

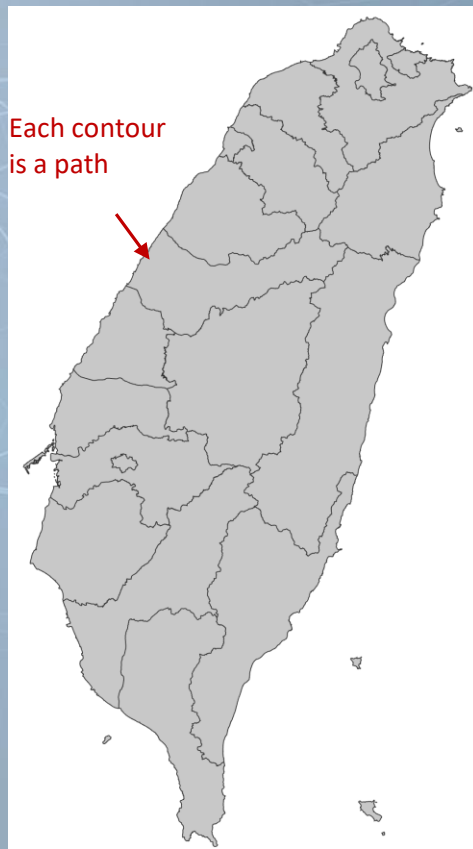


# Map

## Data Visualization

# Three Key Concept of Map in D3

- GeoJSON
  - A json-based format for specifying geographic data
  - D3 creates map based on GeoJSON data
- Map projections
  - Functions that convert from latitude/longitude coordinate to x and y coordinates
- Geographic path generators – **d3.geoPath()**
  - Functions that convert GeoJSON shapes into SVG “paths”
  - Similar to shape generator e.g. d3.line(), d3.area(), etc.
  - Note: d3.geoPath() only recognize **WGS84** geodetic system. Make sure the file you import uses WGS84



# GeoJSON

- A JSON-based format for specifying geographic data
- A segment of GeoJSON data for Taipei City in “taiwan.json”
  - You can find “taiwan.json” in Ex06-01 folder

