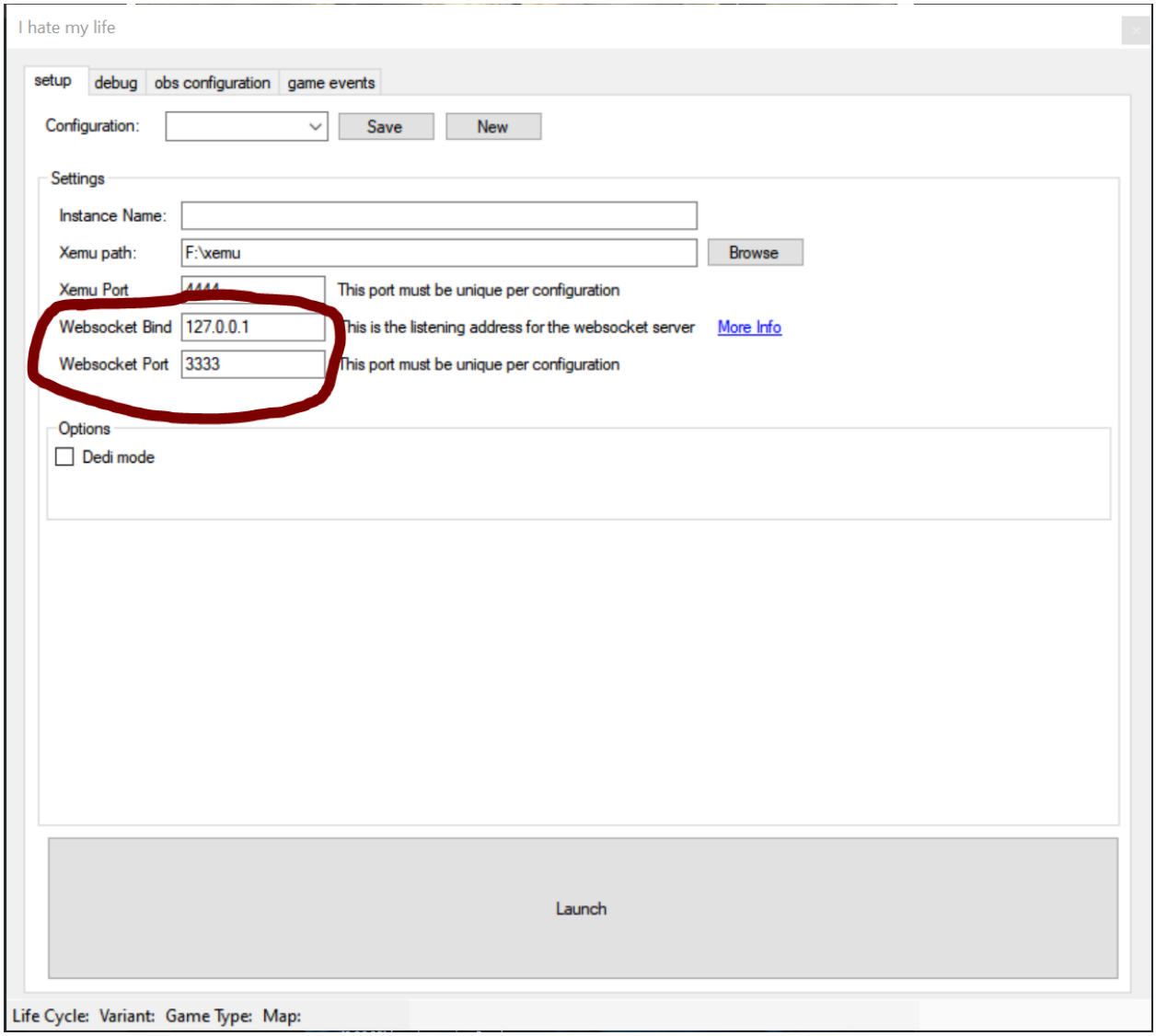
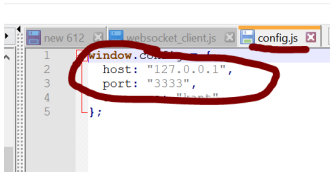
Instructions to use scoreboard overlay:

The scoreboard overlay displays each player's kills, assists, and deaths, as well as what weapon they are actively using, whether they are alive or dead, and what emblem they have.

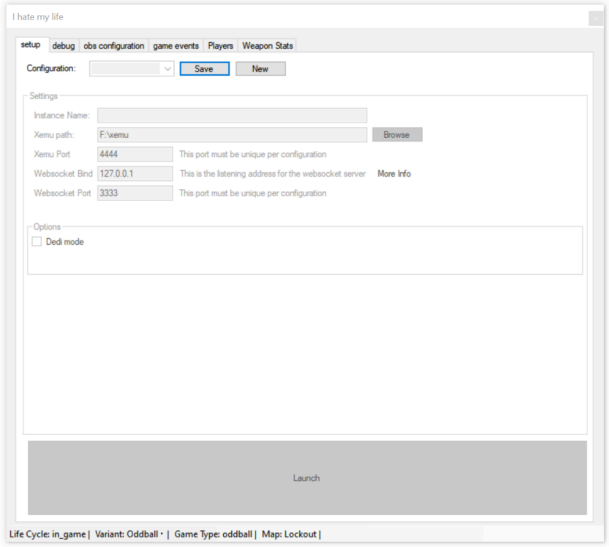
This very simply is a web page that currently is set up to run on the same computer as the dedicated server, so when you launch XEMU the websocket ip address is bound to 127.0.0.1 (which is the localhost, which is yourself), and to port 3333 (we just had to pick one).

