“How to make a Map”

Drafted on 29-March-2017
Authored by Denis Tsoi <denistsoi@gmail.com>

There are many different tools that will enable to render a map on a web page. Our main focus is to try and review existing mapping tools/services.

Later we will try to extend those tiles to be rendered onto a 3D shape. Afterwords, we will try to place markers on top of the shape/texture and enable interactive features.

Contents:

1.Tools/Candidates:

1.1. Showing a Map on a Sphere (*WebGLEarth/Cesium/D3/Planetary.js*)
1.2. Showing a Map on a 2D Plane (*Mapbox/Deck.gl/Leaflet*)

“How to make a Map”

Drafted on 29-March-2017

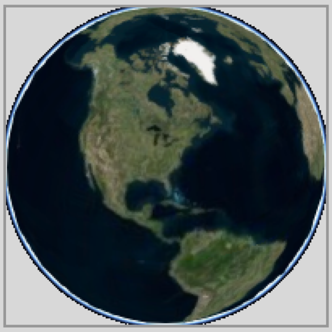
Authored by Denis Tsoi <denistsoi@gmail.com>

1.1. Showing a Map on a Sphere: *(WebGLEarth/Cesium/D3/Planetary.js)*

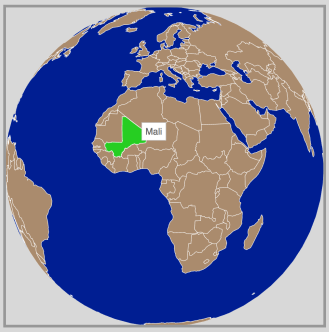
The User Interface with rendering a 2D map onto a 3D sphere is somewhat lacklustre. Rendering a spinning globe may look initially impressive, but the lack of grandular fidelity (i.e. focusing on location, zooming, transitions between locales), presents a technical challenge if we hope to implement such features if they are not supported by those respective libraries.

With D3, we could use an a D3 sphere to generate a globe, but the tranferrance of a 2D plane interaction (Up/Down/Right/Left) presents a slight distortion when trying to maniuplate a 3D projection. *(i.e. challenges with representing a 3D object sphere onto a 2D plane which causes perspective distortion, and vis-versa).*

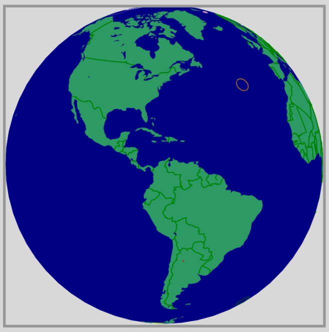
The final consideration with using the above libraries is that, either they are somewhat unmaintained, or outdated. This would indicate challenges later down the development cycle.



Cesium.js



D3 Sphere



Planetary.js

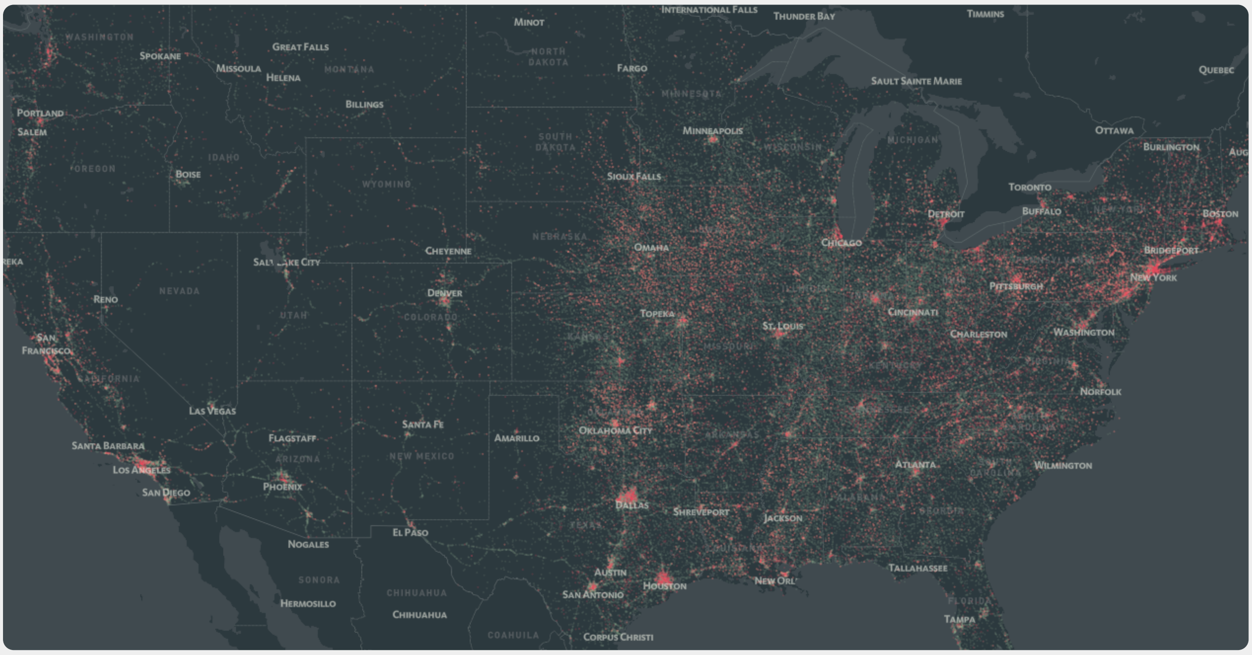
“How to make a Map”

Drafted on 29-March-2017
Authored by Denis Tsoi <denistsoi@gmail.com>

1.2. Showing a Map on a 2D Plane: *(Mapbox/Deck.gl/Leaflet)*

Mapbox

Mapbox is a popular services that provides tools for developers/non-developers to quickly generate a custom interactive map.



