way to use it is to create a cube and size and align it to where you want the boundaries of the audio zone to be. Run the script, click on the cube. The x,y,z coordinates will appear in the scripts window. Use those to define the zones in the domain settings page.

Once you have the coordinates of your audio zones:

1. Open the server settings page for your domain.

2. Click **Settings > Audio Environment**.
3. Define the audio zones that you mapped out earlier. You can keep this simple, and map only the stage and audience, or assign zones for the lobby, wings, microphones, etc.

4. Set Attenuation Zone combinations based on the locations you decided on, setting sources that you’d like to be loud to everyone to 0.

If you are having trouble with your Audio Zones check for the following:

1. For the start to end positions, place them generally in lowest to highest order.

2. Make sure the audio zones don’t cross each other’s paths, but try to make the differences between them small.

3. If you aren’t in an audio zone, the default attenuation will take hold that you can adjust as well. If you want to make sure the audience can’t be heard while the speaker is talking, pay attention to accidently crossing those areas.

4. To change how people in the same zone talk to each other, use the same zone for the speaker and the listener and adjust accordingly.

5. Make sure you think about how each zone should interact with the other zones. Stage to Audience, Stage to Mic1, Stage to Stage, Mic1 to Stage, Audience to Stage, Audience to Audience, etc…

Save your audio settings and test them until you find a combination that works for you. Other audio settings that you may want to experiment with are:

- **Noise muting threshold**: You can customize how loud someone can get before they are muted
- **Reverb settings**: For custom decay and reverberation of audio in specific zones

**Access Control**

You can set custom access controls depending on the type of events you plan on hosting in your space. If the event is limited to certain people, you’ll want to create an access list in your server settings to add specific user accounts.

The People Access List (PAL) is a handy tool for moderating your event once it has begun. You can use the PAL by selecting the People tab and viewing everyone in the domain. Admins of your domain have additional silence and ban permissions to remove ill-behaving visitors. You can also manage blocked users on your server settings page.

**Configure a Place Name**

To authenticate and make your domain easier for users to find, you can purchase a place name that connects to your domain.

**Additional Server Settings**

In addition to security and audio settings, there are several other tools you can use to plan your events. You can:

- Set avatar scaling to prevent people from making avatar sizes outside of a set range.

- Maximum user capacity and fallback location if there are a limited number of people who you want to be able to host at a time.

- Places/Paths: You can customize an exact location that you want people to spawn into your theater by specifying the location under ‘Paths’. The /path is the default, but you can customize paths (ex: /stage to spawn on stage, /audience for audience members) to move to different locations.

**See Also**

- Configure Your Domain Settings
1.5.14 Tutorial: Syncing Services

You can use web entities in High Fidelity to watch videos and presentations simultaneously with other people.

On This Page:
- Prerequisites
- Watch Synced Videos
- Watch Synced Presentations

Prerequisites

Consider getting familiar with the following concepts before starting this tutorial:
- Create New Entities
- Change How Entities Look

In this tutorial, we will use the following external software solutions in this tutorial. Note that these are recommendations, and if you prefer, you can use other tools of your choice.
- Sync Video
- Slides

Watch Synced Videos

To watch synced videos in High Fidelity, you will need to:
1. Set Up a Room in Sync Video
2. Start Streaming Using a Web Entity

Set Up a Room in Sync Video

1. Go to http://sync-video.com/ and click “Create new room”.
2. You will be redirected to a page where you can add a link for a video (YouTube or Vimeo only).
3. Copy the URL of the webpage. This is the link you will need to share with other users.
4. Press ‘Play’.

Start Streaming Using a Web Entity

1. In Interface, pull up your HUD or tablet and click Create.
2. Add a web entity by clicking the ‘WEB’ icon.
3. In the ‘Properties’ tab, enter the URL for the Sync Video (above) in the ‘Source’ field.
4. Close the Create Tools app.
5. On the web entity, click ‘Join’ to watch the video.
You will be watching the video simultaneously with the video being played in Sync Video.

**Watch Synced Presentations**

To watch synced presentations in High Fidelity, you will need to:

1. *Create a Presentation in Slides*
2. *Start Streaming Using a Web Entity*

**Create a Presentation in Slides**

1. Go to https://slides.com/, sign up and create your presentation.
2. Once you're done creating your presentation, click ‘Present’.
3. In the window that opens, copy the URL under the ‘Present Live’ section on the left.

**Start Streaming Using a Web Entity**

1. In Interface, pull up your HUD or tablet and click Create.
2. Add a web entity by clicking the ‘WEB’ icon.
3. In the ‘Properties’ tab, enter the URL for the copied Slides Present Live link in the ‘Source’ field.
4. Close the Create Tools app.
5. On the web entity, watch the presentation live.

**See Also**

- Create New Entities
- Change How Entities Look
- Sync Video
- Slides

### 1.6 Sell

High Fidelity encourages you to *create and build experiences* that you can share with others. Take the opportunity to share your creations, sell them to others, and make money in the metaverse with the Marketplace.

There is no fee to post your items for sale on the Marketplace. We help you host and certify your content, and we process your transactions and payments. In return, you retain 90% of each sale of your item, while we keep a 10% revenue share.

Throughout this chapter, learn how you can upload your items to the Marketplace and make them available for purchase to all users in High Fidelity:
1.6.1 Add Items to the Marketplace

Once you’ve created an avatar, wearable or entity, you have the option to sell it in the Marketplace. This guide walks you through the steps to upload it to the Marketplace so that you can start the process of having it reviewed and published.

Note: Prior to adding your item to the Marketplace, we recommend that you use the Marketplace Item Tester to ensure that your item works the way you expected and does not have any script errors.

Add Your Avatar

Once you create an avatar, it should contain the following files:

- An FST file
- An FBX file
- Scripts folder (optional)
- Textures folder

You will need to upload all these files to the Marketplace.

Add Your Avatar’s Files

1. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.
2. Enter your avatar’s name and select your category as ‘Avatars’. You can add the metadata now or later.
3. Click ‘Save Draft’.
4. Scroll down to the ‘Assets’ section.
5. Click ‘Choose Folder’ and navigate to where your avatar is saved on your computer. Select the folder that contains your FBX file, click ‘Upload’ twice and wait for it to upload.
6. Click on the uploaded FBX file, and copy the Marketplace path including the unique hash for your avatar. Do not copy your folder name or the name of the file.

7. Open your FST file in the text editor of your choice. (Note that the FST file is a local file on your computer)

8. Locate the relative path references to the FBX file, textures directory, and script file(s). Replace these with the absolute path references of the uploaded content.

9. Return to your avatar submission on the Marketplace.

10. In the ‘Assets’ section, click ‘Choose Files’ and navigate to where your avatar is saved on your computer. Select your updated FST file, click ‘Open’ and wait for it to upload.

11. Click the FST file as your root file in the Asset Viewer.

12. Save your submission.
At this time, we recommend that you test your avatar using the Marketplace Item Tester. If your avatar re-imports into High Fidelity without any issues, it is now ready to be uploaded to the Marketplace.

**Find Your Avatar’s URL**

Now you are ready to use your avatar! To locate the URL for your avatar:

1. Go to https://highfidelity.com/marketplace. If you’re not logged in, you will be prompted to do so.
2. Click on your user name, then on ‘My Items’.
3. Locate your avatar in Draft mode and click on it.
4. Click ‘Edit’.
5. Scroll down to the Assets section.
6. Click on the FST file. Below it, click the ‘Copy URL’ button.

This is the URL you will use when you change your avatar in High Fidelity.

If you wish to sell your avatar on the Marketplace, be sure to fill out your metadata. You can format your item’s description using Markdown syntax. Click “Submit for Review” to be verified and certified on the Marketplace.

**See Also**

- Create Your Own Avatar
- Change Your Avatar
- Marketplace Submission Rules

**Add Your 3D Model**

Once you’ve created your 3D model, you need to add the item and its assets in the Marketplace. At a minimum, your model will need the following assets:

- FBX file
- JSON file

You will need to upload these files to the Marketplace, along with any supporting files, such as textures.

**On This Page**

- Prepare Your FBX File
- Add Your 3D Model Assets
- Generate Your Item’s JSON File
- Upload Your Item’s JSON File

**Prepare Your FBX File**

1. Export your file as an FBX file from your 3D modeling program. Do not export it as a default file type such as .mb or .blend.
2. If supported by your 3D modeling software, embed your textures into your FBX file to keep the process simpler.
3. As a practice and to catch any errors, re-import your FBX back into a new scene and check that it still looks right. Look for things like missing textures or normals, and smoothing issues. You may need to tweak your export settings if the re-imported model does not look right.

**Add Your 3D Model Assets**

If your model could be re-imported without any issues, it is now ready to be uploaded to High Fidelity.

1. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.
2. Enter your model’s name and select your category. You can add the metadata now or later.
3. Click ‘Save Draft’.
4. Scroll down to the ‘Assets’ section.
5. If your 3D model contains only an FBX file, click ‘Choose Files’ and navigate to where your 3D model has been saved on your computer. Upload your FBX file. If your 3D model contains multiple files like scripts or textures, click ‘Choose Folder’ and navigate to where your 3D model has been saved on your computer. Upload all related folders, including your FBX file.
6. Click your uploaded FBX file and copy the new Marketplace URL for the FBX file.

   ![Assets](https://example.com/yourfbxfile.png)

   At this time, we recommend that you test your item using the Marketplace Item Tester. If your model re-imports into High Fidelity without any issues, it is now ready to be uploaded to the Marketplace.

**Generate Your Item’s JSON File**

1. In Interface, pull up your tablet or HUD and select Create.
2. In the Create app, click on the ‘Model’ icon to import your 3D model.
3. Enter the URL you copied in step 6. You can only do this in a domain where you have the permission to add an entity.