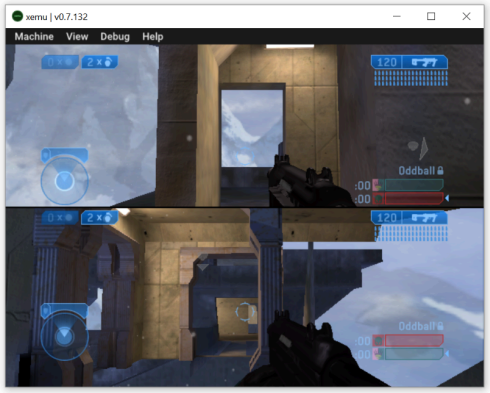


If you want to set a different IP or port, which you would only want to do if you wanted to run the scoreboard or any other overlay on a different computer than the dedi computer, you need to edit websocket_overlays/config.js to match what you set in the app launch window.

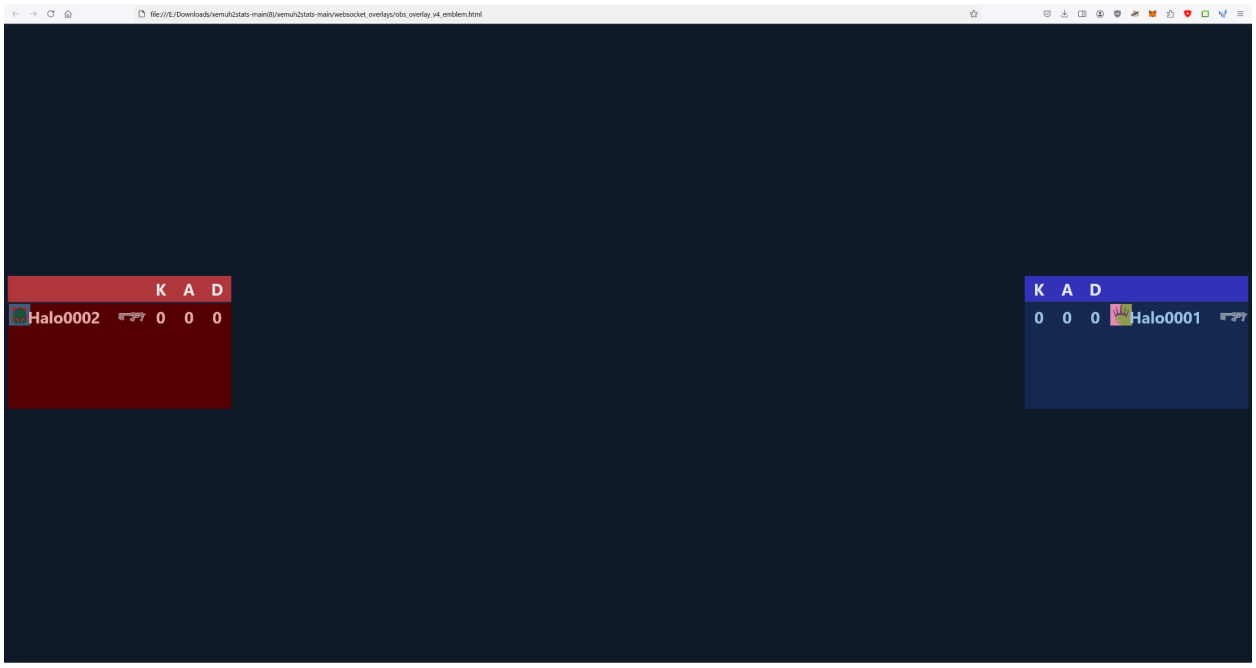


Now launch the game and for easiest testing I just do a two player splitscreen game not in a dedi setup. Once I have that running I just open websocket_overlays/obs_overlay_v4_emblem.html in a browser on my computer to verify it is running as expected.