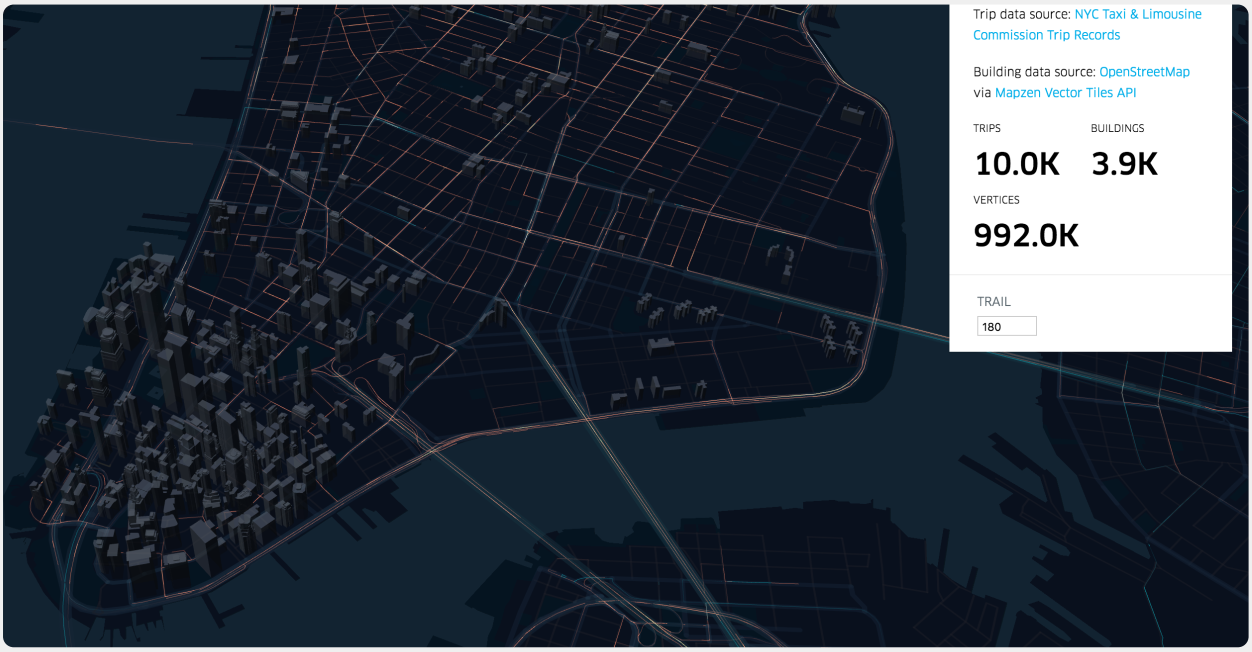
Mapbox with WebGL support.

Before moving to WebGL, *Mapbox* was generated their visualisations with SVGs and could be manipulated with the DOM API. However using SVGs lowers rendering performance, especially inbetween complex interactions.

1.2. Showing a Map on a 2D Plane: *(Cont)*

Deck.gl

Deck.gl is a open source WebGL visualization library that was developed by Uber to manage their driver/passenger routes.



Trip data source: NYC Taxi & Limousine Commission Trip Records

Building data source: OpenStreetMap via Mapzen Vector Tiles API

TRIPS 10.0K

BUILDINGS 3.9K

VERTICES 992.0K

TRAIL 180

Deck.gl

Compared to *Mapbox*, this is a bit more complex (and less support), however, utilizing a visualisation of this magnitude could be thoroughly impressive. The documentation also states that deck.gl can be paired with Mapbox.

Leaflet

Leaflet is another Mapping toolkit, but it’s more useful if say, for rending something that’s more visually simplistic and does not require the need for WebGL rendering.

