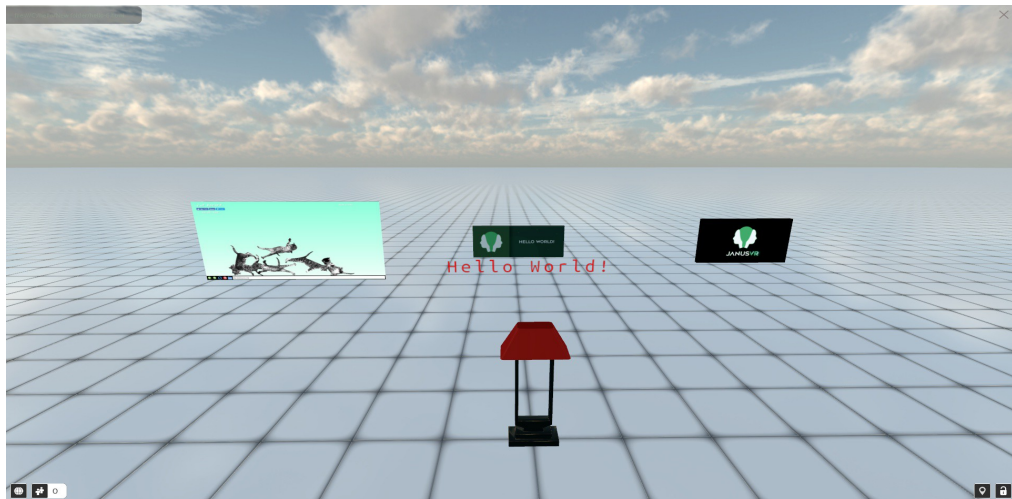




## Introduction to JanusVR Lesson 6

# HOW TO ADD A 3D MODEL TO A ROOM



June 2017

This instructional guide is designed to teach some of the fundamental skills needed to create virtual reality spaces using JanusVR.

This sixth lesson will continue on with the room made in previous lessons, and explains how to add a 3D model.

© JanusVR 2017

This publication is protected by copyright. JanusVR encourages the free transfer, copying and printing of this publication if such activities support the purpose and intent for which this publication was developed.

For further information regarding this publication, contact:

Allan Parker

[training@janusvr.com](mailto:training@janusvr.com)

Visit our website at [www.janusvr.com](http://www.janusvr.com)

## 1. Defining the Asset name and location

In this example we have added a 'Object' as the 'Asset', set the 'id' attribute for our 'Object' as '3d\_model', and the 'src' as the URL attribute for a 3d model of a lamp.

```
<html>
<head>
<title>Hello World!</title>
</head>
<body>
<FireBoxRoom>
<Assets>
<AssetImage id="hello_world" src="http://i.imgur.com/yLxgtkE.png" />
<AssetWebSurface id="cats" src="http://cat-bounce.com/" width="1280"
height="720" />
<AssetVideo id="vid"
src="http://www.janusvr.com.s3.amazonaws.com/janus_vid.mp4" />
<AssetObject id="lamp"
src="http://janusvr.com.s3.amazonaws.com/lamp.dae" />
</Assets>
<Room use_local_asset="room_plane" >
<Text pos="0 2 4" xdir="-1 0 0" ydir="0 1 0" zdir="0 0 -1" col="red" scale="2 2
2" >Hello World!</Text>
<Image id="hello_world" pos="0 1.5 8" xdir="-1 0 0" ydir="0 1 0" zdir="0 0 -1"
/>
<Object id="plane" pos="7 1.5 0" xdir="0 0 1" ydir="0 1 0" zdir="-1 0 0"
scale="4 2 1" collision_id="plane" lighting="false" websurface_id="cats" />
</Room>
</FireBoxRoom>
</body>
</html>
```

## 2. Adding the Object to the Room

The 'id' of the object is 'lamp', and it is 4 metres forward, 1.5 metres above the ground.