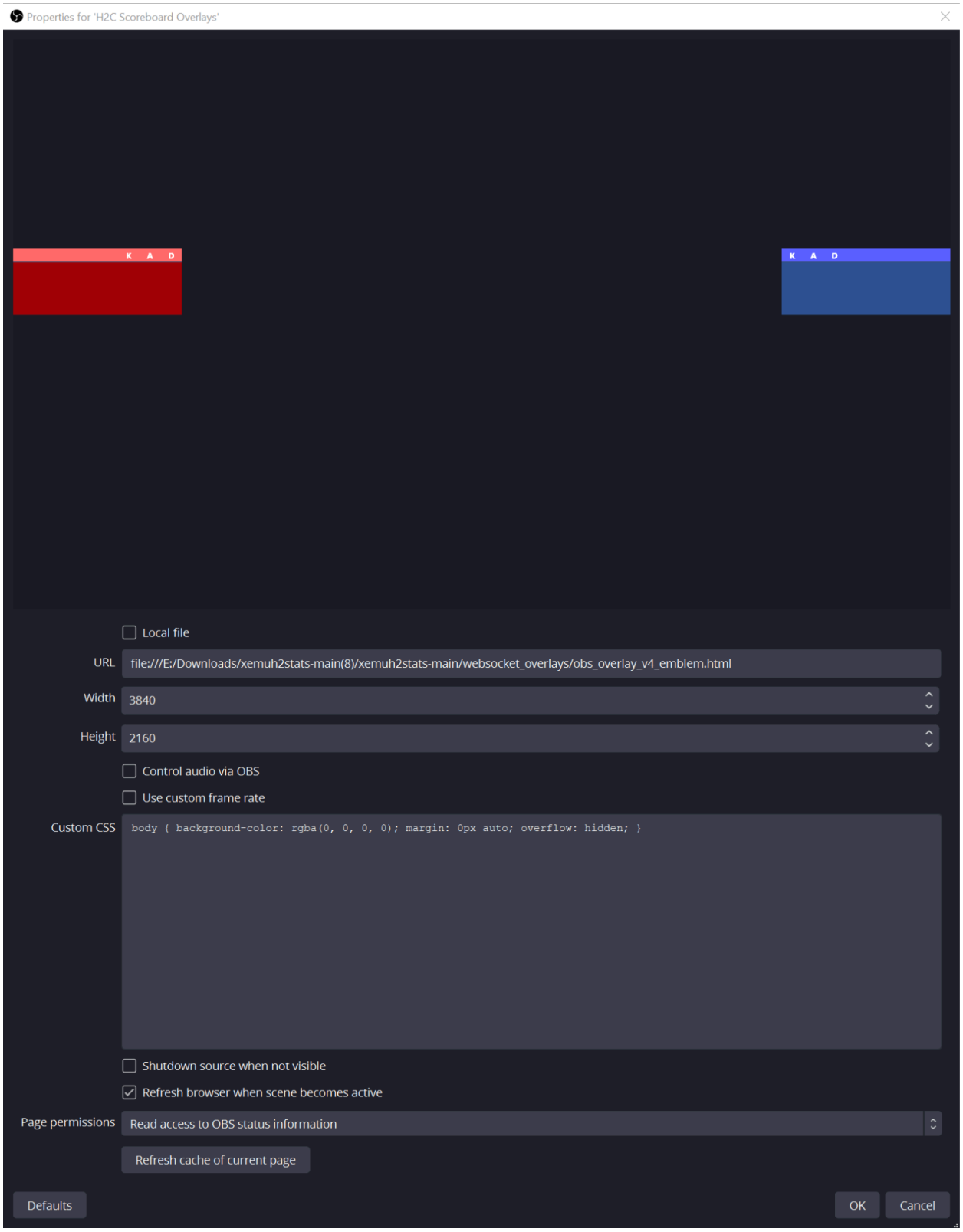


Above is me just running the test game, and below is me verifying websocket_overlays/obs_overlay_v4_emblem.html works in firefox.



Now we can add this into our OBS scene. To do this, add a new source that is the Browser type. In the browser type, take the url in firefox and copy that into the URL field. This is what worked for me, but you can also try just referencing the local file directly but no guarantees that works the same way.

The other thing I did is double the width and height of this source to better scale the scoreboard with respect to the stream. For my 1920x1080 canvas, I went with a 3840x2160 Browser source that is then scaled down to fit the canvas. Here are my settings.



As you can see the default custom CSS gets rid of the background and allows the scoreboard background colors to be a little transparent too. **Note I also make sure to refresh the browser**

when the scene becomes active. If the scoreboard seems to no longer update, just hide and bring back this browser source to refresh the page and it should be good to go. And here we go we have a working scoreboard overlay.

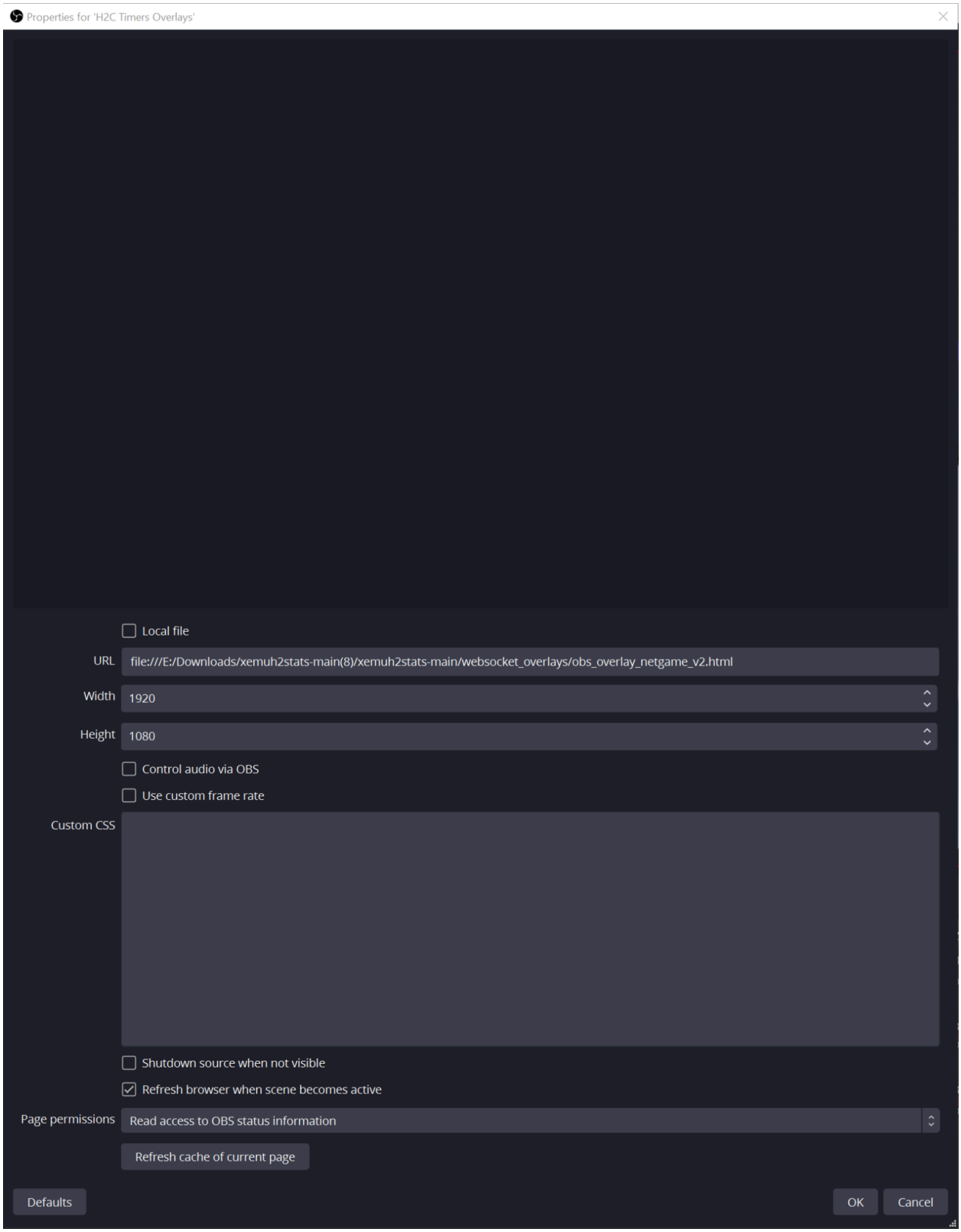
If you want to manipulate the scoreboard the easiest method is probably to manipulate the source. You can crop it around each scoreboard, make a duplicate so you have both, and move them around as you desire if you want. You can also edit the HTML or CSS in the file itself but the deeper you go into that, the more likely some issues will arise. Javascript interfaces with the websocket layer and if there's no div to put the text and images then those will just stop working.

