New frameworks

Highlights

Roadmap

Open governance

Ib Green - ib@unfolded.ai
CTO, Unfolded, Inc
https://unfolded.ai

Public Slides. Updated 03.16.2021

Vis.gl is a suite of composable, interoperable open source geospatial visualization frameworks centered around deck.gl.
vis.gl - a platform for geospatial industry collaboration

- Open Source
- Open Governance
- Inclusive

- Multiple major contributors
- Actively inviting new projects

- Quarterly releases
- Weekly patch releases
- Active community forums
vis.gl - An active ecosystem

CARTO for deck.gl: The Best Way to Develop Spatial Apps

INTRODUCING
CARTO for deck.gl

EARTHEENGINE-LAYERS
deck.gl layers for Google Earth Engine for JavaScript and Python
Vis.gl: New frameworks
vis.gl in 2020

JavaScript

dech.gl
luma.gl
loaders.gl
react-map-gl
kepler.gl

Python
pydeck

The kepler.gl demo application is a widely used geospatial tool for visualization and exploration of geospatial data using deck.gl. It demonstrates the power of the kepler.gl application framework, designed to let users build advanced custom geospatial applications.

Providing high-performance, GPU powered visualization layers for large scale geospatial data, deck.gl is the corner stone of the vis.gl framework suite. A selection of submodules provide layers for various geospatial and 3D use cases.

Python bindings for deck.gl that enable powerful custom geospatial visualizations to be programmed directly in Jupyter Notebooks.

A wide range of highly optimized, framework-independent loaders for geospatial, tabular and 3D file formats. Supports worker based binary data loading. Supported outputs includes geospatial layers, point clouds, 3D geometries, images, textures and tabular data.


A React wrapper for Mapbox GL JS that integrates with deck.gl.
vis.gl in 2021

New UCF frameworks!

- hubble.gl
- nebula.gl
- math.gl
- probe.gl
- flowmap.blue
hubble.gl

- Client-side video generation
- Per-frame data loading
- Animation APIs
- Save to GIF, MP4
- Framework agnostic
- Pre-integrated with deck.gl
Geospatial editing

Editors
deck.gl (performance)
Pure React (no WebGL)

Edit Modes
GeoJSON editing
Measurement modes
Custom modes
flowmap.blue

Geographic flow maps
Movement between locations
Incremental aggregation
Interactive exploration
Google Sheets integration
deck.gl layers
React components
Just add **GlobeView** to your deck app

Most deck.gl layers now supported
  Incl. vector tiles (base maps)

Use your favorite deck.gl features
  3D extrusions/objects/orbits
  Transitions
  Animation
deck.gl: Tile Layers

→ TileLayer
  Geospatial
  Non-geospatial
  (gigapixel zoom)
→ TerrainLayer
→ MVTLayer Mapbox Vector Tiles
→ Tile3DLayer 3D Tiles / Handles the tricky corner cases
  Cross-tile highlighting
  Terrain-based feature offset
  Cesium terrain loader
Use with any JS/WebGL framework

Loads into common formats
  - tables, textures, point clouds, 3D...

Easy to write / plug-in new loaders

Optimized for GPU processing
  - Loads into binary data structures
Massive 3D conversion: Frankfurt in I3S format
Massive 3D conversion: San Francisco in 3D Tiles
Massive Pointclouds - Tile3DLayer
Segmentation
And Metadata

- Full format support
- Geometry + metadata
- CPU + GPU analytics
vis.gl 9.0 roadmap highlights
WebGPU (vis.gl 9.0)
luma.gl + deck.gl

Access luma.gl API through
  WebGLDevice
  WebGPUDevice

WebGPU luma classes
Shader module UBO support
Shader transpiler upgrades

deck.gl-native already uses
WebGPU (C++ / dawn)
CUDA (vis.gl 9.0)
For node.js

NVIDIA RAPIDS + vis.gl
(powered by CUDA + OpenGL)

- luma.gl/CUDA buffer interop
- Rendering on node.js
- Video/Event remoting

Announcement April 2021 @
NVIDIA GPU Tech Conf
loaders.gl 3.0

Compressed Textures
- KTX2, DDS, PVR,
- BASIS

High-bit Data Textures
- numpy tiles, etc

Worker support
- processOnWorker

Stream data to workers

Massive 3D data
- I3S 1.7 and 1.8 support

I3S Loader
- MVTLoader
- OBJLoader
- PCDLoader
- CSVLoader
- PLYLoader
- NPYLoader
- DracoLoader
- QuantizedMeshLoader
- GLTFLoader
- ShapefileLoader
- GLTFWriter
- TerrainLoader
- GLBLoader
- VideoLoader
- GLBWriter
- WKBLoader
- WKTLoader
- ImageLoader
- WKTWriter
- ImageWriter
- ZipLoader
- JSONLoader
- ZipWriter
- KMLLoader
- LASLoader
vis.gl + UCF = Open Governance
The Urban Computing Foundation is a neutral forum for accelerating open source and community development that improves mobility, safety, road infrastructure, traffic congestion and energy consumption in connected cities.

Moving vis.gl to a foundation ensured:
- Not controlled of any specific company
- The projects will remain open
- Anyone is welcome use, contribute and influence
- A legal framework that is designed to supports users rather than owners
UCF: Top Level Projects

**Rendering**

**vis.gl**
A framework suite focused on high-performance, WebGL-based visualization of large datasets on web.

**Analytics**

**kepler.gl**
A powerful web application for geospatial analysis and fast exploration of large-scale datasets.

**Basemaps**

**Mapzen**
A set of open source projects with tools to develop fast and responsive maps using open data.
UCF / vis.gl: Governance Structure

UCF Board (Linux Foundation)
Monthly For UCF Members + Linux Foundation reps

UCF TAC (Technical Advisory Committee)
Monthly for UCF Members + Guests

vis.gl TSC (Technical Steering Committee)
Slack based, open to everyone, lead by contributors.
Quarterly open planning meetings
Where the action happens!
UCF: Contributing Projects

Have an open source project with GPU or geospatial focus?

Want to move to Open Governance?

See synergies with vis.gl / UCF projects?

All contribution proposals are welcome!
Join the ecosystem!

Open to everyone!
UCF Membership not required

Engage as
- UCF Member
- Contributor
- User
- Supplier

Join Us!