Browser-based geoprocessing with Turf.js

FOSS4G, Boston, 16 August 2017

Numa Gremling
The slides

http://geosysnet.com
Turf is a JavaScript library for spatial analysis.

... JavaScript!
Turf is GIS for web maps. It’s a fast, compact, and open-source JavaScript library that implements the most common geospatial operations: buffering, contouring, triangular irregular networks (TINs), and more. Turf speaks GeoJSON natively, easily connects to Leaflet, and is now available as a Mapbox.js plugin on our cloud platform. We’re also working to integrate Turf into our offline products and next-generation map rendering tools.

https://www.mapbox.com/blog/turf-gis-for-web-maps/
Origins

- Morgan Herlocker, 2013
  - [http://morganherlocker.com](http://morganherlocker.com)
  - [@morganherlocker](https://twitter.com/morganherlocker)

- Mapbox, since 2014
  - [https://www.mapbox.com](https://www.mapbox.com)
  - [@Mapbox](https://twitter.com/Mapbox)
Turf has opened a wide range of possibilities for apps running entirely in the browser. In this example, we’ve built a store locator that finds all Starbucks locations near a user’s current position. Drag the little green figure to move it around and mousewheel up and down to adjust the search radius:
https://www.mapbox.com/blog/coffee-with-turf/
GeoJSON in, GeoJSON out

- **GeoJSON**: the data format in Turf.js
- **All in one file:**
  - geometry
  - attributes
  - coordinate system
<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
</table>
| Point        | ```json
{ "type": "Point",
 "coordinates": [30, 10]
}
``` |
| LineString   | ```json
{ "type": "LineString",
 "coordinates": [
 [30, 10], [10, 30], [40, 40]
 ]
}
``` |
| Polygon      | ```json
{ "type": "Polygon",
 "coordinates": [
 [[30, 10], [40, 40], [20, 40], [10, 20], [30, 10]]
 ]
}
``` |
|             | ```json
{ "type": "Polygon",
 "coordinates": [
 [[35, 10], [45, 45], [15, 40], [10, 20], [35, 10]],
 [[20, 30], [35, 35], [30, 20], [20, 30]]
 ]
}
``` |
Where to get a GeoJSON?

- Write from scratch 😊
- [http://geojson.io/](http://geojson.io/)
- QGIS
Geoprocessing in the Web
Geoprocessing in the Web: WPS

- Typically: Web Processing Service
- OGC standard

- requires:
  - complex serverside infrastructure
  - requests
  - internet

http://www.opengeospatial.org/standards/wps
Geoprocessing in the Web: WPS

- user sends a request to a server
- server processes the request in a DB or a GIS
- server sends back the result
... a few minutes later...
Geoprocessing in the Web: WPS

- user sends a request to server
- server processes the request in a DB or a GIS
- server sends back the result
- (user falls asleep)
- (user notices a mistake, does it all over again)
- (user falls asleep again)
- (user notices another mistake?)
WPS - yes or no?

- WPS makes sense when:
  - analyses and calculations are complex
  - data is large

- **However**: WPS is often used to accomplish simple tasks

... until now!

TURF
Turf.js

| .github | Move github related files to .github | 2 months ago |
| packages | Add example testing directly to Turf core testing | 15 hours ago |
| scripts | referenced create-new-module in CONTRIBUTING.md: | 16 days ago |
| .eslintrcignore | Add example testing directly to Turf core testing | 15 hours ago |
| .eslintrc.js | JSDocs enforce @returns instead of @return | 4 months ago |
| .gitignore | Add boolean-shapely script (#802) | a month ago |
| .travis.yml | Add boolean-shapely as external module (#803) | a month ago |
| CHANGELOG.md | CHANGELOG updated with alternative merge() method (#559) | 6 months ago |
| CODE_OF_CONDUCT.md | add code of conduct (#495) | 10 months ago |
| CONTRIBUTING.md | referenced create-new-module in CONTRIBUTING.md: | 16 days ago |
| LICENSE | Clean package.json | 5 months ago |
| README.md | Clean up automated build | a month ago |
| SEE_ALSO.md | Create SEE_ALSO.md | 3 years ago |
| Isma.js | v4.5.2 | 23 days ago |
| package.json | New module `@turf/interpolate` (#832) | 12 days ago |
| tsconfig.json | Specify BBox as 4 digit array | 5 months ago |
| yarn.lock | New module `@turf/interpolate` (#832) | 12 days ago |
### Installation

