gltf in Big Data Visualization

Uber
Our Frameworks

A suite of open-source visualization frameworks
glTF support available throughout the vis.gl stack!
Visualizing Big Data without glTF

- Lines, points or circles
- Not obvious what they represent
Visualizing Big Data with glTF: Model Selection

- Find a model online
- Edit with Blender
Visualizing Big Data with gITF
Adding glTF support to luma.gl

- Reference PBR Shader
- Khronos glTF Sample Viewer
  - luma.gl modular shader
    - Works with geo-coordinate projection system
  - luma.gl code
Adding glTF support to luma.gl: Benefits

- Advanced photorealistic rendering with very little effort
- Consistent visuals with other frameworks
- Allows us to do pixel-to-pixel comparisons of rendering to verify correct rendering
Editing & More...

- Last year we open-sourced nebula.gl
- We combine editing with glTF
- Only ~80 lines of code for this demo

- Future plans: WebVR
LOADERS.GL
Framework Agnostic Loaders for Data Visualization

GET STARTED
The need for a portable glTF parser...

- Rendering glTF was “easy”, thanks to the Khronos PBR reference implementation
- Parsing the glTF data required more effort
- All the resulting parsing code was “WebGL Framework Independent”
- Wouldn’t it be great if there was also a reference implementation for this?
yarn add @loaders.gl/gltf

```
import {load} from '@loaders.gl/core';
import {GLTFLoader} from '@loaders.gl/gltf';
const gltf = await load(url, GLTFLoader);
```

- Returns a javascript object with **typed array** views into the binary chunk
- Optionally decodes and removes Draco encoded meshes
A Growing Family of Framework-Independent loaders

<table>
<thead>
<tr>
<th>GLTF Loaders</th>
<th>Point Cloud Loaders</th>
<th>Other Loaders</th>
</tr>
</thead>
<tbody>
<tr>
<td>@loaders.gl/gltf</td>
<td>@loaders.gl/laz</td>
<td>@loaders.gl/csv</td>
</tr>
<tr>
<td>@loaders.gl/draco</td>
<td>@loaders.gl/pcd</td>
<td>@loaders.gl/arrow</td>
</tr>
<tr>
<td></td>
<td>@loaders.gl/ply</td>
<td>@loaders.gl/zip</td>
</tr>
<tr>
<td></td>
<td>@loaders.gl/obj</td>
<td>@loaders.gl/kml</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

Some loaders are forks from other open source projects, some are newly written.
Loaders: a way to increase community collaboration?

- WebGL will likely remain a multi-framework world
- This is great as it promotes innovation
- But the price is duplication of effort
- If not addressed, our WebGL frameworks will be less innovative and unique
- Can we factor out common WebGL frameworks parts and make them reusable?
First loaders.gl collaboration!

@loaders.gl/3d-tiles: A portable implementation of the 3D Tiles standard

Both Uber and Cesium engineers contributing

Uses both @loaders.gl/gltf and @loaders.gl.draco for 3D model and point tiles

Point Cloud of the Royal Exhibition Hall in Melbourne

18M points in 600+ 3D tiles loaded by Tileset3DLoader, rendered by Tile3DLayer
We are Looking for Potential Users / Partners!

Yes, that means you!