4. Once your model appears before your avatar, check it and make any adjustments needed. Look for missing textures or normals, smoothing issues, issues with the scale of the model, and rotation problems. Make any changes in your 3D modeling software. Once your model looks as expected in High Fidelity, you are now ready to export your model data into a JSON file. Keep in mind that different types of software render models differently.

5. Back in the Create Tools app, find your model in the Entity List. While holding the CTRL, select any other files that are associated with your item.

6. Click ‘Export Selection’ and enter a name for your JSON. This JSON file contains information on how High Fidelity can access your item and its files, and needs to contain a reference to your item’s location.

7. Open your JSON file in a text editor and check if the variable `modelURL` contains your FBX file’s Marketplace URL. If it doesn’t, you can edit it and paste the correct URL.

---

Upload Your Item’s JSON File

1. In High Fidelity, verify that your model looks correct. Open the Create Tools app, then click ‘Import Entities (.JSON)’ and navigate to your model’s JSON file.

2. Go to https://highfidelity.com/marketplace. If you’re not logged in, you will be prompted to do so.

3. Click on your user name, then on ‘My Items’.

4. Locate your 3D model in Draft mode and click on it.

5. Click ‘Edit’.

6. In the ‘Assets’ section, click ‘Choose Files’ and navigate to where your model’s JSON file is saved on your computer. Select your JSON file, click ‘Open’ and wait for it to upload. Click the JSON file as your root file in the Asset Viewer.
7. Save your submission.

If you wish to sell your 3D model on the Marketplace, be sure to fill out your metadata. You can format your item’s description using Markdown syntax. Click “Submit for Review” to be verified and certified on the Marketplace.

See Also

• Import Your 3D Model
• Introduction to Materials, Textures, and Shading
• Marketplace Submission Rules

Add Your Wearable

Once you’ve created your wearable, you need to add the item and its assets in the Marketplace. At a minimum, your wearable will need the following assets:

• FBX file
• JSON file

You will need to upload these files to the Marketplace, along with any supporting files, such as textures.

On This Page

• Prepare Your FBX File
• Add Your Assets
• Generate the JSON File for Your Wearable
• Upload Your Wearable’s JSON File

Prepare Your FBX File

1. Export your file as an FBX file from your 3D modeling program. Do not export it as a default file type such as .mb or .blend.
2. If supported by your 3D modeling software, embed your textures into your FBX file to keep the process simpler.

3. As a practice and to catch any errors, re-import your FBX back into a new scene and check that it still looks right. Look for things like missing textures or normals, and smoothing issues. You may need to tweak your export settings if the re-imported model does not look right.

Add Your Assets

If your wearable could be re-imported without any issues, it is now ready to be uploaded to High Fidelity.

1. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.

2. Enter a name for your wearable and set the category to ‘Wearable’. You can add the metadata now or later.

3. Click ‘Save Draft’.

4. Scroll down to the ‘Assets’ section.

5. If your wearable contains only an FBX file, click ‘Choose Files’ and navigate to where your model has been saved on your computer. Upload your FBX file. If your wearable contains multiple files like scripts or textures, click ‘Choose Folder’ and navigate to where your model has been saved on your computer. Upload all related folders, including your FBX file.

6. Click your uploaded FBX file and copy the new Marketplace URL for the FBX file.

Generate the JSON File for Your Wearable

1. In Interface, pull up your tablet or HUD and click on Avatar.

2. In the Avatar window, click the hat icon next to ‘Wearables’.

3. Click ‘Add custom’ at the top of the window.

4. Select the joint you’d like to use for your wearable. For example, a hat would be on your head, and fairy wings would be on your spine.

5. Click ‘Save’.
6. Using the **Create app**, make any adjustments to your wearable. For example, you can give your wearable more depth, height or angle it differently.

![Image]

7. Back in the **Create app**, find your model in the ‘Entity List’ and click on it.

8. Click ‘Export Selection’ and enter a name for your JSON. This JSON file contains information on how High Fidelity can access your item and its files, and needs to contain a reference to your item’s location.

9. Open your JSON file in a text editor and check if the variable `modelURL` contains your FBX file’s Marketplace URL. If it doesn’t, you can edit it and paste the correct URL.

**Upload Your Wearable’s JSON File**

1. Go to [https://highfidelity.com/marketplace](https://highfidelity.com/marketplace). If you're not logged in, you will be prompted to do so.

2. Click on your user name, then on ‘My Items’.

3. Locate your wearable in Draft mode and click on it.

4. Click ‘Edit’.

5. In the ‘Assets’ section, click ‘Choose Files’ and navigate to where your wearable’s JSON file is saved on your computer. Select your JSON file, click ‘Open’ and wait for it to upload. Click the JSON file as your root file in the Asset Viewer.

6. Save your submission.

If you wish to sell your wearable on the Marketplace, be sure to fill out your metadata. You can format your item’s description using Markdown syntax. Click “Submit for Review” to be verified and certified on the Marketplace.

**See Also**

- Wearables
- 3D Models
- Marketplace Submission Rules

**Upload Your Environment**

Before uploading your environment to the Marketplace, make sure that you created the environment in a domain where you have access to the administration panel. Keep in mind that all content in the domain will be included in your upload. Once your domain is set up, you are ready to upload your environment.

1. Open your ‘Domain Administration Panel’. If you are on a local sandbox, open it by clicking on the High
Fidelity icon in the taskbar notifications and ‘click Settings’.

2. Click on ‘Content’.

3. In the Content Archives section, click ‘Generate New Archive’.

4. Enter a name for your archive and click ‘Generate Archive’. It will appear just below the ‘Generate New Archive’ button.

5. Click the ‘Actions’ button, then select ‘Download’.
6. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.

7. Enter a name for your environment.

8. Under Categories, select ‘Environments’.

9. Click ‘Save Draft’.

10. Scroll down to the ‘Assets’ section.

11. Click ‘Choose Files’ and navigate to where your environment download is saved on your computer. Select the ZIP file, click ‘Open’ and wait for it to upload.

12. Save your submission.

If you wish to sell your environment on the Marketplace, be sure to fill out your metadata. You can format your item’s description using Markdown syntax. Click “Submit for Review” to be verified and certified on the Marketplace.

See Also

- Configure Your Domain Settings
- Backup Your Domain
- Marketplace Submission Rules

Add Your Audio

Once you’ve created music or ambient sounds for your domain, you need to add the sound and its assets to the Marketplace. To sell audio on the marketplace, you will need the following assets:

- WAV or MP3 file
- Script file (sound emitter script)
- FBX file (for the entity to which you will attach the sound emitter script)
- JSON file

On This Page

- Prepare Your Entity’s FBX File
- Prepare Your Script File
- Add Your Assets
- Generate the JSON File for Your Entity
- Upload Your Entity’s JSON File

Prepare Your Script File

1. In Interface, pull up your HUD or Tablet and go to Create.

2. Click on the entity you previously created to edit its properties.

3. In the ‘Properties’ tab, scroll down to ‘Script’ and your script’s path or URL.

4. Test your script file to see if your audio plays in High Fidelity.

If you don’t have a sound emitter script, you can create one by modifying this sample: soundLoopEmitter.js.
Add Your Assets

If your model could be re-imported without any issues, it is now ready to be uploaded to High Fidelity.

1. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.
2. Enter your audio’s name and select your category as ‘Audio’. You can add the metadata now or later.
3. Click ‘Save Draft’.
4. Scroll down to the ‘Assets’ section.
5. Click ‘Choose Folder’ and navigate to where your 3D model and audio file have been saved on your computer. Upload all related folders, including your FBX file and script file (in the scripts folder).
6. Click your uploaded FBX file and copy the new Marketplace URL for the FBX file.

Upload Your Item’s JSON File

1. In High Fidelity, verify that your model looks correct. Open the Create Tools app, then click ‘Import Entities (JSON)’ and navigate to your model’s JSON file.
2. Go to https://highfidelity.com/marketplace. If you’re not logged in, you will be prompted to do so.
3. Click on your user name, then on ‘My Items’.
4. Locate your Audio in Draft mode and click on it.
5. Click ‘Edit’.
6. In the ‘Assets’ section, click ‘Choose Files’ and navigate to where your model’s JSON file is saved on your computer. Select your JSON file, click ‘Open’ and wait for it to upload.
7. Click the JSON file as your root file in the Asset Viewer.
8. Save your submission.

See Also
- 3D Models
- Add Your 3D Model
- Marketplace Submission Rules

Limited Edition Items

You can choose to create a “limited edition” item by setting the quantity available. This ensures that only a specific number of your item is available for purchase on a first-come, first-serve basis.

To create a limited edition item, simply check the box “Item is a limited edition” when you fill out the form to submit your item to the Marketplace. Then, enter the number of items you want available for purchase. If you enter a number, say 25, you are saying that only 25 copies of your item will be available for purchase.

Unlimited Rezzing of Your Item

When you submit your item to the Marketplace, you can allow users to rez your item multiple times within a single domain. This allows users to create forests from a single tree (that they purchased) or to rez multiple streetlights from a single item in their inventory.
This setting is exposed in the item’s JSON file. To allow your item to be added multiple times within a domain, add "certificateType": "domainUnlimited" to your item’s root JSON file prior to submitting your item.

**Marketplace Categories**

All items submitted to the Marketplace must fall under one or more pre-defined categories. It’s important to correctly categorize your items as it helps other users find your content in the Marketplace.

**Note:** Our Marketplace team reviews each submission to ensure items are correctly labeled. Items may be declined if they are not accurately described or extraneous categories are marked.

