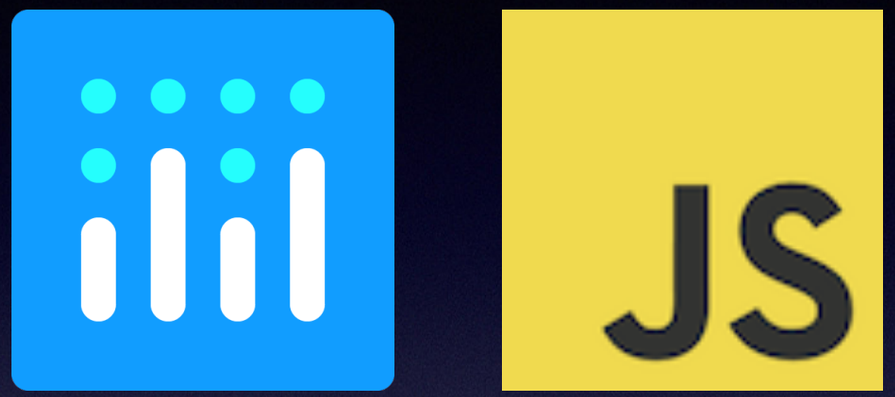


plotly.js

Alex Johnson
CTO - Plotly
github: @alexcjohnson
alex@plot.ly

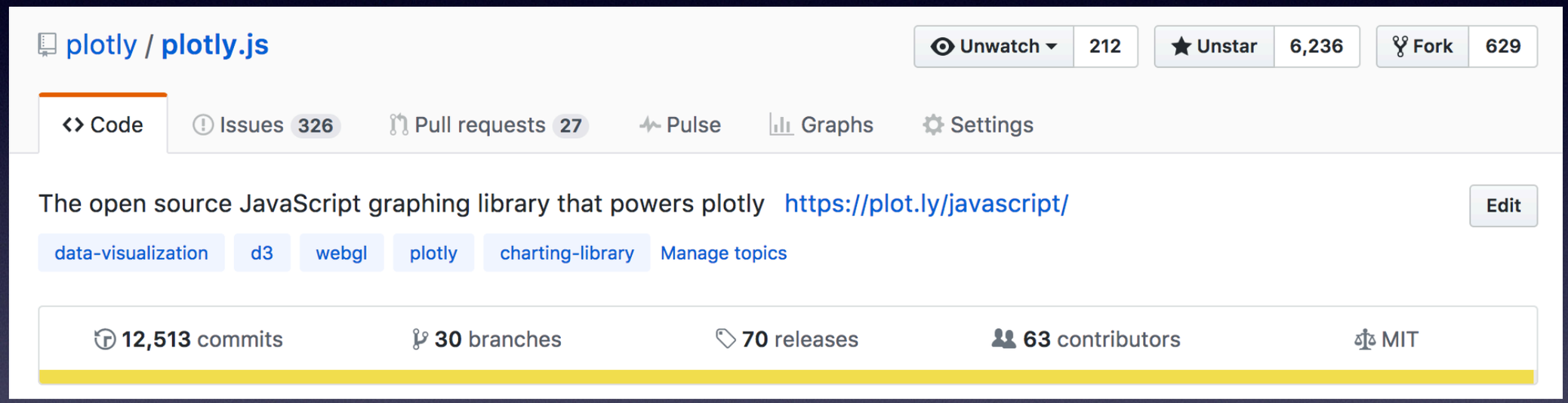
Etienne Tétreault-Pinard
plotly.js lead maintainer
github: @etpinard
etienne@plot.ly

Philosophy

- Open source
- Cross-platform
- Batteries included
- Modular
- Tested

Open source

- https://github.com/plotly/plotly.js



- https://community.plot.ly/c/plotly-js
- https://plot.ly/javascript/
- https://plot.ly/javascript/reference/

Cross-platform

- Graphs are where data is most closely connected to its meaning - so use graphs to communicate between people even in different languages
- Pure JSON declarative chart representation
- example: https://plot.ly/~alex/2210
- Python, R, Matlab, Julia, node/js, … and more!

Batteries included

- Aiming for feature completeness vs. all major platform-specific sources (matplotlib, ggplot, etc)
- Will entertain pretty much any feasible application or feature request (or pull request 😍)
- SVG (based on D3) for publication-quality graphics of moderate-sized data
- WebGL (based on stack.gl, moving to regl) for 3D and large data sets

Modular

- For faster loading, use one of our smaller builds with chart types for a specific domain