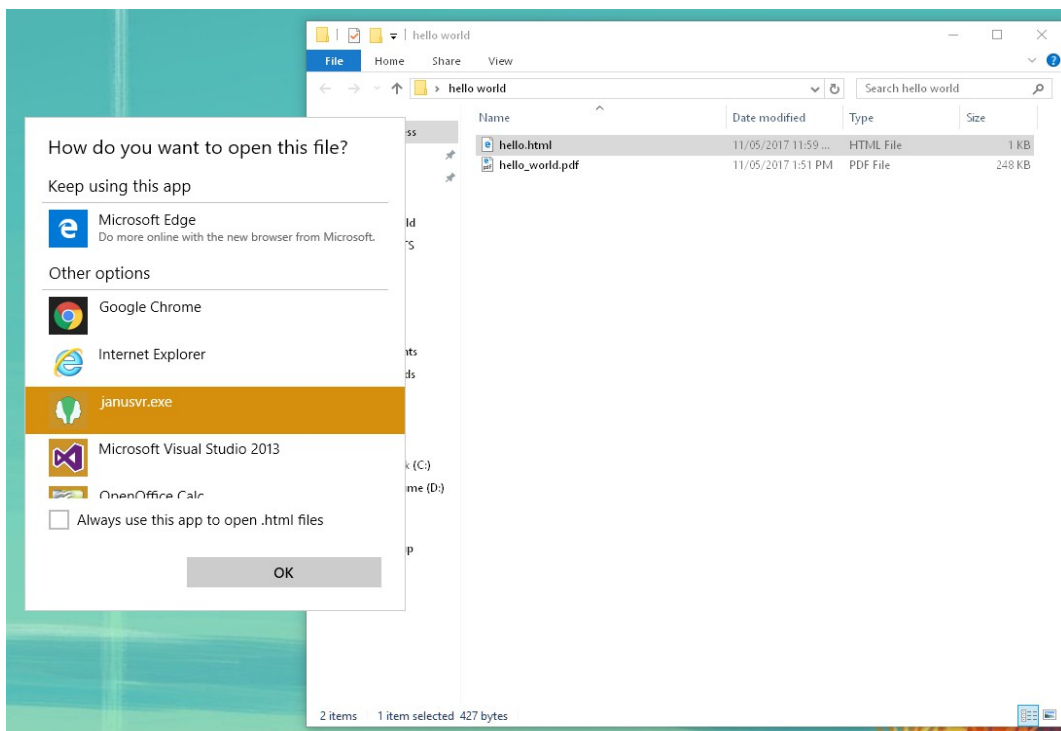
```
<html>
<head>
<title>Hello World!</title>
</head>
<body>
<FireBoxRoom>
<Assets>
<AssetImage id="hello_world" src="http://i.imgur.com/yLxgtkE.png" />
<AssetWebSurface id="cats" src="http://cat-bounce.com/" width="1280"
height="720" />
<AssetVideo id="vid"
src="http://www.janusvr.com.s3.amazonaws.com/janus_vid.mp4" />
<AssetObject id="lamp"
src="http://janusvr.com.s3.amazonaws.com/lamp.dae" />
</Assets>
<Room use_local_asset="room_plane" >
<Text pos="0 2 4" xdir="-1 0 0" ydir="0 1 0" zdir="0 0 -1" col="red" scale="2 2
2" >Hello World!</Text>
<Image id="hello_world" pos="0 1.5 8" xdir="-1 0 0" ydir="0 1 0" zdir="0 0
-1" />
<Object id="plane" pos="5 1.5 8" xdir="-1 0 0" ydir="0 1 0" zdir="0 0 -1"
scale="4 2 1" collision_id="plane" lighting="false" websurface_id="cats" />
<Video id="vid" pos="-5 1.5 8" xdir="-1 0 0" ydir="0 1 0" zdir="0 0 -1"
scale="1 1 1" lighting="false" />
<Object id="lamp" pos="0 0 3" />
</Room>
</FireBoxRoom>
</body>
</html>
```

### 3. Viewing your Room

Save the file on your local drive as a .html.