Below, you’ll find descriptions of each Marketplace category and examples of items they include.
<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animals</td>
<td>Items depicting animals, including static 3D models of animals and functional content.</td>
<td>Fetch App, 3D Animal Models</td>
</tr>
<tr>
<td>Animations</td>
<td>Scripts and apps that modify or add animations to avatars, including inverse kinetics and flow manipulation.</td>
<td>Dance App</td>
</tr>
<tr>
<td>Apps, Scripts, Tools</td>
<td>Items composed primarily of JavaScript that add or modify functionality to High Fidelity’s content and/or Interface. These can range from a single line of code that will change an entity’s color to a full application that enables your avatar to dance. Proper apps must include an app.json root file.</td>
<td>Sit Point, TTS</td>
</tr>
<tr>
<td>Architecture</td>
<td>Structures or materials used to create structures including doors, walls, and flooring. This category does not include full environments like cities.</td>
<td>Apartment Building</td>
</tr>
<tr>
<td>Audio</td>
<td>Scripted items that produce sound effects and/or music.</td>
<td>Boombox, Looping Sound Emitter</td>
</tr>
<tr>
<td>Avatars</td>
<td>3D models that are specifically rigged to work as an avatar. Proper avatars must include a .fst avatar root file.</td>
<td>Artemis, Mountain Ogre</td>
</tr>
<tr>
<td>Education</td>
<td>Used to promote educational materials for anything through the medium of Virtual Reality.</td>
<td>Laser Pointer Flashlight</td>
</tr>
<tr>
<td>Environments</td>
<td>A collection of items meant to fill a domain, providing a complete setting or a base to build a new setting from. Environments must be entity archives, as produced by backup files from High Fidelity domains.</td>
<td>Desert Oasis</td>
</tr>
<tr>
<td>Fun &amp; Games</td>
<td>Interactive games and content, typically a scripted entity or app.</td>
<td>Beach Ball, Holiday Blaster</td>
</tr>
<tr>
<td>Furnishings</td>
<td>Items used to decorate a space, usually furniture and other props for decoration.</td>
<td>Mirror, Chairs</td>
</tr>
<tr>
<td>Light Systems</td>
<td>Items that emit light, including furnishings and other content.</td>
<td>Lamp, Torch</td>
</tr>
<tr>
<td>Materials and Shaders</td>
<td>Items users can use to customize shading or materials, including color and texture, on other entities.</td>
<td>Materials Sets</td>
</tr>
<tr>
<td>Misc</td>
<td>Items that do not fit into any of the other categories. If marked Misc, items should not be marked with any other category.</td>
<td>Particle Packs</td>
</tr>
<tr>
<td>Nature</td>
<td>Items relating to or depicting nature and the outdoors.</td>
<td>Plants &amp; Pots</td>
</tr>
<tr>
<td>Open Source</td>
<td>Free content available for the community to use and build upon. Items in this category require a source code link.</td>
<td>Digital Picture Frame, Drumset</td>
</tr>
<tr>
<td>Pets &amp; Companions</td>
<td>Items with scripted behavior or attachments that mimic a pet or companion. This category does not include static 3D models of animals commonly kept as pets.</td>
<td>Jimi the Snail, Spirit Cat</td>
</tr>
<tr>
<td>Vehicles</td>
<td>Items that are rideable and/or appear rideable through attachments or scripted functionality. This category does not include static 3D models of vehicles.</td>
<td>Skool Flyer</td>
</tr>
<tr>
<td>VR Only</td>
<td>Items that require a VR headset and/or controllers to function properly and will not work with Interface in desktop-mode.</td>
<td>Painting Set</td>
</tr>
<tr>
<td>Weapons</td>
<td>Items depicting any kind of weapon, usually attachable. Both deadly and fun-loving weapons are permitted.</td>
<td>Bow, Ping Pong Gun</td>
</tr>
<tr>
<td>Wearables</td>
<td>Items that attach to your avatar including hats, glasses, jewelry, and more.</td>
<td>Cat Tail</td>
</tr>
</tbody>
</table>
Additional Notes

Follow these guidelines when uploading content to the Marketplace:

1. Use only content which you are legally entitled to.
2. Do not use a code obfuscator, our review team needs to be able to read your scripts.
3. Audio should sound clean and be 16-bit uncompressed WAV or MP3 files.
4. Items may not have .js root files. You should either make an app to run scripts, or attach scripts to entities that will be rezzed through a .json file.
5. All entities must have a name, even if they are invisible or simple primitives.
6. Please use English in your package description, support text, and code comments.
7. If needed, you may use Markdown syntax in your description text.

1.6.2 Update Your Marketplace Item

Once you’ve added an item to sell on the Marketplace, you can update it at any time. This ability lets you offer your customers a new and improved version of your item. When an item is updated, we create a chain of related certificates between old and new versions of your item.

On This Page
- Update Your Item in the Marketplace
- Additional Notes

Update Your Item in the Marketplace

All updates to your item must be reviewed, verified and certified by our Marketplace team. Once you have the new assets, you are ready to update your item. The process is similar to when you first added your item.

1. Go to https://highfidelity.com/marketplace/items/new to create a new item. If you’re not logged in, you will be prompted to do so.
2. Enter the name of your item and select your category.
3. Update the metadata for your item.
4. Click ‘Save Draft’.
5. Scroll down to the ‘Updates’ section. Check the box ‘Updates Existing Item’ and select the old version from the drop-down.
6. Scroll down to the ‘Assets’ section and upload your assets.
7. Click ‘Submit for Review’ to be verified and certified on the Marketplace.

Once your update is approved, a new certificate will be generated. The previous version of the item will be hidden from new customers on the Marketplace and replaced with the new version.

Additional Notes

1. You cannot reference existing URLs in your updates. If you create an update, you must upload all the assets the product uses, even if they remain the same. For example, if the new version of your product uses a .wav file that has not changed in the updated version, you must upload it again in the new item. This is because the certificate system needs to recertify all files used in the product together.
2. Once the update hits the blockchain, your customers will receive notification that an updated version is available. A red dot on the Inventory icon of your Tablet or HUD indicates there is an update available on at least one Marketplace item.

3. If a customer chooses not to update to the latest version, the old version will still be valid and listed in their Inventory.

4. Once a customer has updated an item to the latest version, they will not be able to retrieve an older version.

5. Rezzed items will not be automatically updated to the latest version. This means that an object in-world will remain intact (as the previous version). To update a rezzed item, the customer needs to manually import the new version of the item.

6. If a customer purchased multiple instances of an item, each one needs to be individually updated in their Inventory. This allows users to update one item, but leave the other ones as-is.

7. All of the above policies also apply to limited edition items.

See Also
- Add Items to the Marketplace
- Ownership of Your Items

1.6.3 Ownership of Your Items

In the metaverse, it can be tricky to verify the legitimacy or ownership of a piece of property. High Fidelity implemented a blockchain tracking system that allows content creators to certify their creations and records all transactions made using HFC.

On This Page
- Blockchain Protection
- PoP License and Certificate
- Attributions

Blockchain Protection

A blockchain is a type of database, one that is maintained by thousands of different people and companies, at the same time. Information written to that database is permanent, so it can’t be changed and it can’t be lost.

High Fidelity’s blockchain allows creators to permanently attach digital certificates to their creations that securely identifies their origin and unique ownership when they are later encountered anywhere in the virtual world. The blockchain also gives control over the sale, ownership and transfer of certified goods completely over to their creators and owners—there is no way for intermediary agents (such as a company like High Fidelity or a VR server operator) to take action to change the status of something you own.
Once your Marketplace item is certified, you are permanently and indelibly recorded as its creator. Any instance of that item in the metaverse has a corresponding entry in the blockchain, and its authenticity, history, value, and ownership can be verified at any time. This makes it easy to differentiate the real deal from a counterfeit copy, and allows your property to retain its value.

**PoP License and Certificate**

High Fidelity uses a Proof of Provenance License (PoP License) and certification to protect any digital goods or assets that you have created and sold. In its simplest form, Proof of Provenance (PoP) documents an asset’s chain of ownership, its characteristics, and its entire history, from certification onward.

When you put up an item for review on the Marketplace, it has to be approved by the digital asset registry to ensure that it is functional, that it is not obviously violating copyright laws, and that it is not a copy of any other item previously approved by the registry.

Once an item is approved by the registry, you (the user) will receive a PoP Certificate of edition 0 of the item. This PoP Certificate contains static properties about the item and cannot be altered. This means all the descriptors will always remain the same and your item’s Certificate cannot be changed. It can only be changed when the PoP Certificate is transferred.

If you place this item on sale on the Marketplace, any other users purchasing your item will receive subsequent editions of your item. Every time a user purchases your item, it is certified and cryptographically signed by the Marketplace. The PoP Certificate proves that you possess an item through legal means.

**Attributions**

Attributions are often used to credit other users who have contributed to an item. They can also link to an external portfolio to display more of your work.

For some submissions, attributions are legally required by other content. Some third party content will use licenses like MIT or Creative Commons, which require special attributions. When using third party content in your own items, make sure you understand the license (if any) used by the content and respect attribution requirements accordingly.

When you add your item to the Marketplace, the form includes fields to insert any attributions for your submission.

**See Also**

- Add an Item to the Marketplace
- Marketplace Bill of Rights

### 1.6.4 Marketplace Bill of Rights

The Marketplace Bill of Rights states the privileges given to our creators and the rights High Fidelity holds over content.

High Fidelity will:

- Enforce the Marketplace Guidelines fairly, without giving preference or other advantages to High Fidelity staff or affiliates.
• Review all submissions within seven (7) business days of their submission to the Marketplace, and communicate the result of the review to the creator.
  – If 7 business days is not possible, we will contact the seller within this time frame to discuss any problems.
• Keep content in Draft mode accessible for 30 days. If an item is in Draft mode for longer than this, it will be removed. We strongly recommend keeping local backups of your content for this reason.
• Take action when we detect manipulative Marketplace practices, such as, but not limited to:
  – Flooding the market with sub-par items
  – Pricing manipulation
  – Misleading description and other presentation inconsistencies
  – Attempting to maliciously break or hack the Marketplace
• Notify Marketplace creators via email and/or the High Fidelity forums in cases of major changes of policy, extended Marketplace downtime, and delays in review time.
• Ensure that no offensive, sexual, or generally NSFW content appears on the Marketplace and guarantee that anywhere this kind of content exists has ample warnings requiring explicit permission.

