Creating a 3D World in X3D

Hello, this is an ongoing guide! X3D is a great tool for making 3D spaces on static websites, particularly if you want to avoid the pain of programming in Javascript.

X3D is an XML language like HTML, I'm going to assume you are familiar with HTML, how a HTML page is laid out and words like tags, attributes and links. If you don't understand these words, please get familiar with HTML before following this.

X3D has a website, its quite technical but has some good info if you can find it. The best place to look is: https://doc.x3dom.org/index.html they have a tutorial that should have answers to things I don't cover here, this document will just explain my workflow and the basics to get started. There is a great tutorial here: https://doc.x3dom.org/tutorials/index.html

Tools

The tools we will use are: a HTML editor like VSCode, Brackets or Nova, and Blender, the 3D editing program.

You will also need a local web server to test your X3D worlds, DO NOT try and run them directly from files. Many HTML editors have built in live servers or plugins that provide a server (I recommend this one for VSCode). You can also install a program like Caddy for windows or MAMP for mac. Or use NodeJS.

Finally if you want to host your world on the web, you'll need web hosting. I recommend Neocities. ALTHOUGH you will need supporter rank (5$) to upload your x3d files.
Alternatively Netlify can host it for FREE but the web address will be ugly: https://app.netlify.com/drop (You can get a free nicer address at afraid.org) Or you can use an iframe to display it in a neocities site.

SAMPLE FILES: You can follow along with the sample project files here: https://github.com/ombideshdar/X3D-Demo

Getting Started

Your world will take the form of 2 key files. The HTML wrapper file that renders the world, and a primary X3D file which you can consider as the world. You can also link multiple X3D files together to reuse world parts such as models.

Your HTML File

This is a regular HTML file, it needs to reference the x3d style and javascript. Otherwise you can modify it in any way you wish.

You can download the latest x3d files here. You will get a zip with many files, the only two you need are x3dom.js and x3dom.css. The other files can offer additional features, but that's outside the scope of this write up. (For now)

<script src="x3dom.js"></script>
<link rel="stylesheet" href="x3dom.css">

In your body you will need to define the default x3d file to render
<x3d PrimitiveQuality='Medium' disableDoubleClick="true"><scene><inline url='demo.x3d'/></scene></x3d>

The "**PrimitiveQuality**" can be low, medium or high, it defines the resolution of primitive objects. The **disableDoubleClick** can be on or off, it changes the double click effect, play with it!

Fun fact, you can include everything from your x3d file in your html file instead of doing the **<inline>** tag! But that gets messy! So we will separate the two, but you don't have too!

Your primary X3D File

Ah ok this is the bit you want to know!
Your primary X3D file defines the world, in it you create the objects that make up the world, define animations, set the player controls and set camera details and sky colour and fog. In the sample files this is called "demo.x3d".

Let's explain it top to bottom. ALSO you can get more info about all tags on this page: https://doc.x3dom.org/author/nodes.html

The **<Scene>** tag is like the <body> tag in HTML, it defines the world, everything within it is what gets displayed.

Colours and the background. A background tag defines the colour of the sky. Colours in X3D are in the Generic Decimal format. Most colour editors have an option to select that format.

<Background skyColor='0.000, 0.662, 0.949' />

<Viewport>

The **<Viewpoint>** tag is the player's camera, it is your eyes in the world. It has a number of options.
"**Position**" is the starting location of the player.
"**Orientation**" is the rotation of the players default view.
"**centerOfRotation**" is the pivot point of the player, it deines where the head is as such.
"**fieldOfView**" is the FOV, if you have played any game you understand this.
Fun fact! Having a hard time figuring out what you want the default viewport position to be? You can get locations easily. Do "Ctrl/Command-D" when viewing an x3d world to open the debug, then click a location to get its position.

<navigationInfo>

The navigation info tag is where you define player controls. This is where you set how the player interacts with the world.
"**Type**" this define the kind of navigation, "**LOOKAT**" is a default floating camera. "**WALK**" is a more first person game type view. There are some others too look at this page.
"**Headlight**" gives you a light on the players head!
"**Speed**" and "size" are as they sound, you can play with them.

Note in the sample files the world has been animated to spin. You may want to remove that to see things better!

<Transform>

This is like a DIV in HTML. Its an empty modifiable object.
"**DEF**" this is the name of the transform, its id. Use this to reference it later (mostly used for animations)
"**Translation**" this is the position of the transform. Here you can set the location of the object.
To scale the object add a "scale" attribute like this:
<Transform translation="0 0 0" scale="2 2 2">
You can rotate it too by adding a "rotate"
<Transform translation="0 0 0" scale="2 2 2" rotation="0 1 0 3.1">
Rotations are in Quaternion format!

<Shape>

Shape tags are definers for primitive shapes. Within them you can define a shape and give it a texture. I could explain all this but the example is pretty clear. You can see a list of possible shapes here.

That's all the basics for now!
Activity! See if you can use the documentation to figure out how to add <fog> to your world!

Animations

Ok so animations in X3D are weird and a hassle. SORRY! I'm sure there is a way to define them within blender which might be easier. However I will explain the basic concept here.