Properties: name, id, and other attributes of the region

```
{ "type": "Feature", "properties": { "COUNTYID": "63", "NAME_1984": "臺北市", "NAME_1984_ALIAS": "台北市", "NAME_2010": "臺北市", "NAME_2010_ALIAS": "台北市", "NAME_2014": "臺北市", "NAME_2014_ALIAS": "台北市", "ISO3166": "TPE", "SEGIS_COUNTY_ID": "63000", "AREA_ID": "A", "_id": "18744641", "geometry": { "type": "Polygon", "coordinates": [[[ [121.570980505, 25.197168115], [121.570972325, 25.197035112], [121.570933739, 25.196892124], [121.570916396, 25.196824628], [121.570896707, 25.196769777], [121.570873123, 25.196706945], [121.570838015, 25.196631625], [121.570810569, 25.196563093], [121.570794274, 25.196480668], [121.570809417, 25.196392146], [121.570871837, 25.196258319], [121.57093985, 25.196163419], [121.571050807, 25.196073786], [121.571103112, 25.196027289], [121.571178172, 25.19598511], [121.571279499, 25.195927707], [121.571360862, 25.19588545], [121.571414521, 25.195845818], [121.571479026, 25.195770484], [121.571527208, 25.195699925], [121.571567938, 25.195637512], [121.571599922, 25.195580893], [121.571661955, 25.195509035], [121.571694973, 25.195436367], [121.5717422, 25.195387625], [121.571887893, 25.195346915], [121.572022694, 25.195338467], [121.572156089, 25.195319697], [121.572267904, 25.195290863], [121.572404217, 25.195211251], [121.572506625, 25.195141223], [121.572623032, 25.195080196], [121.572711925, 25.195035561], [121.572904134, 25.194985127], [121.573071745, 25.194978591], [121.573180199, 25.194979625], [121.573264975, 25.19500044], [121.573344693, 25.195020172], [121.57342179, 25.195033045], [121.573534334, 25.195055829], [121.57359519, 25.195079218], [121.573724591, 25.195135087], [121.573824513, 25.195156862], [121.573918002, 25.195169545], [121.574003551, 25.195155933], [121.574094063, 25.195136515], [121.574160911, 25.195138034], [121.574246998, 25.195162264], [121.574470178, 25.195251468], [121.574590627, 25.195298253], [121.574660911, 25.195338467], [121.574746998, 25.195369992], [121.574824513, 25.19540044], [121.574904063, 25.195436367], [121.574978591, 25.195469992], [121.57505829, 25.195509035], [121.575135087, 25.195549992], [121.575211251, 25.195580893], [121.575287893, 25.195611751], [121.575364975, 25.195642628], [121.57544179, 25.195673786], [121.575518002, 25.19570484], [121.575594063, 25.195735922], [121.575670178, 25.195766977], [121.575746199, 25.195798015], [121.575822172, 25.195829052], [121.575898199, 25.195860089], [121.575974217, 25.195891124], [121.576050238, 25.195922161], [121.576126259, 25.195953198], [121.576202279, 25.195984235], [121.576278301, 25.196015272], [121.576354322, 25.196046309], [121.576430343, 25.196077346], [121.576506364, 25.196108383], [121.576582385, 25.196139419], [121.576658406, 25.196170456], [121.576734427, 25.196201493], [121.576810448, 25.19623253], [121.576886469, 25.196263567], [121.57696249, 25.196294604], [121.577038511, 25.196325641], [121.577114532, 25.196356678], [121.577190553, 25.196387715], [121.577266574, 25.196418752], [121.577342595, 25.196449789], [121.577418616, 25.196480826], [121.577494637, 25.196511863], [121.577570658, 25.196542899], [121.577646679, 25.196573936], [121.5777227, 25.196604973], [121.577798721, 25.19663601], [121.577874742, 25.196667047], [121.577950763, 25.196698084], [121.578026784, 25.196729121], [121.578102805, 25.196760158], [121.578178826, 25.196791195], [121.578254847, 25.196822232], [121.578330868, 25.196853269], [121.578406889, 25.196884306], [121.57848291, 25.196915343], [121.578558931, 25.19694638], [121.578634952, 25.196977417], [121.578710973, 25.197008454], [121.578786994, 25.197039491], [121.578863015, 25.197070528], [121.578939036, 25.197101565], [121.579015057, 25.197132602], [121.579091078, 25.197163639], [121.579167099, 25.197194676], [121.57924312, 25.197225713], [121.579319141, 25.19725675], [121.579395162, 25.197287787], [121.579471183, 25.197318824], [121.579547204, 25.197349861], [121.579623225, 25.197380898], [121.579699246, 25.197411935], [121.579775267, 25.197442972], [121.579851288, 25.197474009], [121.579927309, 25.197505046], [121.58000333, 25.197536083], [121.580079351, 25.19756712], [121.580155372, 25.197598157], [121.580231393, 25.197629194], [121.580307414, 25.197660231], [121.580383435, 25.197691268], [121.580459456, 25.197722305], [121.580535477, 25.197753342], [121.580611498, 25.197784379], [121.580687519, 25.197815416], [121.58076354, 