File Formats and Web Interfaces for Image, Sound, Video, and 3D data

M. Colom

miguel.colom@cmla.ens-cachan.fr
http://mcolom.perso.math.cnrs.fr/

July 20, 2013

1 File Formats and Web Interfaces for Image, Sound, Video, and 3D data

The simplest mechanism to present data in a website is generating the final representation in the server (i.e. an image encoded with the PNG standard), send it to the server and let the browser of the client show it. The major benefit of this schema is that the data can be represented in any browser that is able to show an image, that is a capability supported from the first versions of the HTML standard. Also, since the image is generated at the server, its representation doesn't depended on the browser.

However, users can't interact with the data shown in a fixed image. For some simple plots (an noise curve, for example), this interaction isn't needed, but for 3D graphics (a 3D reconstruction, as Google Earth shows) and large 1D signals (a sound signal, for example) some degree of interactivity is demanded. For example, let's think of the case of Google Earth. It's able to reconstruct in 3D a huge area of the planet using elevation data obtained by aerial photographs. Of course, it's not possible to send all the 3D points to the client, since the quantity of points in huge. Also, in the case of a sound file, the signal might contain 22000 samples/second, for example.

Therefore, the classic method of answering the client with a complete response is no longer valid and some interactivity is needed. Technically, what is needed is asynchronous communication and let the client to take the initiative to start the communication and ask the server for the data it needs each moment. For example, in the case of a 3D reconstruction, the client would ask the server for the cloud of points (or triangles, cubes, etc) needed to show a particular view of the data. In the case of a sound signal, it'd ask for the points needed to show a particular zoom of the data at the required level of detail.

Nowadays the most standard way to implement asynchronous communication is using AJAX[1] combined with JSON[2]. These techniques use Javascript in the browser. The JQuery library implements AJAX with a simple Javascript interface.

For the representation of 3D data, the WebGL API is the standard way. There exist some libraries to interact with it. The three.js[3] is one of the most used. The disadvantage of WebGL is that it's relatively new. The last stable version is from 2011. However, this isn't really a problem, since the exact usage of the library can be determined at the server, to avoid functionalities that might no be available in all browsers.

To simplify the interfaces of the programs on the server that need to show 2D/3D data, it's convenient to define some simple structures. For example, in the case of 2D plot, a text file with columns representing the values at the X and Y axes. In the case of a cloud of 3D points, a similar format. In the case of 3D triangles, some fixed format, and so on. In general, most of the demos in IPOL/RunMyCode/etc are likely to match any of these formats. Of course, the formats should be flexible enough to allow some personalization.

In conclusion, it's possible to use advanced and interactive representations of large 2D/3D data in the web browsers by using asynchronous communication (AJAX with JSON using JQuery) and the WebGL for 3D graphics (using the three.js library). The major problem that we're likely to face is the possibility of a demo using formats and representations not provided. In this case, the solution is to not allow the non-standard demo (because of medium/long term maintenance problems) or adapt the accepted formats if more demos are going to use a new representation.

2 Some examples of 3D graphics

https://www.cubeslam.com/crihhx http://www.laplace2be.com/lab/PixelParticles/ http://www.mrdoob.com/lab/javascript/beachballs/ http://lab.aerotwist.com/webgl/surface/ http://lab.aerotwist.com/webgl/photoparticles/

3 Some examples of 2D graphics

http://www.worldwidewhat.net/2011/06/draw-a-line-graph-using-html5-canvas/ http://www.mavenspun.com/javascript/learn-programming/how-to-build-graph-using-html5-canvas-and-json.htm

[1]Asynchronous JavaScript And XML.

[2]JavaScript Object Notation.

[3]https://github.com/mrdoob/three.js/