Instructions to use weapon timers:

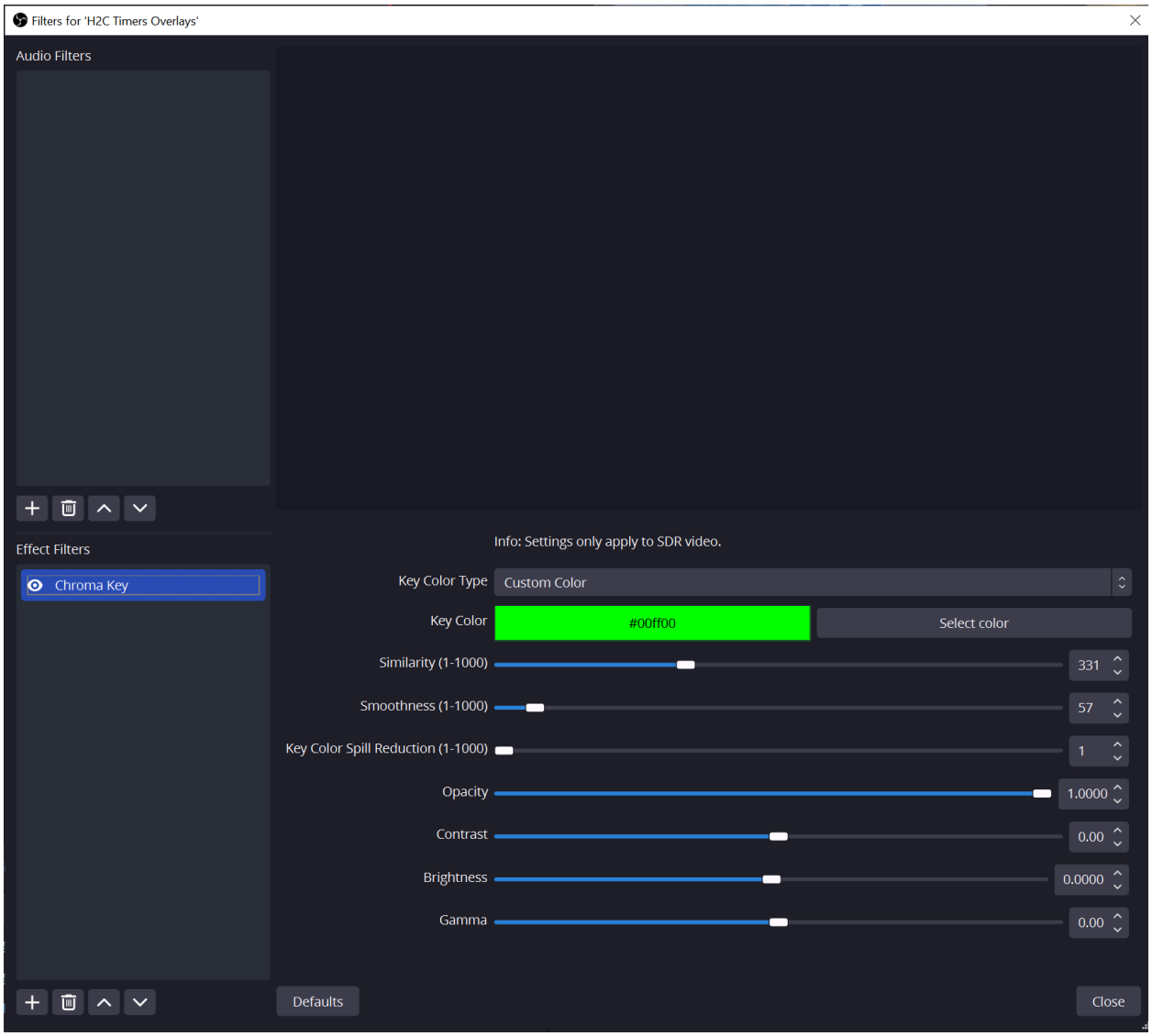
Another cool feature we have is an overlay that pops up whenever a power weapon is going to respawn. This feature I would still like to fine tune some but is very useful from viewers and casters, knowing whether or not players are correctly making a push for a power weapon. Similar to the scoreboard, you just simply have to add a source for the file websocket_overlays/obs_overlay_netgame_v2.html.