“How to make a Map”

Drafted on 29-March-2017
Authored by Denis Tsoi <denistsoi@gmail.com>

2. Implementing with Mapbox

There are serveral considerations:

2.1. Familarity with WebGL (*Able to afford more technical debt?*)

2.2. Browser Compatibility. (*Does it work for our clients that are tech-averse?*)

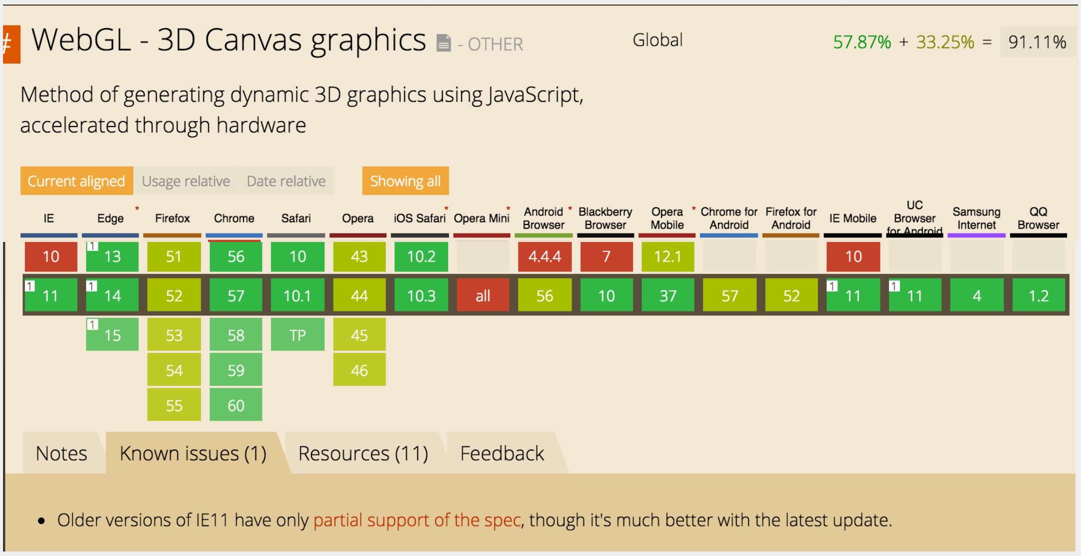
2.3. Meaningful Rendering. (*Does the map make contextual sense?*)

2.4. Hardware Demographic (*Does it render on low-end machines?*)

2.1. Familarity with WebGL
(Able to afford more technical debt?)

WebGL is a browser compatible API for OpenGL. As some developers may not have been exposed 3D rendering projects, there may be some learning curve involved with getting non-familiar developers up to speed, especially if there is a native feature that needs to be implemented (*unsupported by the library*)

2.2. Browser Compatibility.
(Does it work for our clients that are tech-averse?)



WebGL - 3D Canvas graphics - OTHER

Global 57.87% + 33.25% = 91.11%

Method of generating dynamic 3D graphics using JavaScript, accelerated through hardware

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Blackberry Browser	Opera Mobile *	Chrome for Android	Firefox for Android	IE Mobile	UC Browser for Android	Samsung Internet	QQ Browser
10	[1] 13	51	56	10	43	10.2		4.4.4	7	12.1			10			
[1] 11	[1] 14	52	57	10.1	44	10.3	all	56	10	37	57	52	[1] 11	[1] 11	4	1.2
	[1] 15	53	58	TP	45											
		54	59		46											
		55	60													

- Older versions of IE11 have only partial support of the spec, though it's much better with the latest update.

Support Compatibility with Mapbox WebGL

"How to make a Map"

Drafted on 29-March-2017
Authored by Denis Tsoi <denistsoi@gmail.com>

2.2. Browser Compatibility (*Cont.*)

From the previous image, IE 10 does not natively support WebGL Canvas Rendering, and as such, using Mapbox-gl-js (*current stable build as of 30 March 2017*), will require a browser polyfill for those affected browsers (haven't looked at whether there is a fix).

2.3. Meaningful Rendering
(Does the map make contextual sense?)

One of the biggest considerations is how we wish to view and render the amount of information to convey meaning. Rendering a graphical UI with a very narrowed scope or limited amount of data can create a sense of misinformation for the viewer (*i.e. provide no information*).
Conversely, rendering something which provides a vast amount of information may not be able to provide meaning to the targetted viewer. (i.e. information overload). In this scenario, the effort of rendering this information may be an incorrect choice, and further effort to convey meaning may require more effort. (extra charts or using an alternative method of rendering the data).

2.4. Computational/Graphical Limitations

The second consideration with rendering is that, if like above, you wish to render a large amount of information, you may be limited to the performance of the rendering method. An interactive visual interface may also require further considerations to maintain visual performance. (i.e. *FPS*).

First of all, if you anticipate you will be dealing with a large dataset*, then you may need to offset some of the rendering duties to your GPU. This is where WebGL comes into play.

Sources: How to determine if you have a large dataset
[http://stackoverflow.com/questions/18244995/d3-how-to-show-large-dataset]
[http://blog.vida.io/2014/07/06/bigquery-big-data-visualization-with-d3-dot-js/]

“How to make a Map”

Drafted on 29-March-2017
Authored by Denis Tsoi <denistsoi@gmail.com>

3. Conclusion

Supporting IE 10 may present *development problems* - but more importantly is to ascertain the necessary information that is required to be presented/ rendered to derive the most informational value to the user.

However, the user experience will be greatly enchanced, especially from a business administration perspective. *(i.e. navigating between different sites from the application, generating analytic reports based property/sensor categorization).*