25.197846453], [121.580839561, 25.19787749], [121.580915582, 25.197908527], [121.580991603, 25.197939564], [121.581067624, 25.197970601], [121.581143645, 25.198001638], [121.581219666, 25.198032675], [121.581295687, 25.198063712], [121.581371708, 25.198094749], [121.581447729, 25.198125786], [121.58152375, 25.198156823], [121.5816, 25.19818786], [121.581676021, 25.198218897], [121.581752042, 25.198249934], [121.581828063, 25.198280971], [121.581904084, 25.198312008], [121.581980105, 25.198343045], [121.582056126, 25.198374082], [121.582132147, 25.198405119], [121.582208168, 25.198436156], [121.582284189, 25.198467193], [121.58236021, 25.19849823], [121.582436231, 25.198529267], [121.582512252, 25.198560304], [121.582588273, 25.198591341], [121.582664294, 25.198622378], [121.582740315, 25.198653415], [121.582816336, 25.198684452], [121.582892357, 25.198715489], [121.582968378, 25.198746526], [121.583044399, 25.198777563], [121.58312042, 25.1988086], [121.583196441, 25.198839637], [121.583272462, 25.198870674], [121.583348483, 25.198901711], [121.583424504, 25.198932748], [121.583500525, 25.198963785], [121.583576546, 25.198994822], [121.583652567, 25.199025859], [121.583728588, 25.199056896], [121.583804609, 25.199087933], [121.58388063, 25.19911897], [121.583956651, 25.199150007], [121.584032672, 25.199181044], [121.584108693, 25.199212081], [121.584184714, 25.199243118], [121.584260735, 25.199274155], [121.584336756, 25.199305192], [121.584412777, 25.199336229], [121.584488798, 25.199367266], [121.584564819, 25.199398303], [121.58464084, 25.19942934], [121.584716861, 25.199460377], [121.584792882, 25.199491414], [121.584868903, 25.199522451], [121.584944924, 25.199553488], [121.585020945, 25.199584525], [121.585096966, 25.199615562], [121.585172987, 25.199646599], [121.585249008, 25.199677636], [121.585325029, 25.199708673], [121.58540105, 25.19973971], [121.585477071, 25.199770747], [121.585553092, 25.199801784], [121.585629113, 25.199832821], [121.585705134, 25.199863858], [121.585781155, 25.199894895], [121.585857176, 25.199925932], [121.585933197, 25.199956969], [121.586009218, 25.199988006], [121.586085239, 25.200019043], [121.58616126, 25.20005008], [121.586237281, 25.200081117], [121.586313302, 25.200112154], [121.586389323, 25.200143191], [121.586465344, 25.200174228], [121.586541365, 25.200205265], [121.586617386, 25.200236302], [121.586693407, 25.200267339], [121.586769428, 25.200298376], [121.586845449, 25.200329413], [121.58692147, 25.20036045], [121.586997491, 25.200391487], [121.587073512, 25.200422524], [121.587149533, 25.200453561], [121.587225554, 25.200484598], [121.587301575, 25.200515635], [121.587377596, 25.200546672], [121.587453617, 25.200577709], [121.587529638, 25.200608746], [121.587605659, 25.200639783], [121.58768168, 25.20067082], [121.587757701, 25.200701857], [121.587833722, 25.200732894], [121.587909743, 25.200763931], [121.587985764, 25.200794968], [121.588061785, 25.200826005], [121.588137806, 25.200857042], [121.588213827, 25.200888079], [121.588289848, 25.200919116], [121.588365869, 25.200950153], [121.58844189, 25.20098119], [121.588517911, 25.201012227], [121.588593932, 25.201043264], [121.588669953, 25.201074301], [121.588745974, 25.201105338], [121.588821995, 25.201136375], [121.588898016, 25.201167412], [121.588974037, 25.201198449], [121.589050058, 25.201229486], [121.589126079, 25.201260523], [121.5892021, 25.20129156], [121.589278121, 25.201322597], [121.589354142, 25.201353634], [121.589430163, 25.201384671], [121.589506184, 25.201415708], [121.589582205, 25.201446745], [121.589658226, 25.201477782], [121.589734247, 25.201508819], [121.589810268, 25.201539856], [121.589886289, 25.201570893], [121.58996231, 25.20160193], [121.590038331, 25.201632967], [121.590114352, 25.201664004], [121.590190373, 25.201695041], [121.590266394, 25.201726078], [121.590342415, 25.201757115], [121.590418436, 25.201788152], [121.590494457, 25.201819189], [121.590570478, 25.201850226], [121.590646499, 25.201881263], [121.59072252, 25.2019123], [121.590798541, 25.201943337], [121.590874562, 25.201974374], [121.590950583, 25.202005411], [121.591026604, 25.202036448], [121.591102625, 25.202067485], [121.591178646, 25.202098522], [121.591254667, 25.202129559], [121.591330688, 25.202160596], [121.591406709, 25.202191633], [121.59148273, 25.20222267], [121.591558751, 25.202253707], [121.591634772, 25.202284744], [121.591710793, 