See Also
• Marketplace Submission Rules

1.6.5 Marketplace Submission Rules

All your submissions to our Marketplace must adhere to these rules.

On This Page

• Marketplace Submission Rules
  – General Rules
  – Presentation
  – Documentation
  – Licensing
  – Copyright and Trademark
  – Inappropriate Content
  – Restricted Content
  – Submissions with Scripts
  – Root File
  – Wearable-Specific Requirements

General Rules

• The Marketplace Admin Team will have the final say on the publishing of all content.
• Any content that follows all technical rules may still be rejected if it has severe quality issues.
• Any content attempting to exploit loopholes or work around these rules will be rejected.

Note: Content found to infringe on copyrights and content found to be malicious after being accepted will be invalidated. This will halt all access to the item and it will no longer be available for purchase or rezzing.

Presentation

Preview Image
A preview image must be included with all items.

• Preview images must clearly show what is included in the package. Images providing context (avatars holding the item, nice backgrounds etc.) are permitted as long as it is still clear what the item is.

• If the item doesn’t have visual elements, the preview image must show context for the functionality of the item (e.g. including an image of a VR controller for a script that adds controller functionality).

Description Text
The description text must clearly state what is included in the item and describe all relevant functionality.

• Descriptions must be in English and use proper grammar, spelling, and capitalization.

• The Marketplace Admin Team may make minor modifications to the description for corrections or to provide clarity.

Titles
Item titles must be properly capitalized and only include the item name.

• Do not include subtext, taglines, or descriptive text in your title; these belong in the description text. This includes adding your username to the title as branding.

Documentation

• Any item that requires any kind of setup, special functionality, or important technical information must include documentation. This can be included in the description text and/or appear for the user when the item is rezzed in High Fidelity.

• If your item requires any special rez permissions, this must be included in the documentation

Licensing

All Marketplace items are under the Proof of Provenance License (PoP License) v1.0. Custom or alternative licenses are not permitted, unless they are compatible with the PoP license, in which case the additional licenses need to be added to the description or as a comment in the code. You can read more about the PoP license here.

Items may make use of Creative Commons, Open Source, and other free public content. However, the item cannot be primarily composed of such free content. The free content must be either substantially modified or complement content that is entirely your own.

Copyright and Trademark

Sellers must have absolute rights to the content submitted to the Marketplace. No material in violation of copyright law will be permitted.
Inappropriate Content

- **Nudify/Pornography**: No pornographic or overtly sexual content is permitted. Nudity in non-sexual context is permitted at the discretion of the Marketplace Admin Team. This can include content such as nude statues or paintings.

- **Disturbing Content**: Content made to be excessively violent or cause extreme discomfort to users is not permitted. This includes excessive violence or gore.

- **Hate Speech**: Absolutely no hate speech or imagery of any kind is permitted, including, but not limited to attacking race, religion, ethnic origin, national origin, gender, disability, or sexual orientation.

Restricted Content

We accept limited versions of content that allow the user to experience the product prior to purchasing the full version (demo, trial, lite etc.) For these submissions, the item must be free and the description text must clearly state all limitations of the item.

Submissions with Scripts

Your scripts must work as described in the Description Text of your item. All code must be self-contained, and all libraries must be uploaded to the Marketplace alongside your item.

Root File

All submissions must have a legitimate root file, based on the type of submission:

<table>
<thead>
<tr>
<th>Category</th>
<th>Root File Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avatar</td>
<td>FST</td>
<td>The main avatar file, which contains information about the skeleton, blendshapes, FBX file and textures used by an avatar.</td>
</tr>
<tr>
<td>Environment</td>
<td>TAR.GZ</td>
<td>A content archive file containing all of your domain content.</td>
</tr>
<tr>
<td>Tablet Apps</td>
<td>APP.JSON</td>
<td>A JSON file that defines the absolute paths of your JavaScript and HTML files.</td>
</tr>
<tr>
<td>Wearables</td>
<td>JSON</td>
<td>A JSON file, which contains information on how High Fidelity can access your wearable and its files.</td>
</tr>
<tr>
<td>3D Models</td>
<td>JSON</td>
<td>A JSON file, which contains information on how High Fidelity can access your model and its related files.</td>
</tr>
</tbody>
</table>

Wearable-Specific Requirements

- Wearables must have either a `parentJointName` or `parentJointIndex` property specifying the joint that the wearable will be attached to by default.

- Wearables must have `userData` defined that describes how it is to be worn.

See Also

- *Marketplace Bill of Rights*
1.6.6 Marketplace Markdown Guide

When you submit an item to the Marketplace, you can format your item description using Markdown syntax. *Markdown* is a lightweight markup language with plain text formatting syntax. Its design allows it to be converted to many output formats, including HTML.

Below, you can find the commands that we support for Markdown for Marketplace descriptions.

- *Headings*
- *Emphasis*
- *Line Breaks*
- *Blockquotes*
- *Lists*
- *Images*
- *Links*
- *Code Samples*
- *Horizontal Rules*

**Headings**

**Emphasis**

If more than one markdown syntax is listed, feel free to use any of them. The output will be the same.

**Line Breaks**

To create paragraphs, use a blank line to separate one or more lines of text. You should not indent paragraphs with spaces or tabs.

To create a line break, end a line with two or more spaces, and then hit return.

**Blockquotes**

To create a blockquote, add a > in front of a paragraph.

> This is a blockquote.

The rendered output looks like this:

Blockquotes can contain multiple paragraphs. Add a > on the blank lines between the paragraphs.

> This is a blockquote.
> It has a second paragraph.

The rendered output looks like this:

You can also nest blockquotes:

> This is a blockquote.
> The second paragraph is nested.
The rendered output looks like this:

Lists

To create an ordered list, add line items with numbers followed by periods. The numbers don’t have to be in numerical order, but the list should start with the number one.

To create an unordered list, add dashes (-), asterisks (*), or plus signs (+) in front of line items. Indent one or more items to create a nested list.

Images

Links

Code Samples

Horizontal Rules

If there is more than one markdown syntax listed, feel free to use any of them. The output will be the same.

1.6.7 Certified App Design Guidelines

Before submitting an app, make sure it follows our certified app guidelines below:

- A certified app requires 5 files: JSON, JavaScript, HTML, and two SVG
- A certified app must have a button that appears on the tablet or HUD
- A certified app must display a full screen UI in VR and a window in Desktop
- The UI for the certified app should explain how the app works
- A certified app must handle quitting gracefully

Guideline Details

1. A certified app consists of at least five files:

- A JSON file that points to the location of your app, named *.app.json
- A JavaScript file, *.js which contains the script(s) for your app
- An HTML page, *.html that defines the UI for your app
- an SVG or PNG image to display on the app button when the app is active, usually named appName-a.svg
- an SVG or PNG image to display on the app button when the app is inactive, usually named appName-i.svg

The app.json file has two required properties: “scriptURL” (whose value must be the URL of the uploaded JavaScript file), and “homeURL” (whose value must be the URL to the uploaded HTML file) using explicit paths created upon upload of the files to the Marketplace.

Here is an example of the app.json file. You will need to replace the zeros with the path to your Marketplace item:
To find the URL to your Marketplace bucket, follow these steps:

1. First, upload a draft of your app to the Marketplace.

2. Click on the app.json file on the Edit screen and the URL will be displayed at the bottom of the screen.

2. A certified app has a button that appears on the tablet in VR, or the app bar in desktop mode.

The button of the app must:

- display an icon
- display the name of the certified app.

3. The app must display a full screen UI in VR and a window in Desktop.

When the user opens the app with the button on the tablet or HUD, you must display a full screen UI in VR and a standard sized window display on the desktop. Use the AppUI module to automatically add your app’s button to the tablet or HUD and wire up handlers.

Below is an example of how to do this in your JavaScript file:

```javascript
(function () {
    var AppUi = Script.require('appUi');
    function onOpened() {
        console.log("hello world!");
    }
    var ui;
    function startup() {
        (continues on next page)
    }
}(function () {
    // BEGIN LOCAL_SCOPE
    var AppUi = Script.require('appUi');
    function onOpened() {
        console.log("hello world!");
    }
    var ui;
    function startup() {
        (continues on next page)
    }
```

1.6. Sell
If you want your app to do something specific when it is opened, you can use the `AppUi` module’s `onOpened` functionality. For example, you could:

- Query a server to get a response and determine what to show on the UI
- Start displaying a 3D interface separate from the tablet
- Determine the display mode (VR/Desktop) and change things to show on the UI

Here’s an example of using the `onOpened` functionality:

```javascript
(function () {
  // BEGIN LOCAL_SCOPE
  var AppUi = Script.require('appUi');

  function onOpened() {
    console.log("hello world!");
  }

  var ui;
  function startup() {
    ui = new AppUi({
      buttonName: "APP-NAME", // The name of your app
      home: Script.resolvePath("app.html"), // The home screen that appears when clicking the app button
      graphicsDirectory: Script.resolvePath("./"), // Where your button icons are located
      onOpened: onOpened // See the simple function above
    });
  }
  startup();
}());
// END LOCAL_SCOPE
```

4. The UI for the certified app should explain how the app works.

You should provide text on how the app works, and use familiar UI elements that a user knows how to interact with (such as buttons, scroll bars, and links).

Here are some examples of a good app UI:
5. When a user closes the app UI, you must close the app gracefully.

To close the app, you have two options:

1. Stop the active functionality of the certified app.
   - If the app had 3D interfaces on user’s hand, they go away.
   - If the app was enhancing or modified the client using a script (e.g. Spectator Camera), that functionality will end.