The main differences with this one though is you do not need to overscale it like you do with the scoreboard, and you have to add a filter to chroma key out the background (basically applying a green screen). I don't know why but the same background as the scoreboard just did not seem to work with this and that might have to do with this respawn timers only appearing when

necessary. Here’s what my source looks like.



And here’s how that chroma key filter is set up:

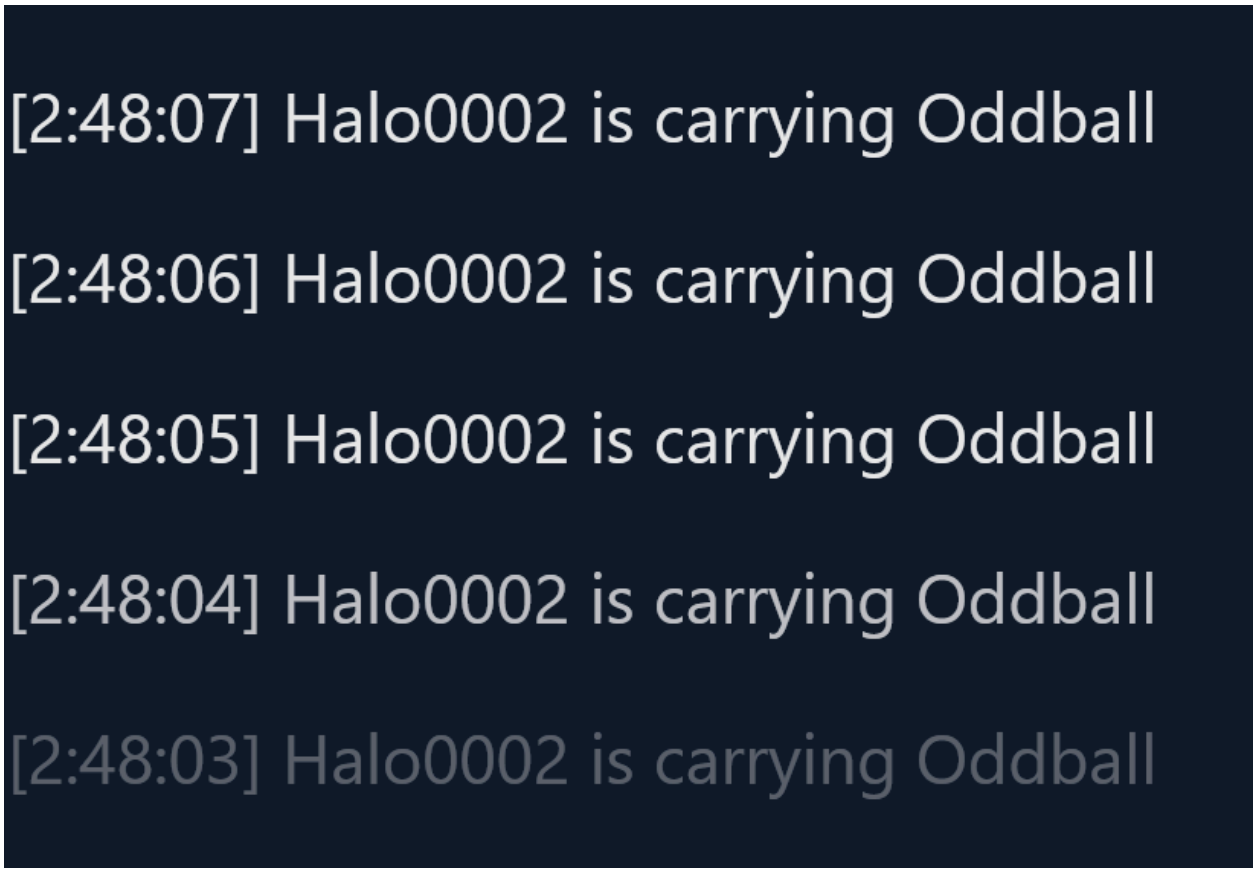


And that’s it. To test it you can pick up a power weapon and throw it off the map. Be patient, but about like a minute later you should see a timer appear in the bottom left corner. When there are multiple power weapons on respawn they will all appear in order down there.