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>npm</td>
<td>npm install @turf/turf</td>
</tr>
<tr>
<td>cdn</td>
<td><a href="https://npmcdn.com/@turf/turf/turf.min.js">https://npmcdn.com/@turf/turf/turf.min.js</a></td>
</tr>
<tr>
<td>download</td>
<td>turf.min.js</td>
</tr>
</tbody>
</table>
Modularity

Every turf function has been broken into its own separate module, so you can install what you need and nothing else.

05.06.2014, http://morganherlocker.com/

- Over 80 modules
Methods

center  centroid  distance  destination
envelope  midpoint  bezier  buffer
concave  convex  difference  intersect
simplify  union  combine  explode
flip  kinks  pointOnLine  random
sample  tin  inside  isolines
Mapbox.js?

- Official examples use Mapbox.js
- But: Turf.js can be used separately, even without a map
Mapping = optional

- Turf.js works without a map
- In most cases a map is used to display the results but that is entirely optional

- Turf.js can be used by itself, as a pure GIS library, without any visualization
Mapping Clients

- Leaflet
- OpenLayers
- Mapbox.js
- Mapbox GL JS
- Google Maps JavaScript API
- etc.

... as long as GeoJSON is supported Turf.js can be integrated
No API Keys

- works without API keys
- download, include, go!
Great docs!

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>TYPE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>Feature(&lt;\text{LineString})&gt;</td>
<td>input LineString</td>
</tr>
<tr>
<td>resolution</td>
<td>number? (default 10000)</td>
<td>time in milliseconds between points</td>
</tr>
<tr>
<td>sharpness</td>
<td>number? (default 0.85)</td>
<td>a measure of how curvy the path should be between splines</td>
</tr>
</tbody>
</table>
Point in Polygon

Point 18 is in Nevada
Point 7 is in Ohio
Point 11 is in Colorado
Point 12 is in Colorado
Point 13 is in Colorado
Point 14 is in Colorado
Point 15 is in Colorado
Point 17 is in Idaho
Point 6 is in Indiana
Point 4 is in Iowa
Point 2 is in Kansas
Point 8 is in Mississippi
Point 3 is in Missouri
Point 5 is in Nebraska
Point 10 is in New Mexico
Point 9 is in Oklahoma
Point 1 is in Oregon
Point 16 is in Wyoming
Point in Polygon

Point 18 is in Nevada
Point 7 is in Ohio
Point 11 is in Colorado
Point 12 is in Colorado
Point 13 is in Colorado
Point 14 is in Colorado
Point 15 is in Colorado
Point 17 is in Idaho
Point 6 is in Indiana
Point 4 is in Iowa
Point 2 is in Kansas
Point 8 is in Mississippi
Point 3 is in Missouri
Point 5 is in Nebraska
Point 10 is in New Mexico
Point 9 is in Oklahoma
Point 1 is in Oregon
Point 16 is in Wyoming
Lines to Points
Lines to Points
Digitizing
Digitizing
Digitizing
Digitizing
Digitizing
Digitizing
```javascript
var spreeBezier = turf.bezier(spree.features[0]);
```
geosysnet.com/demos/dashboard
Learn Turf!

https://learn.gis-trainer.com/
console.log('Thank you! :-)!');

numa.gremling@geosysnet.de

http://geosysnet.com
@geoSYSnet

https://gis-trainer.com
@gis_trainer
gis_trainer

https://learn.gis-trainer.com
gistrainer