2. Continue running some or all of the functions of the certified app.
   - If the app wants to persist the 3D interfaces on a user’s hand, it can keep them turned on.
   - If the app wants to keep enhancing or modifying the client using a script, it can keep doing that.

If you choose to stop all functionality (option 1), then the certified app needs to provide an affordance for the user to suspend the functionality of the certified app. For example, you will need to:

1. Turn off the 3D user interfaces. This can be achieved either by providing that functionality in the 3D user interface, or surfacing a button in the UI of the certified app that achieves the same function.

2. Stop the script from modifying or enhancing the client. This can be achieved by surfacing a button in the UI of the certified app.

See Also

- Write Your Own Scripts

1.7 Contribute

High Fidelity is an open-source project, so all of our code and documentation is available for you to look at. You too can contribute to our endeavor!
Our GitHub repository contains the source to many of the components of our software for creating virtual worlds. The project embraces distributed development. If you find a small bug and have a fix, pull requests are welcome.

On This Page

- **Contribute**
  - **Build Instructions**
  - **Contributor License Agreement (CLA)**

### 1.7.1 Build Instructions

All information required to build is found in the build guide:

- **Build High Fidelity**
  - **Windows Build Guide**
  - **OS X Build Guide**
  - **Linux Build Guide**
  - **Android Build Guide**

### 1.7.2 Contributor License Agreement (CLA)

Technology companies frequently receive and use code from contributors outside the company’s development team. Outside code can be a tremendous resource, but it also carries responsibility. Best practice for accepting outside contributions consists of an Apache-type Contributor License Agreement (CLA). We have modeled the High Fidelity CLA after the CLA that Google presents to developers for contributions to their projects. This CLA does not transfer ownership of code, instead simply granting a non-exclusive right for High Fidelity to use the code you’ve contributed. In that regard, you should be sure you have permission if the work relates to or uses the resources of a company that you work for. You will be asked to sign our CLA when you create your first PR or when the CLA is updated. You can also review it here. We sincerely appreciate your contribution and efforts toward the success of the platform.

### 1.8 Frequently Asked Questions

We get a lot of questions from our users! If you have questions about our product, feel free to browse through this page to learn more. And if you don’t see your answer here, e-mail us at support@highfidelity.com!

On This Page

- **Frequently Asked Questions**
  - **High Fidelity Coin (HFC)**
  - **Sell on the Marketplace**
  - **Buy from the Marketplace**
1.8.1 High Fidelity Coin (HFC)

- How can I get HFC?
- How do I send HFC to other people?
- Why does my HFC balance not update instantly?
- Do I get charged money if a transaction fails?
- How do I convert HFC to other currencies?

How can I get HFC?

Currently, you can buy High Fidelity Coins (HFC) using Ethereum, a blockchain app that trades ETH (Ether).

To get HFC:
1. Book an appointment at the Bank of High Fidelity.
2. At the time of your appointment, pull up your Tablet or HUD and go to GoTo.
3. Go to the TradingRoom domain by entering ‘TradingRoom’ as the domain address.
4. We will provide you with a QR code that you can use to send ETH to the Bank of High Fidelity.
5. Once we receive the ETH amount you sent, we will send you the appropriate amount of HFC to your account based on the current exchange rate. This can take up to three business days.

You can also receive HFC from gifts from your friends or by selling items on the Marketplace.

How do I send HFC to other people?

1. Open the Inventory app.
2. Click ‘Send Money’.
   - To send HFC to one of your connections, click ‘Connection’, and choose the recipient from the list.
   - To send HFC to someone nearby in the domain, click ‘Someone Nearby’ then click on the person in the domain. Confirm that you selected the correct person in the dialog.
3. Add the amount you wish to send, then click ‘Submit’.

Why does my HFC balance not update instantly?

HFC transactions sometimes take a few seconds to update, as they are backed by a blockchain. If your balance has not updated within 5 minutes, contact our team at support@highfidelity.com and they can help you.

Do I get charged money if a transaction fails?

No! Your HFC balance only changes when a purchase or transaction is confirmed on the blockchain.
How do I convert HFC to other currencies?

Our goal is to always have an easy way for creators to retrieve the money they’ve earned. Currently, you can cash out your HFC to US Dollars (USD) using PayPal.

To cash out your HFC:
1. Book an appointment at the Bank of High Fidelity.
2. At the time of your appointment, pull up your Tablet or HUD and go to GoTo.
3. Go to the TradingRoom domain by entering ‘TradingRoom’ as the domain address.

The minimum amount of HFC that you cash out is $25 (or 2,500 HFC), and the maximum cashout value is $2000 (or 200,000 HFC) per calendar month. High Fidelity may, at its discretion, issue the occasional exception to the maximum cashout amount.

1.8.2 Sell on the Marketplace

- What should I charge for my Marketplace Item?
- Does my avatar have too many polys?
- How big can my submission be?
- How do I upload to the Marketplace?
- Can I host content on the Marketplace for personal use?
- What happens after I submit my item to the Marketplace?
- How long does it take to find out about the status of my submission?
- What are the rules for submitting content to the Marketplace?

What should I charge for my Marketplace Item?

You are free to price your item however you’d like. We encourage you to consider these questions:
- How are other items of similar quality in this category priced?
- Are you choosing a price that you feel reflects the time and effort required to create the item? Don’t sell yourself short!
- If you have multiple items for sale on the Marketplace, in essence you have a brand. Do you have a specific target audience in mind? New users? Users looking for something truly unique? Take this into account.
- Is your item a limited edition? If an item is of good quality and in limited supply, it may merit a higher price.

Does my avatar have too many polys?

We do not have a hard poly limit on avatars, however we recommend keeping avatars under 80k. Consider normal mapping your models to preserve high detail rather than excessive poly counts.

How big can my submission be?

Submissions can be as large as necessary for the content you are uploading, but keep in mind that large items can adversely impact loading times for users. If possible, we recommend keeping submissions under 40MB total.

How do I upload to the Marketplace?

Log in to the Marketplace home page and select ‘New Submission’ from the top-right menu.
Can I host content on the Marketplace for personal use?

You can submit content without putting it up for sale. However, you will still need to go through the certification process and be held to Marketplace standards. Content not submitted for review will expire after 30 days and will no longer be accessible.

You should always keep backups of your files, as we cannot ensure the return of your files if they are corrupted or lost.

What happens after I submit my item to the Marketplace?

Your submission will be evaluated by our team. Once our team has completed evaluation, you will receive an email telling you that your item was accepted or declined with reasons and resubmission guidelines.

How long does it take to find out about the status of my submission?

The Marketplace team will respond by email within 7 business days.

What are the rules for submitting content to the Marketplace?

Please refer to our Marketplace Submission Rules.

1.8.3 Buy from the Marketplace

- The Marketplace content I purchased has disappeared. Where did it go?
- Can I rez multiple copies of a Marketplace item?
- What is Dynamic Domain Verification (DDV)?
- What happens to my Marketplace content when I change my domain ID?
- What happens to my Marketplace content when I export a backup to another domain?
- How do I report a DMCA violation?

The Marketplace content I purchased has disappeared. Where did it go?

Marketplace items behave like items in the real world. You can have only one copy of each item you purchase. An item you bought from the Marketplace will disappear if you rez it elsewhere. This is done using a process called Dynamic Domain Verification (DDV).

Can I rez multiple copies of a Marketplace item?

Yes, only if the seller has given permission to do so. A seller can modify settings to allow users to rez multiple copies of their item in a single domain. Otherwise, you will need to buy multiple copies of the item. Marketplace items behave like items in the real world. You can have only one copy of each item you purchase unless specified by the seller.
What is Dynamic Domain Verification (DDV)?

Dynamic Domain Verification (DDV) is a process running on your domain, which ensures that you can rez only one copy of each item purchased on the Marketplace. This is done to protect the intellectual property rights of the creators. DDV will determine if an item has been moved to a new domain and will remove it from the old domain. DDV requires the domain to be running and will delete items from domains with no Place or domain name. Temporary names are accepted. DDV often runs within an hour after you rez a Marketplace item again. Its frequency is controlled by the domain’s control panel.

What happens to my Marketplace content when I change my domain ID?

When you change your domain ID, it’s the same as creating a new domain. Even though the existing content may temporarily appear in the new domain, the system considers the Marketplace items as existing on the old domain. DDV will remove the Marketplace items from the new domain. You will have to rez the items again in the new domain.

What happens to my Marketplace content when I export a backup to another domain?

When you restore a backup or exported domain onto a new server with a different domain ID, DDV will consider the Marketplace item as belonging to the old domain, and will delete them from the new domain. You will have to rez the Marketplace items again in the new domain.

How do I report a DMCA violation?

Email us at support@highfidelity.com to report content violation, and we will take appropriate action.

1.9 Release Notes

See what’s been released in High Fidelity.

For historical release notes (prior to Beta Release 80), view our forum.

1.9.1 Beta Release 83

We released Beta Release 83 on Wednesday July 31, 2019.

Note: There is a protocol change in this version.

New in v0.83.0!

• Many efforts have been made to improve stability and performance in this release:
  – Over 75 crashes have been identified and fixed.
  – Desktop mode now works on business-class PCs, supporting a wide range of Nvidia, AMD, and Intel graphics cards for Windows PCs.
  – Improved Mac stability and performance, including support for MacBook Air:
    • Throttling the refresh rate to require less power in a desktop.
The default screen size is now set to 1/2 scale to decrease the amount of power and improve startup time.

The default resolution of High Fidelity is now set at 1/2 the screen resolution to reduce pixel processing bottleneck for the low-end Mac computers. Due to retina screen resolution and combination of integrated or low-end graphics cards, a lot of Macs suffer from processing too many pixels due to the high screen resolution. To change this setting, go to Settings > Graphics.

