Streaming (media) in NodeJs

Like the title suggests I tried to write a simple http server that is able to stream[1] content to the browser. Main reason for this is that chrome is annoyingly bitchy when it comes to playing sounds using HTML5. You can load and play a sound once no problem, but the problem is the “once”. As you can read here[2] HTML5 and audio is already a complicated thing but in my opinion chrome goes one step further and is dependand on a webserver that is able to stream the audio files.

At first I tried it using this google group discussion here[3] but failed miserably (at least under chrome). I tried to get my head around all those crazy http header flags[4] but I did not get it right..

Then I finally got the right clue from a blog post[5] describing streaming using the node express framework. But since I have written the rest of the webserver (a simple one that is) without express and because I am notoriously curious I did not want to throw everything out of the window and start over using express.

So last but not least I got it working and here is some code to grind:

```
var app = require('http').createServer(function(request, response){
        //...yada yada yada...
        // get file name
        //...yada yada yada...
        fs.readFile(filename, "binary", function(err, file) {

                var header = {};
                // add content type to header

                //TODO: any more clean solution ?
                if(typeof request.headers.range !== 'undefined')
                {
                        // browser wants chunged transmission

                        var range = request.headers.range;
                        var parts = range.replace(/bytes=/, "").split("-");
                        var partialstart = parts[0];
                        var partialend = parts[1];

                        var total = file.length;

                        var start = parseInt(partialstart, 10);
                        var end = partialend ? parseInt(partialend, 10) : total-1;

                        header["Content-Range"] = "bytes " + start + "-" + end + "/" +
(total);
                        header["Accept-Ranges"] = "bytes";
                        header["Content-Length"]= (end-start)+1;
                        header['Transfer-Encoding'] = 'chunked';
                        header["Connection"] = "close";

                        response.writeHead(206, header);
                        // yeah I dont know why i have to append the '0'
                        // but chrome wont work unless i do
                        response.write(file.slice(start, end)+'0', "binary");
                }
                else
                {
                        // reply to normal un-chunked request
                        response.writeHead(200, header );
                        response.write(file, "binary");
                }

                response.end();
        });

});
app.listen(80);
```

Important to note are just a couple of things that took so long to find out:

1. set ‘Transfer-Encoding’ to ‘chunked’
2. set ‘Connection’ to ‘close’
3. send one more trailing byte … I have no idea why 😀

That’s it. I hope this will save other people some time 😉

1. http://en.wikipedia.org/wiki/Chunked_transfer_encoding [↩]
2. http://www.wappworks.com/2012/06/15/the-html5-audio-troubleshooting-guide/ [↩]
3. https://groups.google.com/forum/?fromgroups#!topic/nodejs/gzng3IJcBX8 [↩]
4. http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html [↩]
5. http://delog.wordpress.com/2011/04/25/stream-webm-file-to-chrome-using-node-js/ [↩]