25.202315781], [121.591786814, 25.202346818], [121.591862835, 25.202377855], [121.591938856, 25.202408892], [121.592014877, 25.202439929], [121.592090898, 25.202470966], [121.592166919, 25.202502003], [121.59224294, 25.20253304], [121.592318961, 25.202564077], [121.592394982, 25.202595114], [121.592471003, 25.202626151], [121.592547024, 25.202657188], [121.592623045, 25.202688225], [121.592699066, 25.202719262], [121.592775087, 25.202750299], [121.592851108, 25.202781336], [121.592927129, 25.202812373], [121.59300315, 25.20284341], [121.593079171, 25.202874447], [121.593155192, 25.202905484], [121.593231213, 25.202936521], [121.593307234, 25.202967558], [121.593383255, 25.202998595], [121.593459276, 25.203029632], [121.593535297, 25.203060669], [121.593611318, 25.203091706], [121.593687339, 25.203122743], [121.59376336, 25.20315378], [121.593839381, 25.203184817], [121.593915402, 25.203215854], [121.593991423, 25.203246891], [121.594067444, 25.203277928], [121.594143465, 25.203308965], [121.594219486, 25.20334], [121.594295507, 25.203371037], [121.594371528, 25.203402074], [121.594447549, 25.203433111], [121.59452357, 25.203464148], [121.594599591, 25.203495185], [121.594675612, 25.203526222], [121.594751633, 25.203557259], [121.594827654, 25.203588296], [121.594903675, 25.203619333], [121.594979696, 25.20365037], [121.595055717, 25.203681407], [121.595131738, 25.203712444], [121.595207759, 25.203743481], [121.59528378, 25.203774518], [121.595359801, 25.203805555], [121.595435822, 25.203836592], [121.595511843, 25.203867629], [121.595587864, 25.203898666], [121.595663885, 25.203929703], [121.595739906, 25.20396074], [121.595815927, 25.203991777], [121.595891948, 25.204022814], [121.595967969, 25.204053851], [121.59604399, 25.204084888], [121.596119911, 25.204115925], [121.596195932, 25.204146962], [121.596271953, 25.204178], [121.596347974, 25.204209037], [121.596423995, 25.204240074], [121.5965, 25.204271111], [121.596576021, 25.204302148], [121.596652042, 25.204333185], [121.596728063, 25.204364222], [121.596804084, 25.204395259], [121.596880105, 25.204426296], [121.596956126, 25.204457333], [121.597032147, 25.20448837], [121.597108168, 25.204519407], [121.597184189, 25.204550444], [121.59726021, 25.204581481], [121.597336231, 25.204612518], [121.597412252, 25.204643555], [121.597488273, 25.204674592], [121.597564294, 25.204705629], [121.597640315, 25.204736666], [121.597716336, 25.204767703], [121.597792357, 25.20479874], [121.597868378, 25.204829777], [121.597944399, 25.204860814], [121.59802042, 25.204891851], [121.598096441, 25.204922888], [121.598172462, 25.204953925], [121.598248483, 25.204984962], [121.598324504, 25.205016], [121.598400525, 25.205047037], [121.598476546, 25.205078074], [121.598552567, 25.205109111], [121.598628588, 25.205140148], [121.598704609, 25.205171185], [121.59878063, 25.205202222], [121.598856651, 25.205233259], [121.598932672, 25.205264296], [121.599008693, 25.205295333], [121.599084714, 25.20532637], [121.599160735, 25.205357407], [121.599236756, 25.205388444], [121.599312777, 25.205419481], [121.599388798, 25.205450518], [121.599464819, 25.205481555], [121.59954084, 25.205512592], [121.599616861, 25.205543629], [121.599692882, 25.205574666], [121.599768903, 25.205605703], [121.599844924, 25.20563674], [121.599920945, 25.205667777], [121.600000, 25.205698814], [121.600076021, 25.205729851], [121.600152042, 25.205760888], [121.600228063, 25.205791925], [121.600304084, 25.205822962], [121.600380105, 25.205854], [121.600456126, 25.205885037], [121.600532147, 25.205916074], [121.600608168, 25.205947111], [121.600684189, 25.205978148], [121.60076021, 25.206009185], [121.600836231, 25.206040222], [121.600912252, 25.206071259], [121.600988273, 25.206102296], [121.601064294, 25.206133333], [121.601140315, 25.20616437], [121.601216336, 25.206195407], [121.601292357, 25.206226444], [121.601368378, 25.206257481], [121.601444399, 25.206288518], [121.60152042, 25.206319555], [121.601596441, 25.206350592], [121.601672462, 25.206381629], [121.601748483, 25.206412666], [121.601824504, 25.206443703], [121.601900525, 25.20647474], [121.601976546, 25.206505777], [121.602052567, 25.206536814], [121.602128588, 25.206567851], [121.602204609, 25.206598888], [121.60228063, 25.206629925], [121.602356651, 25.206660962], [121.602432672, 25.206692], [121.602508693, 25.206723037], [121.602584714, 25.206754074], [121.602660
```