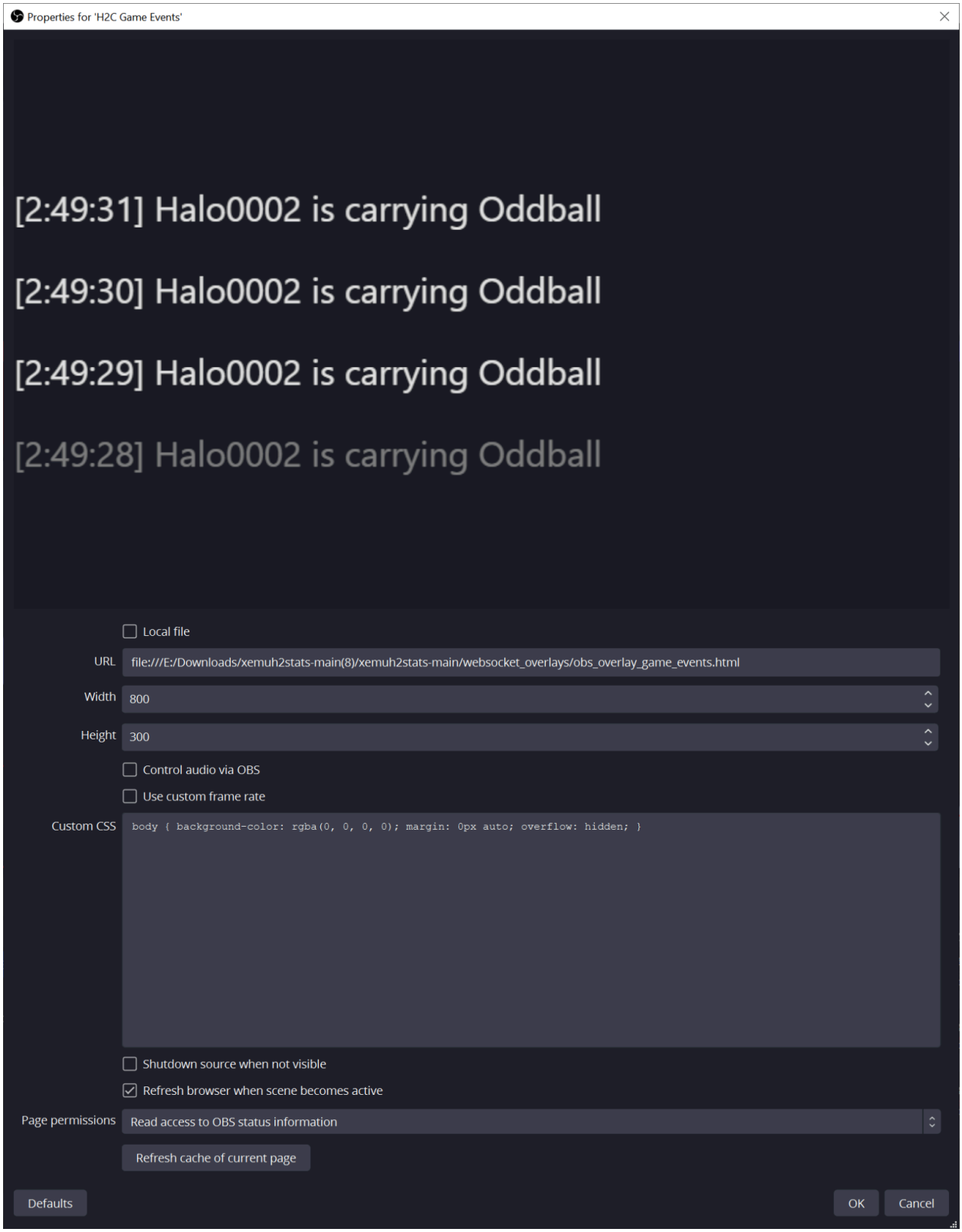
Instructions to use game event logs:

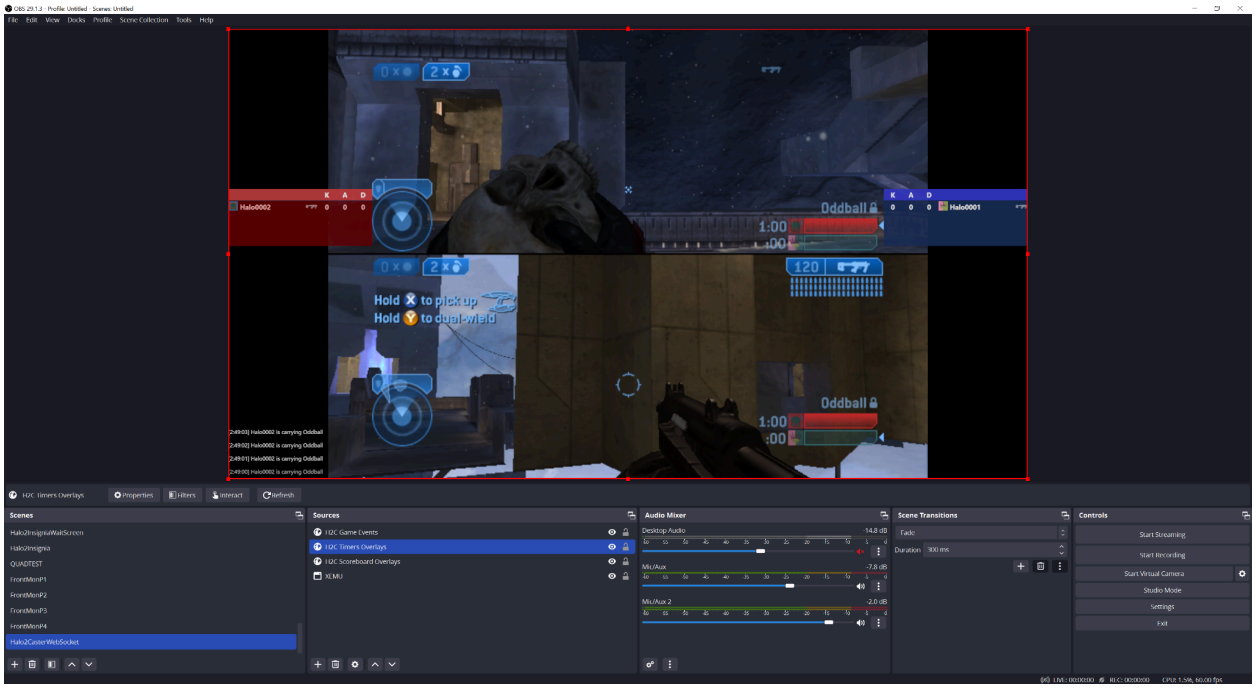
The latest thing we have, Kant built out but I haven’t fully designed is the game events. Not terribly important as the in-game HUD features nearly the same info but it can allow you to put

this anywhere on screen and make it more legible and not team biased. The page for this is websocket_overlays/obs_overlay_game_events.html.