A lot of attention has been made to improving disconnecting and short disconnects/reconnects. This would manifest in pink Orbs never resolving properly and white orbs sticking around; we rarely see any of those these days!

- **Auto-Optimization for Rendering and Graphics**: In this release, we add controls for world detail, rendering effects, and refresh rate to optimize your experience. By default, you will be loaded into the recommended settings based on your hardware configuration. You can further refine these settings by going to Settings > Graphics.

- **Welcome back the webcam and microphone access!**: In a previous release, we completely removed any access to webcams and mics due to security concerns. We’ve since addressed and fixed those security issues. In addition, if a webcam or microphone need access at any time, you will see an access control dialogue where you can either grant or deny this. (Canny request: [https://highfidelity.canny.io/bugs/p/embeded-chromium-browser-are-not-enumerating-camera-and-microphone-inputs](https://highfidelity.canny.io/bugs/p/embeded-chromium-browser-are-not-enumerating-camera-and-microphone-inputs)).

- **Comfort Mode**: Comfort Mode is designed to decrease the effects of motion sickness in VR mode by:
  - Decreasing your field of vision by darkening the edges of the screen.
  - Adding a ground plane and grid.
  - This feature is OFF by default and can be activated and adjusted in Settings > Controls > Calibration.

- **Avatar Theft Protection**: Users are no longer able to wear certified avatars that they do not own. If someone attempts to wear a certified avatar that they don’t own, they instead will wear a generic avatar and given instructions on how to change to another avatar.

- **Baking of Avatars**: Avatars uploaded through our Avatar Packager will now be automatically sent through our Oven process and baked. This provides for better optimization and performance.

- **Simultaneous Editing of Objects**: In Create mode, you can now simultaneously edit multiple entities at once. Select the entities you want to change, then modify the properties. For certain properties (such as color and behavioral properties), changing the value will change it for all selected entities. Other properties (such as spatial properties) will offset the original values by the changed value.

- **New User Experience**: We’ve returned to our previous start-up behavior so that new users will start their HiFi journey in a local domain by themselves. The updated content introduces the movement controls and teaches users how to use High Fidelity.

- **Update to QT 5.12.3**: This update fixes bugs and errors related to the outdated QT APIs that were in use.

- **New Documentation includes**:
  - Upgrade High Fidelity
  - Comfort Mode
  - FAQ for High Fidelity Coin (HFC)

**Improvements**
Avatars

- Avatar bandwidth is decreased by 13.9%. We achieved this by removing faux joints that are not required unless the user is actively grabbing an object.
- We’re continuing to improve our avatars and make them more lifelike!
  - Added more talks, idles and fidgets, and polish system to hide loops and repetition of animations, and also to reduce glitches as the component animations blend in and out. See this PR for technical details.
  - Reworked the stop and settle animation to be smaller and less bouncy, especially for small adjusts.
  - Reworked all locomotions to match movement speeds that were changed by design to be 2.5 m/s for walk and 5.2 m/s for run. Polish the animations to work for authored height (Artemis) and also the short wolf avatars. See this PR for more details.
  - We added the ability to override animations for the right and left hand. This enables users to create scripts for custom hand poses. See this PR for more information on hooking in your scripts.
- Improved our safe landing code to properly find local entities when teleporting or entering a domain. This prevents avatars from showing up in undesirable locations when moving around. Many thanks to Oren Hurvitz for the code!
- Added support for morph Targets in GLTF Avatars so that animations can be used, specifically Walk and Jump!
- Set different default run speeds between HMD and Desktop mode for the Analog control scheme (Settings > Controls > Control Scheme Selection).

Create Tools

- The Independent and Entities views have been removed from the View menu in Create mode.
- Introduced shadowBias and shadowDistance as zone properties to allow precision when creating realistic shadows in different lighting conditions. NOTE: We tried to match the default values as closely as possible to the previous hard-coded values, but many existing domains that were already using shadows prior to 83 may now need a little additional adjustment to get these new properties set correctly.
- Added ignorePickIntersection to Create UI.

Rendering

- Improved CPU usage by optimizing avatars and the _Blender tasks associated with rendering. This dropped CPU 10% on Windows. See this PR for more technical information.
- Optimized the way we handle switching between different GL contexts in frames. This will remove the stuttering that could occur when loading content.
- Improve gLTF importer correctness and support for the following features: mesh deformer, unnamed materials and vertex colors, sparse accessors, thin bounding boxes. Many thanks to SaracenOne for your contribution!
- Added Haze to our forward renderers.
- Reduced the complexity of managing/editing our common shaders, and made it easier to add more variants in the future by combining shaders variants in SLP defines.
- Baking FST files with the oven now includes the existing materialMap in a baked FST file.
API Documentation

- Documentation is now updated and complete for the following APIs:
  - AvatarBookmarks
  - AvatarInputs
  - Clipboard
  - Controller
  - Desktop
  - Entities
  - location
  - LocationBookmarks
  - Midi
  - Overlays
  - Paths
  - PlatformInfo
  - Picks
  - PickType
  - Pointers
  - Quat
  - RayPick
  - ResourceRequestObserver
  - Reticle
  - Scene
  - Script
  - Selection
  - Settings
  - Snapshot
  - Tablet
  - Users
  - Uuid
  - Vec3
  - Window

Other Mentions

- Now allow users to disable automatic flying (Settings > Controls > Hover When Unsupported). Many thanks to SaracenOne for the contribution.
– When enabled, if you walk off of a tall building (or off the edge of the content so there is only the void below you), you will start to hover.
– When disabled, when you walk off, you will start to fall. If flying is enabled, you will still be able to start flying while you are falling.

• We now call leaveEntity before unloading a script if our avatar is inside the entity.
• Unmuting while away will now remove your away state.
• Cleaned up some warnings in syntax-error.js that were appearing in Visual Studio. Thanks to Oren Hurvitz!
• Improved loading by fixing up some race conditions that could trigger waiting for login to complete, even when it already did, edit.js would fail to load as scripts started running before the Keyboard had been initialized. Thanks Oren Hurvitz!
• Made improvements for Linux users by correctly minimizing (iconizing) Interface. Before this fix, the Interface window would immediately pop back up.
• Added min-listen-port command line parameter to assignment monitor. This will allow users to specify a port range for assignment client UDP ports, allowing easier port forwarding, etc.
• Added vhacd-util and Oven to all installers. VHACDUtil is a content optimization tool like our Oven and is an alternative for content creators.
• Updated our installer icons.

Fixes

Audio

• The Dynamic Jitter Buffer switch in the Domain Admin UI was inverted, causing audio quality degradation. This has disabled dynamic buffers when it’s set to enable them. This now works correctly.
• Adjusted the adaptive audio threshold value we use when Muted. This fix improved the "MUTED" indicator (which warns when talking while muted) to stop constantly flickering.
• We now persist the mute settings between sessions.
• When unmuting with keyboard shortcut (Ctrl + M) in PTT mode, the icons now correctly indicate the current mute state.
• Plugging and unplugging USB headsets no longer cause Interface to hang or freeze. (Canny request: https://highfidelity.canny.io/bugs/p/unplugging-headphones-with-hifi-open-causes-freeze)
• Fixed an issue in audio where clicks/pops could be heard when gain is rapidly changing.

Avatars

• Fixed an issue where the getJointRotation and setJointRotation were not the same values when you use different overloads of the function. Specifically, the MyAvatar.getJointRotation(index,quat) was working but MyAvatar.getJointRotation("jointname", quat) was not.
• In HMD mode, avatars will no longer sink into the floor (this behavior previously manifested in certain hand/shoulder poses).
• In cases where avatars have no eye joints, eyes no longer pop out of joint and the camera view no longer appears inside the avatar’s head. (Canny request: https://highfidelity.canny.io/bugs/p/head-flashes-into-view-while-in-hmd)
• User scripts now correctly override the rotations of the eyes using the MyAvatar.setJointRotation() API.
• Users who change from a GLTF avatar to an FBX avatar now correctly move, rather than floating in the air.
• Limited the amount of data that can be read from a single binary array in an FBX node. This was potentially causing crashes due to corrupt FBX files.
• Changes from AvatarBookmarks.updateAvatarEntities() now appear correctly on remote interfaces.
• Equipping a wearable on a missing joint no longer duplicates the wearable. (Canny request: https://highfidelity.canny.io/bugs/p/clicking-cancel-in-wearable-app-causes-problems)
• Fixed an issue where Head-mounted avatar entities were visible in first-person after releasing from grab when trying on. This would only manifest in HMD mode.

Create Tools

• In the Create app, Copy/Paste now works on grandchildren of entities. Before this fix, only an entity’s direct children were copied, but not their children. Thanks to Oren Hurvitz!
• Fixed a long-standing issue where Undo and Redo were not following the command chain correctly when editing Zones. As a result, some Undo/Redo steps would get skipped.