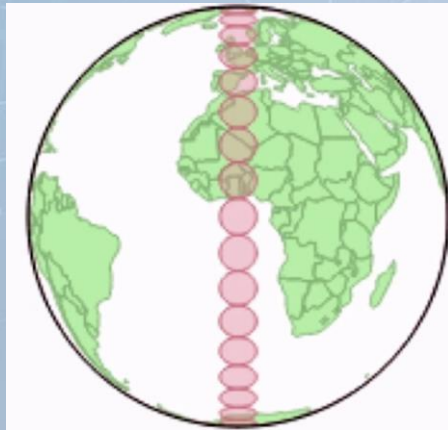
# TopoJSON

- Comparing with GeoJSON, TopoJSON is smaller
- To load and use topoJSON file to draw map
  - You need <https://d3js.org/topojson.v3.min.js> to convert topojson format to geoJson, then send to data to d3.geoPath
  - We do not use or introduce topoJSON in details in this lecture

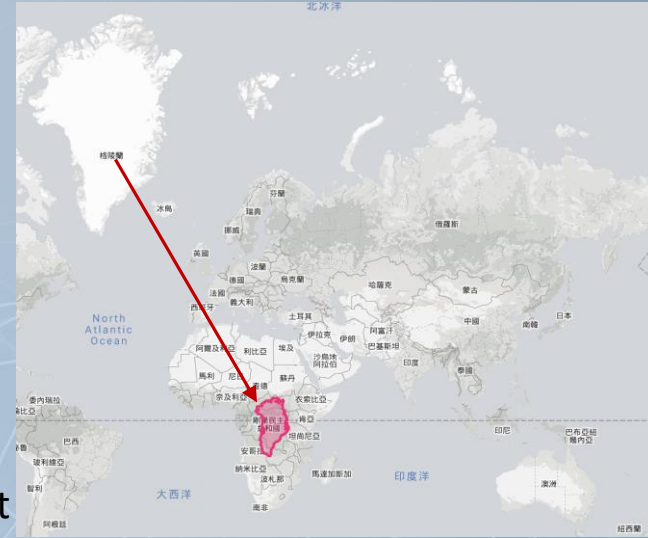
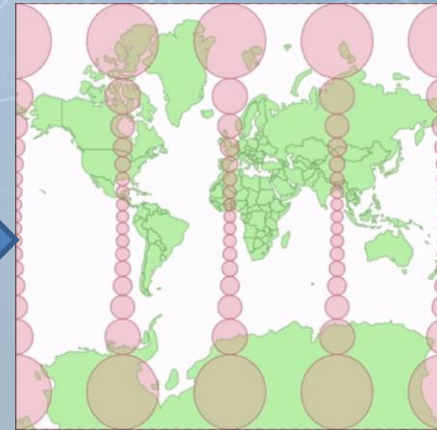


# Map Projection

- Function that convert from latitude/longitude coordinates to x and y coordinates
- Why?
  - “The true size of” website
    - <https://thetruesize.com/>
  - Our earth is a sphere. If we want to map it to a rectangle, the distortion must exist



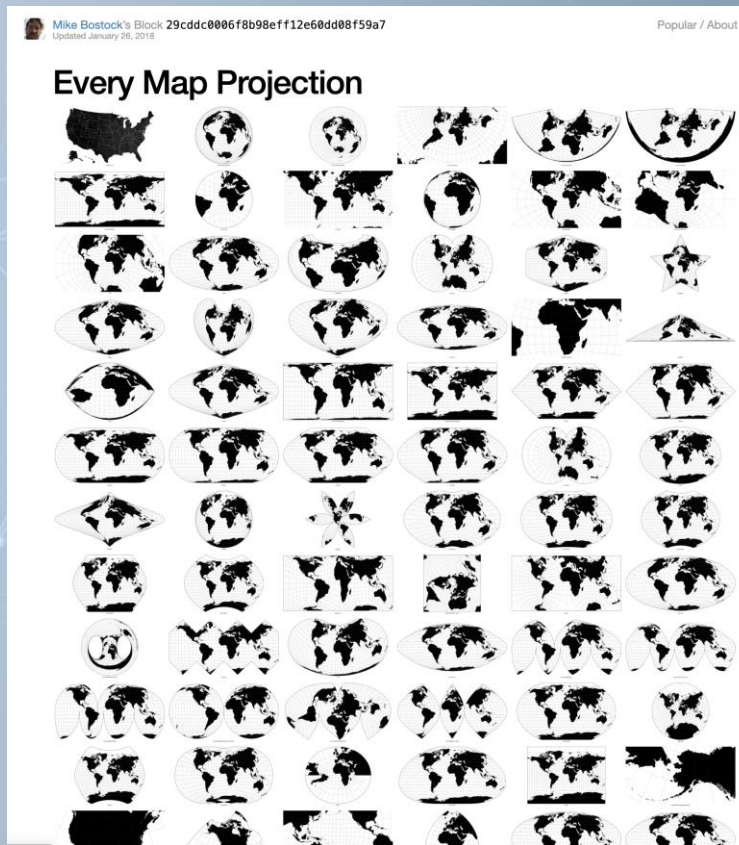
projection



# Map Projection

- D3 supports various map projection
  - <https://github.com/d3/d3-geo-projection>
  - Every projection will distort shape, area, distance and/or direction. Every projection also try to keep some attributes have as less distortion as possible
  - Think what properties you do not want to be distorted in your visualization. Then, choose a projection which has as less as possible distortion on these properties