The browser source I set to a resolution of 800x300, that seemed to more or less crop the page to where the text would be. You can play around with this but this source is not one you would just have stretch to fit over the whole canvas and instead should just be in one pocket.

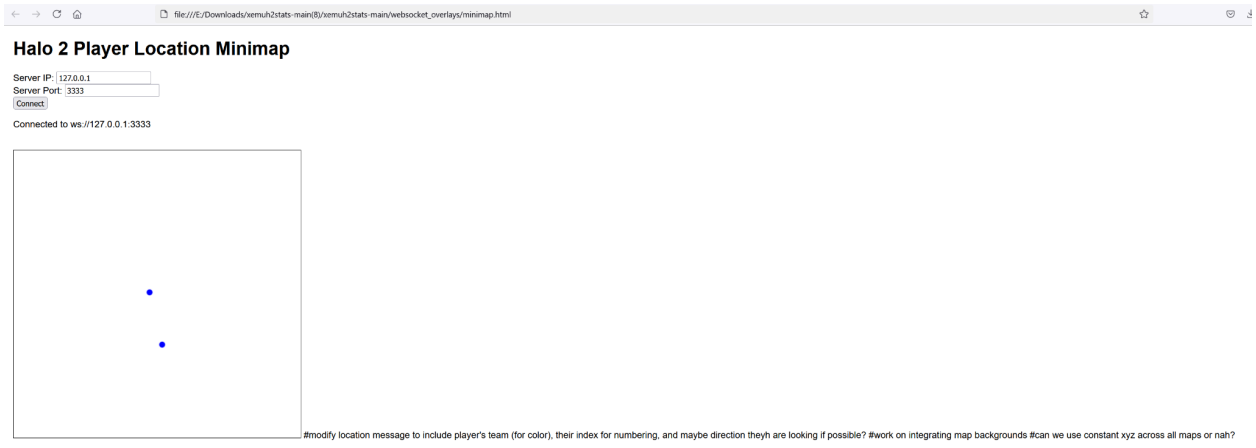

Properties for 'H2C Game Events'
[2:49:31] Halo0002 is carrying Oddball
[2:49:30] Halo0002 is carrying Oddball
[2:49:29] Halo0002 is carrying Oddball
[2:49:28] Halo0002 is carrying Oddball
Local file
URL
file:///E:/Downloads/xemuh2stats-main(8)/xemuh2stats-main/websocket_overlays/obs_overlay_game_events.html
Width
800
Height
300
Control audio via OBS
Use custom frame rate
Custom CSS
body { background-color: rgba(0, 0, 0, 0); margin: 0px auto; overflow: hidden; }
Shutdown source when not visible
Refresh browser when scene becomes active
Page permissions
Read access to OBS status information
Refresh cache of current page
Defaults
OK
Cancel



Instructions to use minimap:

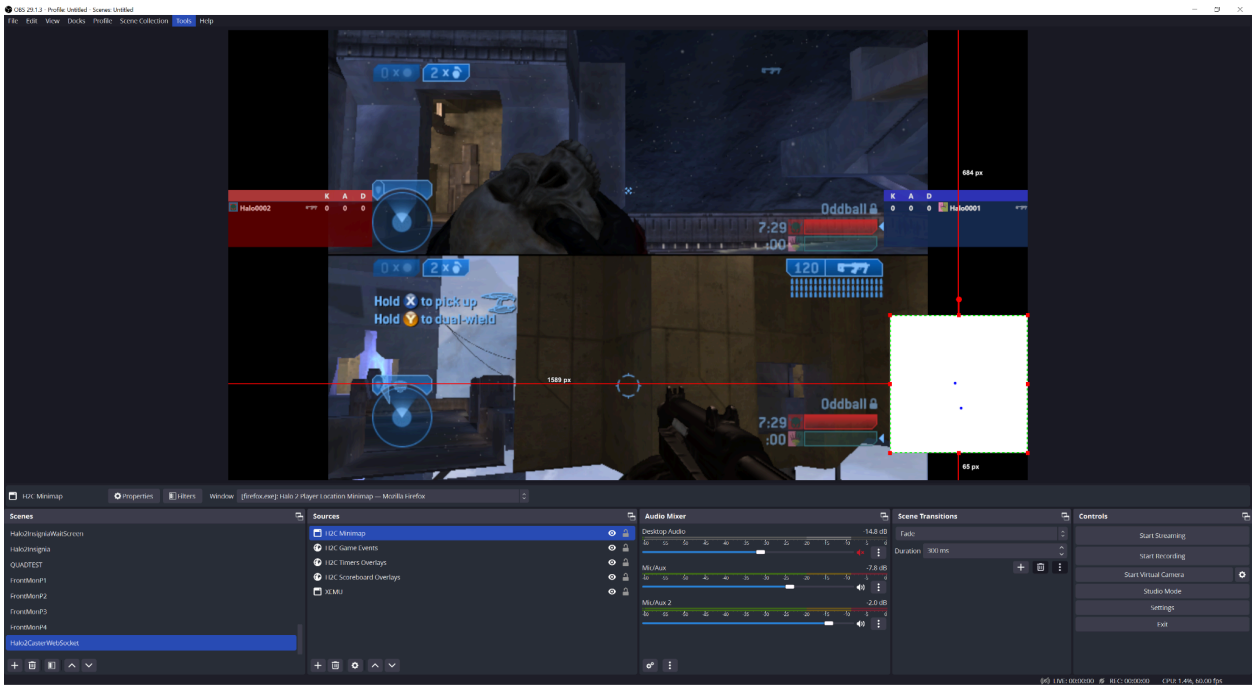
Now this is the least developed overlay, but maybe the coolest. Basically you can get the player's represented as dots and have that update in realtime, but currently there is no "map" for that to be displayed over. So I don't expect this to get used yet but figured I would share. The page for this is websocket_overlays/minimap.html.