There are 2 ways to see the room you have just made:

- i. Right click on the file and select 'open with' (Windows), select 'Choose another app' – scroll down and find JanusVR.
- ii. Start JanusVR and press 'Tab' – type or copy/paste the local file path for the .html file (c:\helloworld\hello.html for example). To paste a link in JanusVR desktop mode press 'Tab', click on input box and press Control-V



You now know how to display an object, the next section has some exercises to help you get more familiar with displaying an object.

At the end of this document is a list of all the options for loading objects - have fun!

## Exercises:

Change the `src=""` to see different 3D objects displayed. This example is a .dae model of a car that has been compressed using .gzip, which JanusVR automatically decodes.

**Swap:**

```
<AssetObject id="lamp" src="http://janusvr.com.s3.amazonaws.com/lamp.dae" />
```

**To:**

```
<AssetObject id="lamp" src="http://www.thevrbar.com/build/Belair.dae.gz" />
```

**Make the object rotate by adding the `rotate_deg_per_sec=""` attribute**

**To:**

```
<Object id="lamp" pos="0 0 3" />
```

**Add:**

```
<Object id="lamp" pos="0 0 3" rotate_deg_per_sec="180" />
```

**Add collision to the lamp**

**To:**

```
<Object id="lamp" pos="0 0 3" />
```

**Add:**

```
<Object id="lamp" collision_id="lamp" pos="0 0 3" />
```

**Add physics to the lamp**

**To:**

```
<Object id="lamp" pos="0 0 3" />
```

**Add:**

```
<Object id="lamp" collision_id="cube" pos="0 3 3" collision_static="false" />
```

## Defining Object as an Asset:

These are the 3D geometric objects which can be used within the FireBoxRoom. Supported formats are OBJ, DAE, 3DS and FBX.

Here is the complete list of attributes:

- id - id of the AssetObject
- src - location of the file containing the geometry
- tex0 (default "") - location of a texture image (not using a material file)
- tex1 (default "") - location of a texture image (not using a material file)
- tex2 (default "") - location of a texture image (not using a material file)
- tex3 (default "") - location of a texture image (not using a material file)
- mtl (default "") - location of the material file (not using texture images)
- tex\_clamp (default "false") - whether to perform texture clamping (GL\_CLAMP\_TO\_EDGE), or allow textures to repeat (GL\_REPEAT)
- tex\_linear (default "true") - if true, textures have bilinear filtering applied (GL\_LINEAR). If false, a nearest sampling method (GL\_NEAREST) is used which gives textures a pixellated look
- tex\_compress (default "false") - if true, uses hardware-supported texture compression on textures (GL\_COMPRESSED\_RGBA).
- tex\_mipmap (default "true") - if true, uses mipmapping for textures.

The URL to the file is specified by the src attribute. You can also specify materials for the file using either a single texture file (specified with the tex attribute), or more traditionally by specifying the location of the material file (specified with the mtl attribute).

Here is an example of the first method (specifying a single image as a texture):

```
<AssetObject id="pinetree" src="pinetree.obj" tex="pinetree.png" />
```

Here is how to specify a material file (which may reference many textures):

```
<AssetObject id="pinetree" src="pinetree.obj" mtl="pinetree.mtl" />
```

<http://www.janusvr.com/guide/markuplanguage/index.html#assetobject>

## Displaying a Object in a Room:

An Object refers to an instance of 3D geometry placed in the room. Objects can be used to define both the geometry of the room, as well as the boundary for the room, by using the `collision_id` attribute, detailed below.