<https://bl.ocks.org/mbostock/29cddc0006f8b98eff12e60dd08f59a7>

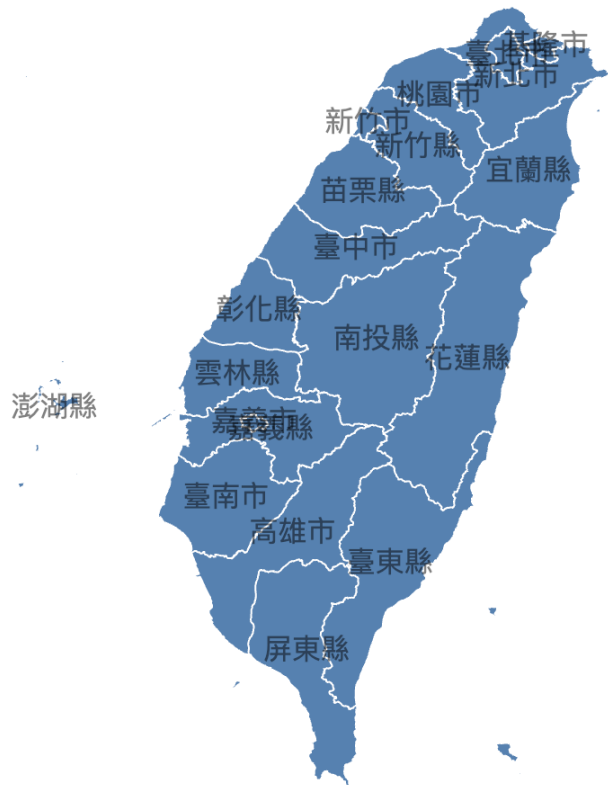


## Ex08-01

- Load "taiwan.json" and draw city/county with their name

金門縣

連江縣



# Ex06-01

- In this example, we use `d3.geoEquirectangular()` as our projection function
  - `.fitExtent(extent, GeoJSON)`
    - The specified region will be scaled to fill the extent on the screen
  - `[[0,0], [width, height]]`:
    - map the top-left latitude/longitude in the GeoJSON to `[0,0]` and bottom-right latitude/longitude in the GeoJSON to `[width, height]`

Load taiwan.json. If the loading is done, run `drawTaiwan()`

`d3.json("taiwan.json").then(drawTaiwan);`

```
function drawTaiwan(taiwan) {
  var width = 1000;
  var height = 800;

  var projection = d3.geoEquirectangular()
    .fitExtent([[0,0], [width, height]], taiwan);

  var geoGenerator = d3.geoPath()
    .projection(projection);

  var paths = d3.select('svg')
    .selectAll('path')
    .data(taiwan.features)
    .enter()
    .append('path')
    .attr('stroke', 'white')
    .attr('fill', 'steelblue')
    .attr('d', geoGenerator);

  var texts = d3.select('svg')
    .selectAll('text')
    .data(taiwan.features)
    .enter()
    .append('text')
    .attr('text-anchor', 'middle')
    .attr('alignment-baseline', 'middle')
    .attr('opacity', 0.5)
    .text(function(d) {
      return d.properties.NAME_2014;
    })
    .attr('transform', function(d) {
      var center = geoGenerator.centroid(d);
      return 'translate (' + center + ')';
    });
}
```

Content of taiwan.json



# Ex06-01

- In this example, we use `d3.geoEquirectangular()` as our projection function
  - `.fitExtent(extent, GeoJSON)`
    - The specified region will be scaled to fill the extent on the screen
  - `[[0,0], [width, height]]`:
    - map the top-left latitude/longitude in the GeoJSON to `[0,0]` and bottom-right latitude/longitude in the GeoJSON to `[width, height]`
- Use “`d3.geoPath()`” to create the generator
  - Remember to set the projection function to it by `.projection()`

Load taiwan.json. If the loading is done, run `drawTaiwan()`

`d3.json("taiwan.json").then(drawTaiwan);`

```
function drawTaiwan(taiwan) {  
  var width = 1000;  
  var height = 800;  
  
  var projection = d3.geoEquirectangular()  
    .fitExtent([[0,0], [width, height]], taiwan);  
  
  var geoGenerator = d3.geoPath()  
    .projection(projection);  
  
  var paths = d3.select('svg')  
    .selectAll('path')  
    .data(taiwan.features)  
    .enter()  
    .append('path')  
    .attr('stroke', 'white')  
    .attr('fill', 'steelblue')  
    .attr('d', geoGenerator);  
  
  var texts = d3.select('svg')  
    .selectAll('text')  
    .data(taiwan.features)  
    .enter()  
    .append('text')  
    .attr('text-anchor', 'middle')  
    .attr('alignment-baseline', 'middle')  
    .attr('opacity', 0.5)  
    .text(function(d) {  
      return d.properties.NAME_2014;  
    })  
    .attr('transform', function(d) {  
      var center = geoGenerator.centroid(d);  
      return 'translate (' + center + ')';  
    });  
}
```