This page requires entering in the ip and port and pressing connect first, so it might make more sense to leave this open in a browser, and then add this a a window source and not a browser source.

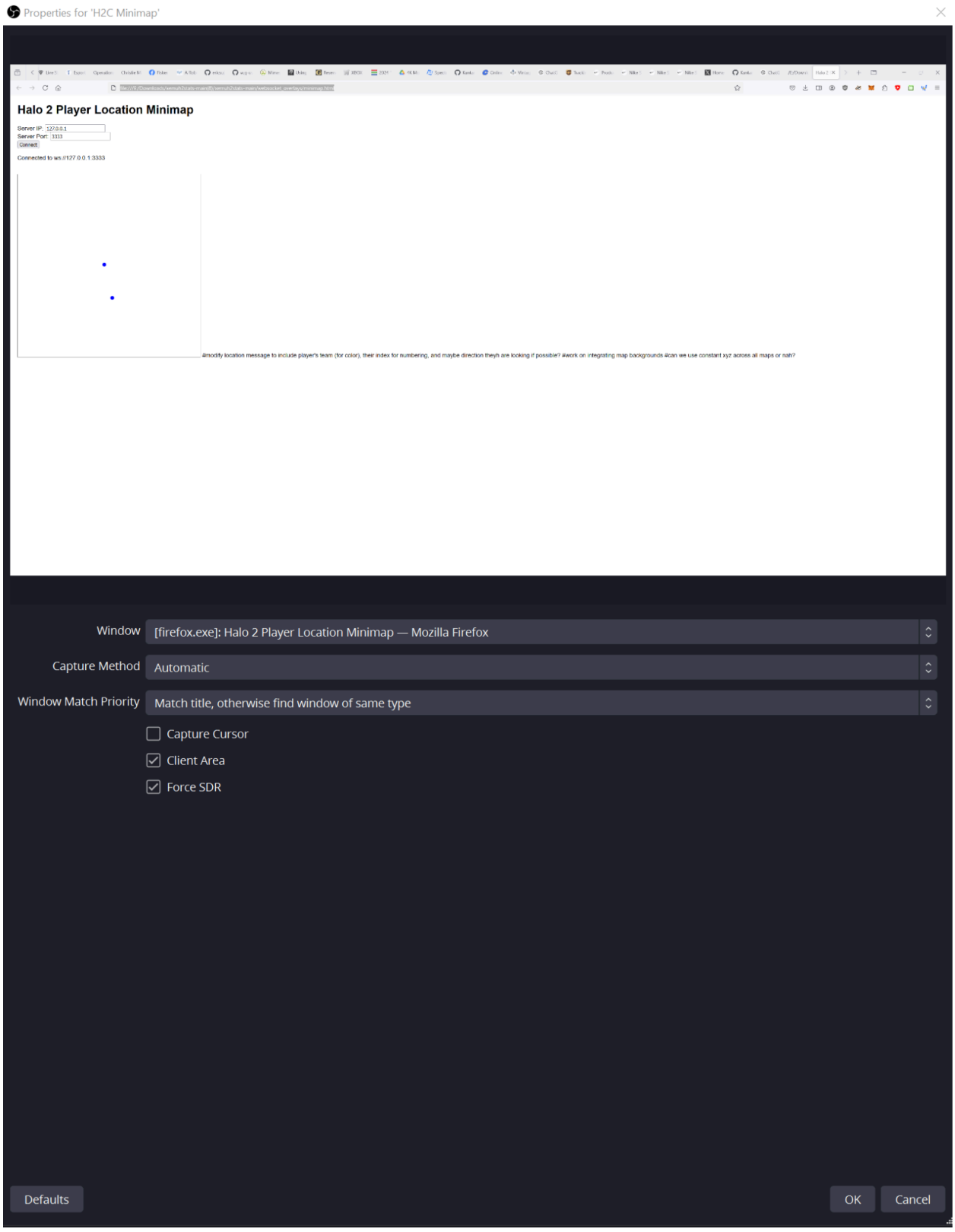


#modify location message to include player's team (for color), their index for numbering, and maybe direction theyh are looking if possible? #work on integrating map backgrounds #can we use constant xyz across all maps or nah?

Those notes are for me to add to it. But yeah you can add the window source and crop it to something like this:



And here's the source settings:


Properties for 'H2C Minimap'
Halo 2 Player Location Minimap
Server IP: 127.0.0.1
Server Port: 3333
Connect
Connected to ws://127.0.0.1:3333
#modify location message to include player's team (for color), their index for numbering, and maybe direction theyh are looking if possible? #work on integrating map backgrounds #can we use constant xyz across all maps or nah?
Window
[firefox.exe]: Halo 2 Player Location Minimap — Mozilla Firefox
Capture Method
Automatic
Window Match Priority
Match title, otherwise find window of same type
Capture Cursor
Client Area
Force SDR
Defaults
OK
Cancel

But like I said this one is the most WIP and I don’t expect it to be used on a stream. If you want to test it out and give me feedback though I would certainly welcome it. The dots represent the player colors and they get bigger and smaller depending upon how high or low they are in the map, relatively speaking.