Build a (virtual) world in half a day workshop handout – ASTC 2017

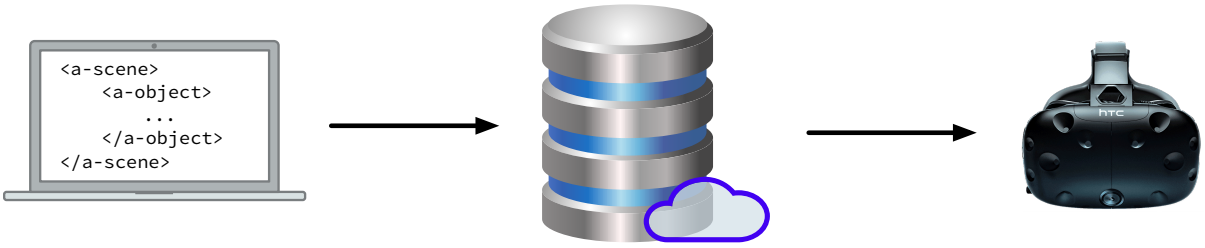
Geva Patz <geva@making.do>

Thanks for attending the VR workshop I presented at ASTC. Here's a brief recap of the software we used and some links and resources if you'd like to take things further. I enjoyed talking to many of you afterwards about your projects and thoughts; if you ever have any other questions or if you'd just like to bounce ideas off someone, don't hesitate to contact me at the email address above.

Software setup

We used the ***A-Frame VR framework*** (https://aframe.io/) which provides an easy to use yet surprisingly powerful set of primitives and tools for creating virtual experiences which are delivered over the web and can run on everything from a basic phone/Cardboard viewer to a high end headset like the Vive or the Oculus Rift.

One of the aspects of A-Frame which contributes to its ease of use is that you can specify a virtual scene using HTML tags just like a web page. The workflow looks like this:



Develop scene(s) **Upload to web server** **Device downloads from web server**

The web server could be anything public server on the Internet, a server within your institution, or a $25 Raspberry Pi mini-computer set up as a dedicated VR server for an exhibit.

The only thing you need to develop an A-Frame scene is a ***text editor*** to create and save the HTML file(s) with. You can use any text editor for this, even the built in TextEdit (Mac)/Notepad (Windows), but it's generally easier to use one which understands HTML well enough to provide features like coloured highlighting of tags. There are lots of options, many of them free; which one is 'best' is entirely a matter of personal taste. The one we used in the workshop is ***Atom*** (https://atom.io/).

When initially testing an A-Frame scene, it's often helpful to run a mini web server on your own machine. There are lots of options for doing this; the one we used in the workshop was

the ***easyrtc-server***[1] which is written in JavaScript and runs under an application framework called ***node.js***. If you ever need to reinstall these, there are comprehensive instructions for all supported operating systems at https://github.com/priologic/easyrtc (click on the 'Install instructions' link).

Finally, we installed a piece of 'bridge' software called ***ngrok*** (https://ngrok.com/download) This takes the mini web server running on your laptop and makes it accessible to browsers which are running elsewhere, be it on a phone in the same room or halfway across the internet. It also allows you to access your local machine using the secure 'https' protocol, which is necessary to get features like video overlays to work on some phones.

Running the server (Mac version)

Assuming you have the mini web server and ngrok installed as above, then the following steps should work on a Mac:

1. Open a Terminal window (Applications > Utilities > Terminal from the Finder)
2. In the Terminal window, change to the directory (folder) which has your VR documents, for example (this assumes your files are in a 'vr' folder on the desktop; change the directory name as needed):

   ```
   cd Desktop/vr/
   ```

3. Run the mini web server by typing the following command in Terminal:

   ```
   node ./server/easyrtc-server.js
   ```

4. Open a second Terminal window
5. In the second terminal window, change directory to wherever you have copied ngrok to (the example assumes it's also in the 'vr' folder on your destkop) and run ngrok:

   ```
   cd Desktop/vr/
   ./ngrok http 8080
   ```

Running the server (Windows version)

Assuming you have the mini web server and ngrok installed as above, then the following steps should work on Windows:

---

[1] For more technical users: I chose easyrtc-server because, unlike many basic local web servers, it supports the WebSockets protocol well. This is not necessary for single user A-Frame scenes but is necessary for shared scenes. As the name suggests, easyrtc-server also supports WebRTC, which is another option for shared scenes. I used the WebSockets option in the workshop because WebRTC takes a bit more work to get it to play nicely with phone security models.