Animations require 3 parts and 4 tags.
<TimeSensor>, this is like a clock. It defines a sequence of time. You can modify the “cycleInterval” to make the clock run faster or slower.
<OrientationInterpolator>, this lists the keyframes of the animation, in the example project the keyframes are for a rotation. Honestly they are confusing and a mess and I just stole them from stack overflow!
<ROUT>, Routs are basically connectors that connect different parts of the animation together. You need 2 routs for an animation, the first rout connects the TimeSensor to the OrientationInterpolator. The second rout connects the OrientationInterpolator to the object you want to animate. Its a little confusing.

The best I can suggest is messing around with what’s there and see what you can figure out!

Extra tip, here is an example path based animation.
<PositionInterpolator DEF='npc_walk' key='0 .25 .5 .75 1' keyValue='10 -1 11, 10 -1 -7, -10 -1 -7, -10 -1 11, 10 -1 11'/>
Note it is a PositionInterpolator instead of an OrientationInterpolator. You also need to make the ROUT a little different **<ROUTE toField='rotation'/>** and **<ROUTE toField=translation/>**

<Inline> X3D and exporting with Blender

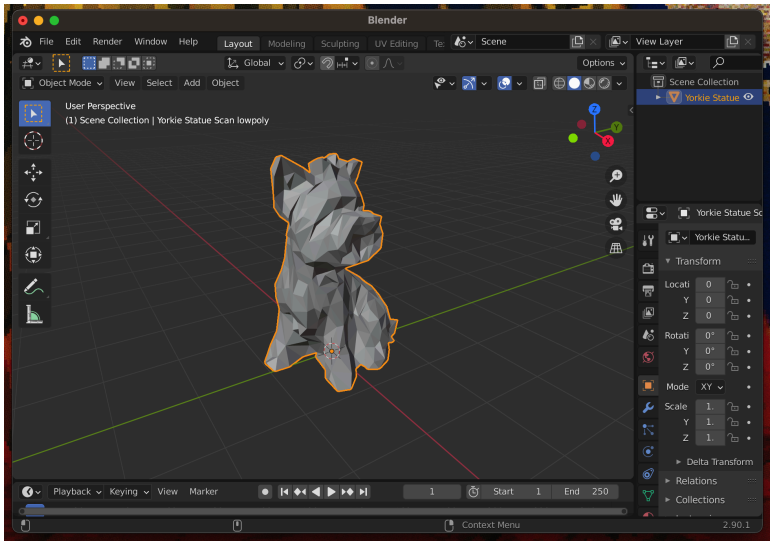
OK SO, defining your whole world in one file would get very long and confusing to work with, so it’s easier to connect in multiple files using the inline tag. If you have ever used an <img> or a <link> tag in HTML it’s very similar.
Example: **<inline url='assets/hill.x3d' />**

So now that you know how to put an x3d file inline, lets try and make one with blender!

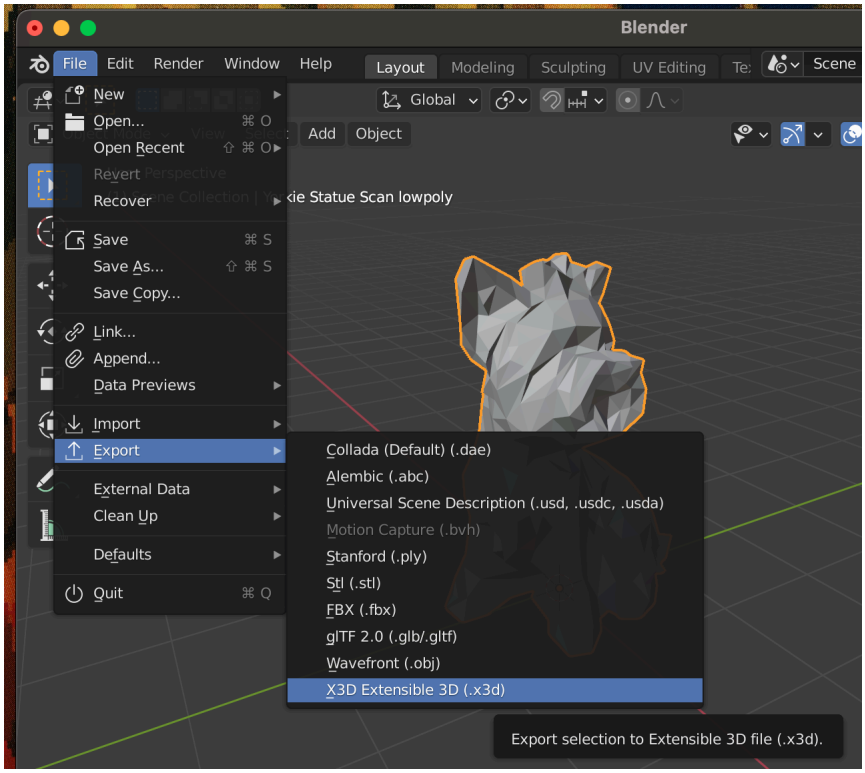
If you know how to use blender THATS GREAT, if you don’t, you can still muddle through.