Rendering

• Transparency of UI elements and graphics now display correctly on Mac computers with AMD chips.
• Avatars no longer freeze in the middle of their transition state, resulting in a partially rezzed avatar. This could occur during log out, on a disconnect, when logging in, or anywhere else where the avatar was in a state of transition.
• Switching between HMD and desktop modes now works seamlessly. Users will no longer be stuck in desktop mode permanently. (Canny request: https://highfidelity.canny.io/bugs/p/fix-moving-between-desktop-and-hmd-on-oculus)
• Fixed the issue where we were artificially clipping the FOV in Oculus devices, resulting in a black border appearing around your field of view. (Canny request: https://highfidelity.canny.io/bugs/p/black-border-appears-at-all-times-in-vr-at-least-on-oculus)

Other Mentions

• The keyboard now correctly scales to your avatar when you reload content or scripts while your avatar is larger or smaller than a default mannequin.
• Special characters in your username no longer causes blank inventory and HFC balance.
• Added size checks for trait packets parsing to ensure we don’t read past the end of packet buffers. This can help prevent certain crash cases when sending/receiving Avatar Skeleton Traits.
• When using the Oven, the materialURL “materialData” is no longer parsed as a URL.
• New users are no longer asked for a wallet passphrase on login.
• Fixed potential error in network code, involving congestion control.
• Fixed an instance where you would upload content to your domain, it would appear to fail but actually succeed. For example, when restoring a content backup, the error message “There was a problem loading your list of automatic and manual content archives, please reload the page to try again” would appear but the content is actually there.
Crash Fixes

- Fixed crashes in the following scenarios:
  - Accessing Avatar::getJointIndex or Avatar::getJointNames due to memory corruption in _modelJointIndicesCache
  - Accessing flow collision arrays from different threads when touching flow bones or hair on other avatars
  - Backing to invalid indices in baker::calculateNormals
  - Shutting down while other avatars are rezzing into the domain
  - Incorrect material mapping from FST files

Android

- Fixed an instance where the Android app would crash on launch
- Corrected the gamma levels that were set too high and making some content appear washed out.

1.9.2 Beta Release 82

We released Beta Release 82 on Tuesday April 30, 2019.
Note: There is a protocol change in this version.

Hotfix v0.82.1

A hotfix (v0.82.1) was released on April 30, 2019.
Note: There is a protocol change in this version. All Clients and Servers need to update to this latest release.

- A feature to slow the rate that nodes are added to the avatar mixer was introduced with RC82.0. This code path was not working correctly and a server-side fix has been made 82.1. Any domains running should be immediately updated to 82.1.
- A change to Entities was causing a deadlock in the Entity Server. The fix required a protocol change. All clients will need to update to 82.1.

New in v0.82.0!

- Improved Locomotion: With new controls in both VR and Desktop mode, movement is easier and more intuitive! Customize your settings by going to Settings > Controls. (Canny request: http://roadmap.highfidelity.com/feature-requests/p/improved-locomotion-numerous-features)
  - Choose from three modes for more granular control over how you move:
    - Default: You will always walk at the same speed. To run, fully push the stick forward.
    - Analog: Your movement speed is determined by far you push the stick forward on your controller. To run, fully push the stick forward.
    - Analog++: This is the same as Analog, except that you can define your maximum walk speed with a slider in the UI.
  - Choose what direction you move in while in VR mode: either in the direction your head is facing, or in the direction your hand is pointing.
– In Desktop, hold Shift while using the arrow keys to walk sideways (strafe).

– In VR mode, you can now disable Strafe (walking sideways).

**New Audio Settings:** In this version, we introduced independent volume controls for avatars, ambient background volume, and overall system sound. The audio level meter is smaller and takes up less space in the window. Go to Settings > Audio to change your settings. (Canny Request: [http://roadmap.highfidelity.com/feature-requests/p/implement-2-main-volume-sliders](http://roadmap.highfidelity.com/feature-requests/p/implement-2-main-volume-sliders))

**Additional Controls for Entities:**

– All Entities: Render Layer and Primitive Mode

– Web Entities: New controls for Web Color, Web Alpha, Script URL and Max FPS

– 3D Models: Group Culled

– Particles: Emit Shape Type

**Features Added to Avatar “Hero” Zones:** The Hero zone was introduced in v0.79.1, where avatars within the zone receive more bandwidth over other avatars in the domain. Additional features have been added:

– Hero zones are updated dynamically as you change the zone properties. Creating, deleting, moving and resizing the hero zones will immediately affect the avatar’s hero status.

– Domain owners can now specify the amount of bandwidth that is dedicated to Hero zones set up in their domain. Go to [http://localhost:40100/settings > Avatar Mixer > Advanced Settings > Hero Bandwidth](http://localhost:40100/settings > Avatar Mixer > Advanced Settings > Hero Bandwidth) to set the bandwidth.

– Hero zone properties are now accessible via scripting and our JavaScript API.

**New Documentation includes:**

– Prioritize Zones within Your Domain

– Create Avatar Entities

– Create a Custom Avatar Animation JSON file

– Bake Your Content Using the Oven

– Run Your Domain Using Docker

– Best Practices for Hosting Events

– Tutorial: Create a Smoke Fountain (learn how to use Particle entities)

**Improvements**

**Avatars**

– The new Flow app (released in v0.81.0) has been updated to work with the “sim” naming scheme.

**Create Tools**

– New support for the OpenEXR high dynamic range image file format on skyboxes and ambient maps.

– A damping value is now added by default to dynamic 3D models.
Oven and Baking

- Oven output format has changed from FBX to FST, but remains backward compatible to support our existing baked FBX files.
- Oven now supports baking materials, so that all textures with the material are now automatically baked.
- Oven now supports baking avatars directly.
- Oven now supports Dropbox hosting and correctly reports invalid FBX files.
- View updated documentation at https://docs.highfidelity.com/host/oven.html.

API Documentation

- Documentation is now updated and complete for the following APIs:
  - AnimationCache
  - Audio
  - AvatarList
  - AvatarManager
  - Camera
  - HMD
  - Menu
  - Messages
  - ModelCache
  - SoundCache
  - TextureCache

Other Mentions

- The privacy bubble has been renamed “shield”. You can turn it on/off using the Tablet/HUD or the new indicator next to the audio level mixer.

Fixes

Avatars and People App

- The High Fidelity Avatar Exporter for Unity includes fixes and improvements to the warnings and errors. Download the updated exporter here (v0.4.1): https://github.com/highfidelity/hifi/raw/57c3620587ae1ec4638a58988909b46602c14633/tools/unity-avatar-exporter/avatarExporter.unitypackage
- Users will no longer see duplicate avatars in a domain after someone else disconnects and reconnects.
- Users can now correctly see a friend’s location when their availability is set to ‘Friends Only’.
- Fixed the People app filter bar. It no longer breaks when you delete connections.
Create Tools

- Grab handles correctly stop pulling object after the trigger or mouse is released.
- Using the DEL key in HMD no longer erases active User Data objects.

Marketplace and Inventory

- Pressing ENTER while searching the Marketplace now initiates the search, rather than removing the search text.
- Pressing the TAB key while searching the Marketplace no longer repeats the current search.
- Updates to a Marketplace item now correctly displays the cost (if applicable), rather than “free”.
- Users can no longer select the URL of a Marketplace item by selecting and dragging the mouse on a Marketplace item in the drop-down list.
- Fixed some text formatting issues in the Marketplace App buttons.
- Fixed an edge case where non-creators could view and purchase items that were not for sale.
- The LOG IN and CANCEL buttons on the Inventory page now correctly respond if the user logs out while Inventory app is open.
- We now check the a user’s HFC balance prior to sending money to another user.
- We added failure handling when sending money via a script fails.

Rendering and Core Engine

- Notifications now render correctly in VR mode, even when an avatar is scaled very small.
- The seek lasers on hands are no longer offset when an avatar is resized.
- Near-grabbed entities now update correctly for the observer.
- Web entities display the correct transparency.
- Fixed an edge case in which the view perspective of avatars disagreed on whether a held entity is kinematic. With this fix, objects look clean to an observer and pass through the Kinematic object smoothly.

Domains

- The ‘Create Temporary Placename’ button now correctly appears when you change the domain ID of a domain.
- Modifications to entities no longer disappear or unexpectedly change when the domain starts up.
- Backups of domain archives no longer fail when content sets are null.
- Fixed an edge obscure case where rezzing an item more than once would cause the following issues:
  - Subsequent items were not added to the simulation.
  - Parents/children don’t get fixed up.
  - The signals addingEntity and addingEntityPointer don’t get emitted.
Other Mentions

- Users can again create logins through Oculus.
- You can now test your voice even when muted or Push to Talk is enabled.
- Restarting scripts will not correctly trigger the ‘Tablet screen changed’ login.
- Fixed an intermittent bug that occurred when equipping a cloneable entity in HMD. The new entity would get created but didn’t actually get equipped. This is now fixed.

Crash Fixes

- Fixed crashes in the following scenarios:
  - Closing interactive windows in the Create app
  - Using the Appreciate app

Android

Nothing specific for Android in this release.

1.9.3 Beta Release 81

We released Beta Release 81 on Friday, March 29, 2019.
Note: There is a protocol change in this version.

Hotfix v0.81.1

A hotfix (v0.81.1) was released on March 31, 2019. This release disables the ability to create a web view in Interface that enables access to a user’s webcam or microphone audio.
Note: There is a protocol change in this version.

New in v0.81.0!

- **Flow Technology:** We have migrated High Fidelity’s flow technology from JavaScript to C++. This improves performance across the board and makes it more accessible to non-programmers. To use it, download the new Flow app from the Marketplace, adjust the settings to configure physics such as gravity and elasticity, and copy the resulting code directly into your avatar’s FST file.