Content of taiwan.json

# Ex06-01

- Use taiwan.features as the data to draw the paths

Each element in the data array (Taiwan.features) will be sent to “geoGenerator” to generator the path descriptor

```
d3.json("taiwan.json").then(drawTaiwan);

function drawTaiwan(taiwan) {
  var width = 1000;
  var height = 800;

  var projection = d3.geoEquirectangular()
    .fitExtent([[0,0], [width, height]], taiwan);

  var geoGenerator = d3.geoPath()
    .projection(projection);

  var paths = d3.select('svg')
    .selectAll('path')
    .data(taiwan.features)
    .enter()
    .append('path')
    .attr('stroke', 'white')
    .attr('fill', 'steelblue')
    .attr('d', geoGenerator);

  var texts = d3.select('svg')
    .selectAll('text')
    .data(taiwan.features)
    .enter()
    .append('text')
    .attr('text-anchor', 'middle')
    .attr('alignment-baseline', 'middle')
    .attr('opacity', 0.5)
    .text(function(d) {
      return d.properties.NAME_2014;
    })
    .attr('transform', function(d) {
      var center = geoGenerator.centroid(d);
      return 'translate (' + center + ')';
    });
}
```

22 cities/counties (so, 22 paths)

taiwan.features

Data to generate a path

## Ex06-01

Use Taiwan.features as  
the data to draw the  
paths

geoGenerator looks  
for data in the  
attribute with the  
name "geometry"  
to generate the path

```
▼ Array(22)
  ▼ geometry:
    ▼ coordinates: Array(183)
      ► [0 ... 99]
      ► [100 ... 182]
        length: 183
      ► __proto__: Array(0)
    type: "MultiPolygon"
    ► __proto__: Object
  ▼ properties:
    AREA_ID: "Z"
    COUNTYID: "9007"
    IS03166: "LJF"
    NAME_1984: "連江縣"
    NAME_1984_ALIAS: ""
    NAME_2010: "連江縣"
    NAME_2010_ALIAS: ""
    NAME_2014: "連江縣"
    NAME_2014_ALIAS: ""
    SEGIS_COUNTY_ID: "9007"
    _id: 18744625
    ► __proto__: Object
  type: "Feature"
  ► __proto__: Object
  ► 1: {type: "Feature", properties: {...}, geometry: {...}}
  ► 2: {type: "Feature", properties: {...}, geometry: {...}}
  ► 3: {type: "Feature", properties: {...}, geometry: {...}}
  ► 4: {type: "Feature", properties: {...}, geometry: {...}}
  ► 5: {type: "Feature", properties: {...}, geometry: {...}}
  ► 6: {type: "Feature", properties: {...}, geometry: {...}}
  ► 7: {type: "Feature", properties: {...}, geometry: {...}}
  ► 8: {type: "Feature", properties: {...}, geometry: {...}}
  ► 9: {type: "Feature", properties: {...}, geometry: {...}}
  ► 10: {type: "Feature", properties: {...}, geometry: {...}}
  ► 11: {type: "Feature", properties: {...}, geometry: {...}}
  ► 12: {type: "Feature", properties: {...}, geometry: {...}}
  ► 13: {type: "Feature", properties: {...}, geometry: {...}}
  ► 14: {type: "Feature", properties: {...}, geometry: {...}}
  ► 15: {type: "Feature", properties: {...}, geometry: {...}}
  ► 16: {type: "Feature", properties: {...}, geometry: {...}}
  ► 17: {type: "Feature", properties: {...}, geometry: {...}}
  ► 18: {type: "Feature", properties: {...}, geometry: {...}}
  ► 19: {type: "Feature", properties: {...}, geometry: {...}}
  ► 20: {type: "Feature", properties: {...}, geometry: {...}}
  ► 21: {type: "Feature", properties: {...}, geometry: {...}}
  length: 22
  ► __proto__: Array(0)
```

d3.json("taiwan.json").then(drawTaiwan);

```
function drawTaiwan(taiwan) {
  var width = 1000;
  var height = 800;

  var projection = d3.geoEquirectangular()
    .fitExtent([[0,0], [width, height]], taiwan);

  var geoGenerator = d3.geoPath()
    .projection(projection);

  var paths = d3.select('svg')
    .selectAll('path')
    .data(taiwan.features)
    .enter()
    .append('path')
    .attr('stroke', 'white')
    .attr('fill', 'steelblue')
    .attr('d', geoGenerator);

  var texts = d3.select('svg')
    .selectAll('text')
    .data(taiwan.features)
    .enter()
    .append('text')
    .attr('text-anchor', 'middle')
    .attr('alignment-baseline', 'middle')
    .attr('opacity', 0.5)
    .text(function(d) {
      return d.properties.NAME_2014;
    })
    .attr('transform', function(d) {
      var center = geoGenerator.centroid(d);
      return 'translate(' + center + ')';
    });
}
```