Startup blender, make a new general file, delete all the default assets it provides. Next import or create the object that you want to convert to X3D.

I have imported a low poly dog I found on the internet!



Next just go to export and export it as an X3D file. THATS IT!



You can not put this file in your site directory and load it in using an inline tag:

<Transform translation='0 0 0' >

<inline url='dog.x3d' />

</Transform>

OK, but you'll want to do some edits to it, so open up the dog.x3d file in your editor and you'll get a long intimidating list of stuff. Much of it is the same as in our main x3d file and I have already explained it.

Feel free to delete the **<NavigationInfo>** and **<Background>** tags from the new file as those will not be used.

I imagine the first thing you will want to do is modify the size of scale of your new object. To do that look for the first **<Transform>**, it should have a “scale” attribute. Increase this to modify the size. OR put a scale attribute on the transform in your main x3d file to accomplish the same thing (this is what I typically do).

```
<Transform translation='0 -25 0' scale="1 1 1" >
```

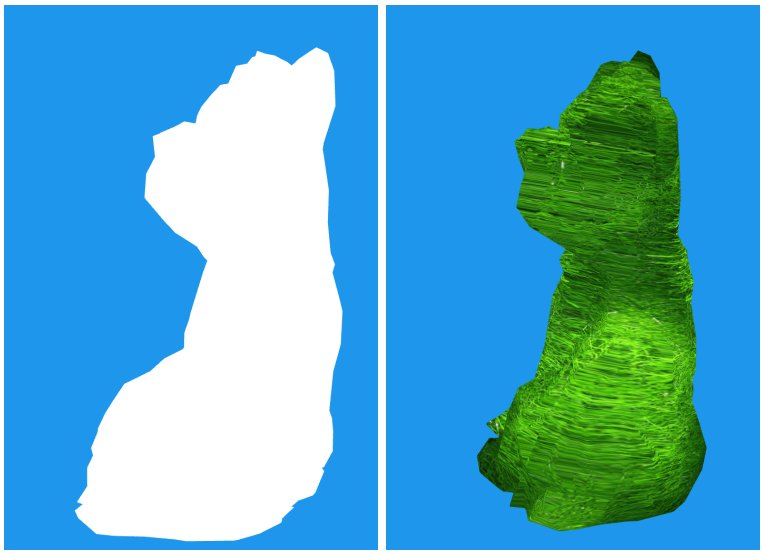
OK NEXT, you’ll prob want to give your new object a texture. This can be done in blender, and you may want to do this from blender to get more advanced texture options. If you did that you should already have a texture, make sure the texture files blender exported are in the right place in your site directory!

I will quickly explain how to manually add a texture though.

If your export had no texture, you’ll see an empty **<Appearance>** tag just before a bunch of numbers. This is where our texture info goes.

```
<Group DEF="group_ME_Yorkie_Statue_Scan_lowpoly">
    <Shape>
        <Appearance>
        </Appearance>
        <IndexedFaceSet solid="false"
                        coordIndex="0 1 2 -1 3 4 5 -1 5 6 3 -1 7 8 9 -1 9 10 7 -1 0
                            3 18 -1 19 20 21 -1 21 22 19 -1 23 8 7 -1 0 12 16 -1 19
                            31 -1 31 34 35 -1 35 36 31 -1 37 17 16 -1 17 37 38 -1 24
                            -1 19 47 48 -1 48 20 19 -1 27 30 31 -1 31 49 34 -1 50 51
                            59 -1 62 19 24 -1 63 64 65 -1 65 25 63 -1 31 30 66 -1 34
                            76 73 -1 77 44 15 -1 78 12 23 -1 12 78 16 -1 79 16 78 -1
```

Inside these tags simply add **<ImageTexture url='"grass.jpg"' />** to give the object an image as a texture! There is a little more to it than this if you want to get better control of the texture, see the hill.x3d file in the examples for all that extra stuff.



Javascript Linking

You can run Javascript from within X3D! Here is an example of how to do a message console, so you can click stuff and have messages appear.

You can make functions on your index.html page and call them within X3D. On the index page put this:

```
<!-- This is a console style message system and an example of javascript you can call from X3D -->
<div id="messages"></div>
<script>
    let textbox = document.getElementById('messages');
    let textStream = "";
    function txtDisplay( message )
    {
        textStream = message + '<br/>' + textStream;
        textbox.innerHTML = textStream;
    }
</script>
```

Now on your X3D page, add an onclick to the object you want to click to show a message.

```
<inline url='assets/hill.x3d' onclick="txtDisplay( 'You have clicked the Hill!!!' );" />
```

The Result? “You have clicked the Hill!!!” should now have been added to the messages div. You can style the div any way you want. You’ll probably want to make it floating over your x3d page so you can see it! You can do all that with css.

You can also use this method to load links and pages from within X3D. Just look up “How to load a page from javascript” and I'm sure you’ll figure out how.

How to add SOUNDS?

You can add sound objects into the world with a SOUND tag. I need to go into this in more depth, but I have included a basic example in the files for playing world music.

Adding Multiplayer via a server

Yes this is possible, I will talk more about it at a later time!