- **Microphone and Mute Experience:**
  - **Push-to-Talk:** Push-to-Talk will keep you muted except when you are pressing buttons on either your keyboard or your controllers. Go to Settings > Audio > Push to Talk to enable. To talk in Desktop mode, press and hold the “T” button. To talk in VR, press and hold the grip triggers on both of your controllers.
  - **Mute Notification:** We now provide visual feedback when you try to speak in a muted state. You will see the word “MUTED” in red on your screen if you try to talk while muted.
  - **Test Microphone Settings:** You can now test your microphone settings from within Interface. Go to Settings > Audio > Test Your Voice. You will see your voice input on the Audio Level Meter.
  - Your mute state is now correctly saved when reopening Interface.
• **Track Hand Controllers in Oculus Home**: When leaving High Fidelity to the Oculus Home, we now continue to track your hand position rather than putting you in an “Away” state. Go to Settings > Controls > Calibration to turn this on or off. Thank you to our user Saracen! (Canny request: [http://roadmap.highfidelity.com/feature-requests/p/track-hands-in-oculus-dash](http://roadmap.highfidelity.com/feature-requests/p/track-hands-in-oculus-dash))

**Improvements**

**Avatars**

• The High Fidelity Avatar Exporter for Unity has been updated:
  – You can adjust the avatar’s scale prior to exporting it from Unity.
  – Better error detection now notifies you of scale and material errors you may encounter while using the tool.
  – Download the updated exporter here (v0.4.0): [https://github.com/highfidelity/hifi/raw/77ea47a9dbbb49c626e3ccae79fbc34645fffd1/tools/unity-avatar-exporter/avatarExporter.unitypackage](https://github.com/highfidelity/hifi/raw/77ea47a9dbbb49c626e3ccae79fbc34645fffd1/tools/unity-avatar-exporter/avatarExporter.unitypackage)
• In the Avatar Packager, click on any error messages you encounter to view troubleshooting tips on how to resolve the error prior to uploading your avatar.
• Improved UI to clarify that uploading your avatar to High Fidelity’s servers via the Marketplace does not automatically put it up for sale. You can always put your Avatar for sale by going through our Marketplace process.

**Security**

• Your personal bubble will automatically be enabled when “Away” to prevent avatar harassment.
• When someone is banned from a domain, we now send a message to the owner or admin of the domain to confirm the ban, assuming they have kick rights. This ensures that users cannot be banned using unsolicited scripts within a domain.

**Other Mentions**

• Far grab now uses two hands to reduce accidental activation.
• Choose whether wearables can be grabbed while being worn using the Lock icon on the Avatar app. (Canny request: [https://highfidelity.canny.io/bugs/p/wearables-can-get-moved-by-grabbing](https://highfidelity.canny.io/bugs/p/wearables-can-get-moved-by-grabbing))
• We’ve removed grab scaling. We’ll work on ways to improve this in a future release and turn it back on. (Canny request: [http://roadmap.highfidelity.com/feature-requests/p/ability-to-prevent-grab-scaling](http://roadmap.highfidelity.com/feature-requests/p/ability-to-prevent-grab-scaling))
• New toggle determines whether or not wearables can be adjusted by hand (Canny request: [http://roadmap.highfidelity.com/bugs/p/wearables-are-detaching-from-avatars](http://roadmap.highfidelity.com/bugs/p/wearables-are-detaching-from-avatars))
• Marketplace items now have the ability to be rezz’d multiple times within a single domain. For example, someone can buy a single tree, then rez it multiple times to create a forest. When submitting an item to the Marketplace, content creators control whether or not an item can be rezz’d multiple times in the JSON file (Read documentation here). Coming in future releases: We’ll make it easier to mark which items can be rezz’d multiple times, along with visual indications of these items in your inventory. (Canny request: [http://roadmap.highfidelity.com/feature-requests/p/allow-marketplace-items-to-specify-the-number-of-copies-or-infinite-a-buyer-can-](http://roadmap.highfidelity.com/feature-requests/p/allow-marketplace-items-to-specify-the-number-of-copies-or-infinite-a-buyer-can-))
Fixes

Avatars

• Fixed a bug in which wearables became detached from your avatar.
• Fixed a bug where avatars occasionally shifted into flying pose while on the ground.

Other

• The Escape key now consistently places the user in “Away” mode. (Canny request: http://roadmap.highfidelity.com/bugs/p/escape-key-doesnt-set-avatar-to-away-mode)
• Ambient Occlusion works again! We inadvertently broke this a couple of releases ago.
• In a web entity, you can now input text in a web text box or text field. This fixes the ability to use the 3D keyboard to type in web entities. (Canny request: http://roadmap.highfidelity.com/bugs/p/web-entity-keyboard)
• Hand lasers no longer disappear when selecting another avatar’s sphere with the People app open. We inadvertently introduced this bug in v0.79.0.
• When multiple entities are selected, using the arrows or dragging numerical fields will now display the correct value, rather than the minimum value.

Android

Nothing specific for Android in this release.

1.9.4 Beta Release 80

We released Beta Release 80 on Tuesday, March 19, 2019.

Note: There is a protocol change in this version.

New in v0.80.0!

• **Added Clock to Tablet**: Local time is displayed in the upper right of the tablet in HMD, above the username (or log in, if not logged in).
• **Downloadable Assets**: Content creators can now download their assets for their Marketplace items, including both approved and invalidated items. This is supported on the web-based Marketplace by opening it in a browser; it is not yet supported using the Market app.
• **Digital Ocean Password Reset**: Domain Admin users can now reset their admin username and password on Digital Ocean Domain Servers. Go to highfidelity.com/user/cloud_domains, sign in to High Fidelity, and select the Reset Domain Server password option from the menu.
• **Avatar Doctor**: The Avatar Packager now contains error detection, aka the “Avatar Doctor”. This tool will notify you of common errors before you upload your avatar files to our server for hosting. **Coming in future releases**: click on the error to view troubleshooting tips on how to resolve the error prior to uploading your avatar.
• **Creators Stocking Inventory**: Allows creators to stock their inventory with their own NFS certified items and gift them to others.

• **New and Updated Documentation**:
  
  – The API Reference has been updated to a new, easy-to-read style that helps developers find what they’re looking for more easily, complete with a comprehensive search. It is available at https://apidocs.highfidelity.com.

  – New documentation will include (clear cookies if you’re getting 404 errors):
    
    * Firewall Settings
    * Using High Fidelity with Android (Canny request: http://roadmap.highfidelity.com/feature-requests/p/requested-documentation-more-info-for-high-fidelity-on-android)
    * Improved Documentation for Entity Filters and ‘Protecting Your Domain Content’ (Canny request: http://roadmap.highfidelity.com/feature-requests/p/improve-filter-documentation)
    * Add Your Place to the GoTo Menu
    * Tutorial: Build a Boom Box
    * Tutorial: Create an Avatar Scaling Button
    * Tutorial: Open Web Page with Entities
    * Tutorial: Build a Painting Set
    * Tutorial: Create a Purchase Button for Marketplace Sales

This release completes the overlay work we began a couple of releases back. We have converted our internal use of overlays to local entities and are moving toward deprecation of 3D overlays.

Content Creators: From now on, **please use local entities instead of overlays**. Any work you have that uses Overlays should be migrated to use local entities.

**Improvements**

**Avatars**

• The High Fidelity Avatar Exporter for Unity has been updated to notify you of texture and file errors you may encounter while using the tool. Download the updated exporter here: https://github.com/highfidelity/hifi/tree/master/tools/unity-avatar-exporter

**Create Tools**

• The ‘Entity Types’ dropdown of the Entity List has new buttons for Select All (check all entity types) and Clear All (uncheck all entity types).

**Rendering**

• We added the ability to load GLB (GLTF binary) format assets with on our ongoing work to support glTF.

• The Market tablet app is rewritten in QML (from HTML) to support future mobile-compatible devices.
Other Mentions

- When you update a Marketplace item, the number of likes now carries over from the original item to the updated item in the Marketplace.
- During a custom install, you now have the ability to install only the Sandbox or only the Interface **without uninstalling** the other.
- The list of live-speakers in the People tablet app are sorted live. You no longer need to refresh or close the PAL to re-sort the list except in very noisy or low-level places.
- We’ve added new text to warn when sending an inventory to the Trash is an irreversible action.

Fixes

Avatars

- UTF-8 characters are read correctly in FST files. This fixes a bug where Japanese characters in FST files were incorrectly read, causing the avatars to fail while loading.
- We continue to address the “white orb” issue where avatars appear as white orbs when loading into a domain and never fully load as avatars.

Create Tools

- Fixed the ability to change animations of an FBX model entity by a script or via the Create menu (Canny Request: http://roadmap.highfidelity.com/bugs/p/fix-entity-model/animations).
- Sorting entity and files lists now correctly sorts alpha and non-alphanumeric characters.
- Drop-downs now close when a user clicks outside of the list.
- Users can now change a particle effect’s URL by typing into the field directly.

Rendering

- Tablet no longer disappears due to LOD issues (Canny Request: http://roadmap.highfidelity.com/bugs/p/make-overlays/local-entities-not-effected-by-current-lod)
- Entity properties are respected during far/near grab while in HMD (Canny Request: http://roadmap.highfidelity.com/bugs/p/entity-properties-not-set-in-vr)
- Fixed an issue where avatars drop through the floor when loading, because they were able to move too soon. We now don’t allow avatar movement until the floor renders and physics is enabled.
- Fixed an issue where you could fly in zones that had “Flying Allowed” set to **off** when creating a Zone in Create tool. (Canny Request: http://roadmap.highfidelity.com/bugs/p/avatar-dont-fall-and-can-start-to-fly-when-the-zone-is-set-to-no-fly).
- Continuing to improve and work on the issue of soft wearables **flickering** when worn.
- Keylights now update correctly when a zone is rotated.
- Procedural entities now render both sides of their triangles correctly. This fixes a bug that was introduced in a recent release.
- As part of overlay work, we’ve deprecated Overlays.keyboardFocusOverlay in favor of Entities.keyboarFocusOverlay which can be used to set keyboard focus via scripting.
Other

- Fixed a long-standing issue that would sometimes occur where an ESS script couldn’t get the properties of its parent. There has been a workaround in place where a blank javascript function was placed inside of the parent entities serverScripts property. After this fix, you should no longer need to use this workaround.

- Fixed an issue in the PAL where index numbers would append after display names for users even if they were the only user with that name in the domain. This was occurring due to an issue with refreshing content, re-entering a domain, or recovering from crashes.

Crash Fixes

Fixed crashes in the following scenarios:

- Resizing entities
- Parsing AvatarEntity data with too many LocalJointTranslations
- On Mac Desktop computers

Android

Nothing specific for Android in this release.