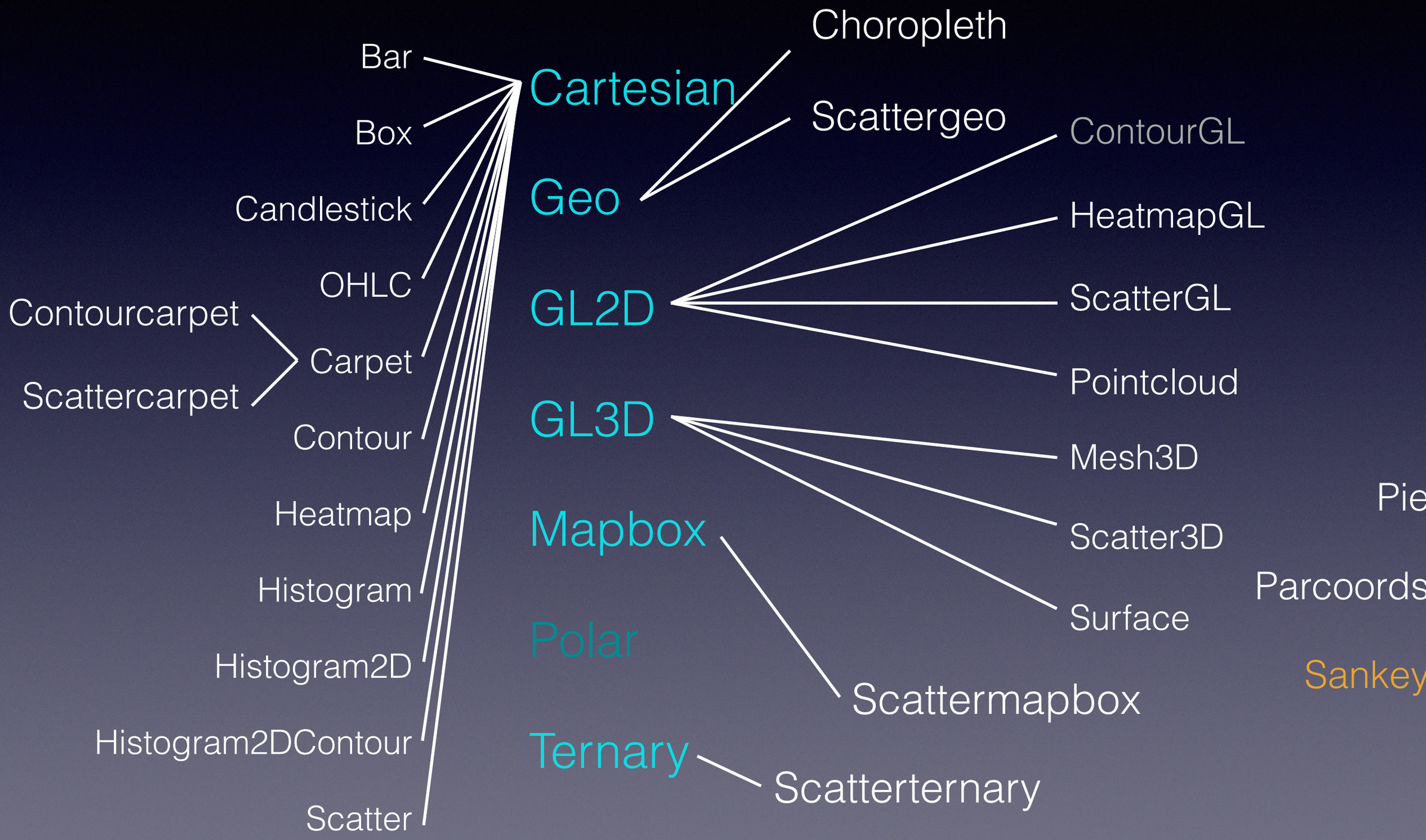
- Or even make your own build with the exact trace types you need

- plot.ly always uses the full build so your charts remain portable

Tested

- 2000 test cases of code & plot behavior: creation, modification, interactions
- 400 test images ensuring pixel-perfect consistency
- No code accepted without related tests (but if you make a PR we can help)

Plot and Trace Types



Features & Components

- Axis types: Linear, Log, Category, Date (15 calendar systems)
- Multiple subplots, insets, overlaid axes
- On/off-plot components: Annotations, Images, Shapes
- Interactive components: Legend, Modebar, Range selectors & sliders, Dropdown menus & sliders
- Snapshot (png, jpg, svg) and download
- Animations

Creating a plot

```
Plotly.newPlot(graphDiv, data, layout, config, frames)
```

graphDiv: HTML element or ID (deprecated)

data: Ordered array of trace objects. A trace represents one functional relationship - z(x,y), or (x, y, size), plus its presentation attributes

layout: All presentation info not related to a specific trace (axes, components, titles…)

config: Context-dependent flags for plot behavior

frames: Animation steps

How Plotly Makes a Plot

1. clean - convert old attributes to current API. Done once when new trace(s) or layout arrive. Mutates data & layout.

2. supplyDefaults - fill in defaults and sanitize values.
   `data-> gd._fullData, layout-> gd._fullLayout`

3. calc - trace-dependent reshaping into JSON representing objects to draw

4. plot - create the relevant DOM elements

5. style - set presentation aspects

Modifying a plot

- `Plotly.newPlot(gd, newData, newLayout)` or mutate **`gd.data`** and **`gd.layout`** and call **`Plotly.redraw(gd)`** (deprecated)
  - Still the right approach if you're changing so much it's hard to describe incrementally - but for smaller changes Plotly is often able to redraw faster using the methods below
  - **`gd.data`** and **`gd.layout`** are NOT guaranteed to be the same objects you passed in originally
- `Plotly.restyle, Plotly.relayout, Plotly.update`
  - Incremental updates to data, layout, or both
- `Plotly.addTraces, Plotly.deleteTraces, Plotly.moveTraces`
  - **`restyle / update`** can only edit the existing items in **`data`**
- `Plotly.extendTraces, Plotly.prependTraces`
  - For streaming applications, add new points to the end/beginning of existing traces

Events

Most user actions on a plot generate an event prefixed with **`plotly_`**:

- `click, doubleclick, clickannotation, buttonclicked`
- `selecting, selected, deselect`
- `restyle, relayout, update`
- `beforehover, hover, unhover`
- `sliderchange, sliderstart, sliderend`
- `beforeplot, afterplot, framework, redraw`
- `animating, animated, animationinterrrupted`
- `transitioning, transitioned`
- `beforeexport, afterexport`

```
gd.on('plotly_click', function(eventData) {
    console.log(eventData.points);
});
```