- **id** - set to the id of an AssetObject
- **pos** (default "0 0 0") - specify the position (anchor point is centered horizontally, and at the bottom vertically)
- **fwd** (default "0 0 1") - specify the orientation (or use `xdir`, `ydir`, `zdir`, defaults "1 0 0", "0 1 0", "0 0 1")
- **col** (default "#ffffff") - Allows the user to change the colour of the associated element. The formatting is as follows:
  - `#rgb/#rrggb/#rrrgggbbb/#rrrrggggbbbbb` where any valid hex color is accepted
  - `r g b` Where each value is from 0 to 1.
  - `r g b a` Where each value is from 0 to 1.
  - `rgb(r,g,b)` where `r g` and `b` are from 0 – 255
  - `rgba(r,g,b,a)` where `r g` and `b` are from 0 - 255, `a` is from 0 to 1.
  - `hsl(h,s,l)` where `h` is from 0 to 360, `s` & `l` are from 0 to 100.
  - `hsla(h,s,l,a)` where `h` is from 0 to 360, `s` & `l` are from 0 to 100, `a` is from 0 to 1.
  - Also, any legal web color name is accepted as input.
- **scale** (default "1 1 1") - scale the object along each of its x (horizontal), y (vertical) and z (forward) axes
- **locked** (default "false") - if "true", prevents modification of attributes
- **cull\_face** (default "back") - options are "back", "front", "none" which specify what polygons are culled when the Object is rendered
- **collision\_id** (default "") - when set to the id of an AssetObject, collision testing is performed with that AssetObject. This makes it possible to define the boundary for the room using one's own custom geometry. (Note that the `id` and `collision_id` attributes can be set differently - the `collision_id` may refer to an AssetObject which is a low-polygon count version of a more detailed model, such as a bounding cube or sphere. Note also that collision tests are not performed if the player is not within the bounding volume of the AssetObject.)

- **collision\_trigger** (default "false") - When set to true, registers the object into the physics system so that it can be tested for collisions, but allows other objects to pass through it as if it were not a solid object.
- **collision\_radius** (default "0") - when set to a value greater than zero, an invisible cylinder at the specified radius prevents the player from passing through this Object (note that presently, this ignores the vertical or Y-position of the player, the cylinder extends in both directions along the Y-axis infinitely)
- **collision\_pos** (default "0 0 0") - Defines the collision mesh's x y and z offset from the parent object
- **collision\_scale** (default "1 1 1") - Scales the collision mesh along each of its x (horizontal), y (vertical) and z (forward) axes
- **rotate\_axis** (default "0 1 0") - defines an axis of rotation
- **rotate\_deg\_per\_sec** (default "0") - specifies the number of degrees to rotate per second about the axis defined by rotate\_axis (note use of this feature is discouraged, as presently it breaks other interactions with FireBox - use at your own risk)
- **video\_id** (default "") - set to the id of an AssetVideo to shade the Object using frames of the video as a texture (see the section on AssetVideos for more information on defining an AssetVideo). Also note that the Object if clicked will serve as a control to start/stop the AssetVideo.
- **image\_id** (default "") - set to the id of an AssetImage to shade the Object using the image as a texture (see the section on AssetImages for more information). Note that the AssetImage will work even if SBS/UO formatted, or has animation.
- **shader\_id** (default "") - set to the id of an AssetShader to shade the Object with a GLSL fragment shader (see the section on AssetShaders for more information on defining an AssetShader)
- **websurface\_id** (default "") - set to the id of an AssetWebSurface to texture the Object with a 2D web view (see the section on AssetWebSurfaces for more information on defining an AssetWebSurface)
- **thumb\_id** (default "") - set to the id of an AssetImage, to show an image/thumbnail when an attached websurface is not selected (the AssetImage can animate)
- **lighting** (default "true") - if "true", uses the default shading which includes diffuse and specular components

- **anim\_id** (default "") - set to the id of an AssetObject containing an animation
- **anim\_speed** (default "1.0") - changes the rate of playback of the animation defined by anim\_id
- **loop** (default "false") - if set to true, an animation will repeat from the start once completed
- **visible** (default "true") - if set to false, the geometry for the Object is not visible/rendered
- **teleport\_id** (default "") - if set to an AssetObject's id, will constrain teleportation in the HTC Vive to the defined mesh.
- **collision\_static** (boolean, default "true") - Whether this Object is motionless/immovable (e.g. set to "true" for a wall or obstacle like the floor, set to "false" for a projectile).
- **draw\_layer** (default 0) - Allows the user to manually sort the depth order of an object. For instance, an object of depth 0 will draw above an object of depth 1. Priority may be negative. Each priority group is also sorted based on distance.

<http://www.janusvr.com/guide/markuplanguage/index.html#object>