Each element in the data array  
(Taiwan.features) will be sent  
to "geoGenerator" to  
generate the path descriptor

# Ex06-01

Put the cities/counties name on the map

```
▼ Array(22) 1
▼ 0:
  ▼ geometry:
    ▼ coordinates: Array(183)
      ► [0 ... 99]
      ► [100 ... 182]
        length: 183
      ► __proto__: Array(0)
    type: "MultiPolygon"
    ► __proto__: Object
  ▼ properties:
    AREA_ID: "Z"
    COUNTYID: "9007"
    IS03166: "LJF"
    NAME_1984: "連江縣"
    NAME_1984_ALIAS: ""
    NAME_2010: "連江縣"
    NAME_2010_ALIAS: ""
    NAME_2014: "連江縣"
    NAME_2014_ALIAS: ""
    SEGIS_COUNTY_ID: "9007"
    _id: 18744625
    ► __proto__: Object
  type: "Feature"
  ► __proto__: Object
► 1: {type: "Feature", properties: {...}, geometry: {...}}
► 2: {type: "Feature", properties: {...}, geometry: {...}}
► 3: {type: "Feature", properties: {...}, geometry: {...}}
► 4: {type: "Feature", properties: {...}, geometry: {...}}
► 5: {type: "Feature", properties: {...}, geometry: {...}}
► 6: {type: "Feature", properties: {...}, geometry: {...}}
► 7: {type: "Feature", properties: {...}, geometry: {...}}
► 8: {type: "Feature", properties: {...}, geometry: {...}}
► 9: {type: "Feature", properties: {...}, geometry: {...}}
► 10: {type: "Feature", properties: {...}, geometry: {...}}
► 11: {type: "Feature", properties: {...}, geometry: {...}}
► 12: {type: "Feature", properties: {...}, geometry: {...}}
► 13: {type: "Feature", properties: {...}, geometry: {...}}
► 14: {type: "Feature", properties: {...}, geometry: {...}}
► 15: {type: "Feature", properties: {...}, geometry: {...}}
► 16: {type: "Feature", properties: {...}, geometry: {...}}
► 17: {type: "Feature", properties: {...}, geometry: {...}}
► 18: {type: "Feature", properties: {...}, geometry: {...}}
► 19: {type: "Feature", properties: {...}, geometry: {...}}
► 20: {type: "Feature", properties: {...}, geometry: {...}}
► 21: {type: "Feature", properties: {...}, geometry: {...}}
  length: 22
  ► __proto__: Array(0)
```

taiwan.features

d3.json("taiwan.json").then(drawTaiwan);

```
function drawTaiwan(taiwan) {
  var width = 1000;
  var height = 800;
```

```
  var projection = d3.geoEquirectangular()
    .fitExtent([[0,0], [width, height]], taiwan);
```

```
  var geoGenerator = d3.geoPath()
    .projection(projection);
```

```
  var paths = d3.select('svg')
    .selectAll('path')
    .data(taiwan.features)
    .enter()
    .append('path')
    .attr('fill', 'steelblue')
    .attr('d', geoGenerator);
```

Calculate centroid of the cities/counties to place the name

```
  var texts = d3.select('svg')
    .selectAll('text')
    .data(taiwan.features)
    .enter()
    .append('text')
    .attr('text-anchor', 'middle')
    .attr('alignment-baseline', 'middle')
    .attr('opacity', 0.5)
    .text(function(d) {
      return d.properties.NAME_2014;
    })
    .attr('transform', function(d) {
      var center = geoGenerator.centroid(d);
      return 'translate (' + center + ')';
    });
```



## Ex06-01

- Convert longitude and latitudes to x-y and draw a circle
- Send longitude/latitudes to the projection function. It returns an array with x and y.
  - 120.9575, 23.47: Yushan (玉山)

```
d3.select('svg').append('circle')  
  .attr('cx', projection( [120.9575, 23.47 ])[0] )  
  .attr('cy', projection( [120.9575, 23.47 ])[1] )  
  .attr('fill', 'red')  
  .attr('r', 5);
```