1. Open a 'node command prompt' from the node.js selection on the Start menu
2. In the command prompt window, change to the directory (folder) which has your VR documents, for example (this assumes your files are in a 'vr' folder on the desktop; change the directory name as needed):

   ```
   cd Desktop\vr\
   ```

3. Run the mini web server by typing the following command in Terminal:

   ```
   node server\easyrtc-server.js
   ```

4. Open a second node command prompt
5. In the second command window, change directory to wherever you have copied ngrok to (the example assumes it's also in the 'vr' folder on your destkop) and run ngrok:

   ```
   cd Desktop\vr\
   ngrok http 8080
   ```

Using the server

You should see something like this in the ngrok window:

```
Session Status                online
Version                       2.2.8
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://12766355.ngrok.io -> localhost:8080
Forwarding                    https://12766355.ngrok.io -> localhost:8080
```

Use the address on the https line (highlighted in red above) to access your server. If your main file is called index.html it should just load; if not, you'll have to include the file name in the URL you type (e.g. https://12766355.ngrok.io/scene.html)

VR example files

If you need to re-download the example files used in the workshop, you can download it from https://making.do/astc-vr.zip The VR scene files and resources are in the vr/server/static folder (or vr\server\static if you're on Windows).

If you haven't yet done so, try the 11shared.html demo page. Run easyrtc-server and ngrok, then take more than one phone (or a phone and a desktop computer browser) and visit https://12766355.ngrok.io/11shared.html (replacing the '12766355' with whatever random string of letters and/or numbers ngrok shows you). You'll be in a shared version of the workshop world, where each participant can see a basic 'avatar' of the others (a blobby head with eyes).

If you open the 11shared.html file in atom, you'll see that it doesn't take much to turn a single user world into a shared one, Most of the extra HTML is just defining the shape of the 'head' avatar, and there's a small and completely optional piece of JavaScript near the top which helps new users 'spawn' in a sensible location, but the heavy lifting is done by just adding three new JavaScript include lines at the top (networked-aframe.min.js, easyrtc.js and socket.io.min.js) and a `networked-scene` attribute to the `<a-scene>` tag.

The 11shared.html scene also demonstrates more sophisticated animation and a video overlay on the airship.

Links and resources

The A-Frame site (https://aframe.io) has many examples online, along with good documentation (under the 'Docs' link), a list of add-on components which extend A-Frame's functionality (under Community > Ecosystem > A-Frame components) and a nice introductory tutorial at https://aframe.io/aframe-school/ This covers similar ground to the workshop; I highly recommend working through it if you would like a refresher.

If you need more phone VR viewers, there are literally hundreds of options now. I like the ones from Unofficial Cardboard (https://www.unofficialcardboard.com/collections/all) The viewer I handed out at the workshop is their 'e-lite' viewer, which is very compact and is one of the few designs I've encountered which provides holes on the sides so you can stick your thumbs through and manipulate the phone screen whilst it's in the viewer. Their '2.0' model is sturdier, has adjustable lenses (which can make viewing more comfortable for some users) and supports an optional elastic headband for hands-free operation. There's also a larger '2.0+' model for larger phones.

Half a step above the Cardboard headsets literally made out of cardboard are a host of slightly sturdier and more comfortable ones made out of plastic. To be clear, these offer exactly the same visual experience, but are more durable and more comfortable for extended use. One example is the Freefly VR beyond (hltps://freeflyvr.com/us/shop-us/) but a quick Amazon or Google search will turn up many others.

The next step up in both cost and capability are viewers like the Samsung Gear VR (http://www.samsung.com/global/galaxy/gear-vr/) and the Google Daydream View (https://store.google.com/product/google_daydream_view). These are limited to a handful of Android phones, and because they're still using a phone screen as the display device the

overall visual resolution will not be any higher, but they do both come with a controller[2] which expands the range of potential interaction significantly.

Finally, there are the two (current) high end headsets, the HTC Vive (https://www.vive.com/) and the Oculus Rift (https://www.oculus.com/rift/) I personally prefer the Vive because it supports a significantly greater range of movement, its software and tracking hardware systems are more open, and because it doesn't send your private data to Facebook whilst in VR (https://www.roadtovr.com/oculus-vr-privacy-policy-serves-needs-facebook-not-users/) The Rift is a highly capable device, however, and is a bit cheaper (as of the time of writing, at least – pricing is an ever-moving target) if you're willing to trade dollars for privacy.

Bear in mind that both the Vive and the Rift need a reasonably beefy computer/graphics card to run VR applications; read the minimum specs for each and budget an extra $800-1000 for a VR computer if you don't already have a suitable machine handy. You get what you pay for, though; the high end VR headsets are a substantial step up in visual quality, interactivity and overall immersion.

The headset market is constantly evolving, so if you're reading this many months after the workshop it's bound to be out of date. There are, for example, standalone Vive and Oculus headsets in the works, which should provide a middle ground between phone based headsets and ones which need a powerful PC to run the software. If you're planning a longer term (1+ years) VR project and would like an educated guess as to what level of capability you can plan for, feel free to contact me for some free handwaving pontification.

If you want a taste of the very near future of phone level capability, check out the video at https://youtu.be/ludoKrg2It8 which demonstrates using the new AR features in Apple's new iOS 11 to provide not just head angle tracking but head position tracking in 3D space. This allows for much more immersive VR which users can literally walk through, and supports both VR and Augmented Reality (where the real world is mixed with virtual objects)

---

[2] Early versions of the Gear VR did not include a controller, but the latest version does (with a corresponding price bump from $80 to $100 to match the Google Daydream)