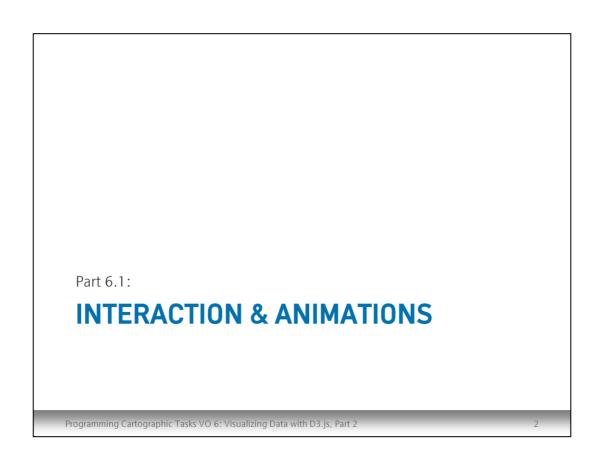


Programming Cartographic Tasks 2015

Lecture 6: Visualizing Data with D3.js, Part 2

Florian Ledermann





Adding Interaction: Elements

```
<svg class="map" id="map1" viewBox="0 0 800 400">
<g class="geometry"></g>
</svg>
<!-- add some buttons to trigger interaction -->
<button id="populationButton">Population</button>
<button id="areaButton">Area</button>
</script src="d3.js"></script>

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
```

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2

// old version
d3.csv('places-AT-cleaned.csv', draw_map);

```
// new version
d3.csv('places-AT-cleaned.csv', function(data) {
    d3.select('#populationButton').on('click', function() {
        // draw population map
        console.log("population");
    });

d3.select('#areaButton').on('click', function() {
        // draw area map
        console.log("area");
    });

});

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js. Part 2
```

... so we want to draw two different kinds of maps how can we set this up?

Let's look at the draw_map function from last time...

```
function draw_map(data) {
  var selection = d3.select('.map')
    .selectAll('circle')
    .data(data);

selection.enter()
    .append('circle')
    .attr({
        'fill-opacity': 0.5,
        cx: function(d) {return projection([d.lon, d.lat])[0]},
        cy: function(d) {return projection([d.lon, d.lat])[1]},
        r: function(d) {
            return Math.sqrt(parseFloat(d.population)/500);
        }
    });
}
```

This is the code from last time

What line of code makes it draw a population map, specifically?

```
function draw_map(data, radiusFunction) {
  var selection = d3.select('.map')
    .selectAll('circle')
    .data(data);

selection.enter()
    .append('circle')
    .attr({
        'fill-opacity': 0.5,
        cx: function(d) {return projection([d.lon, d.lat])[0]},
        cy: function(d) {return projection([d.lon, d.lat])[1]},
        r: radiusFunction
    });
}
```

```
var populationRadius = function(d) {
    return Math.sqrt(parseFloat(d.population)/500);
};
var areaRadius = function(d) {
    return Math.sqrt(parseFloat(d.area)/10);
};
d3.csv('places-AT-cleaned.csv', function(data) {
    d3.select('#populationButton').on('click', function() {
        draw_map(data, populationRadius);
    });
d3.select('#areaButton').on('click', function() {
        draw_map(data, areaRadius);
    });
});
Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
8
```

If we run this, it works only the first time we click on a button...

This is our current version of the code...

D3 Selections Revisited

var selection = d3.selectAll('circle');

Selection: Array of (groups of) elements

selection.data(arrayOfData);

- selection.data() joins elements to an array of data
 - matchmaking by Array position (index) or key function
 - placeholders are created for non-existing elements
- After join: 3 parts
 - selection: updated elements
 - selection.enter(): placeholders for missing elements
 - selection.exit(): elements without matching data items

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2

D3 Selections Revisited

```
var selection = d3.select('.map')
    .selectAll('circle')
    .data(data);

selection.enter()
    .append('circle')
    .attr({
        'fill-opacity': 0.5,
        cx: function(d) {return projection([d.lon, d.lat])[0]},
        cy: function(d) {return projection([d.lon, d.lat])[1]},
        r: radiusFunction
});
```

The enter() part is never executed for existing elements!

D3 Selections Revisited

```
var selection = d3.select('.map')
    .selectAll('circle')
    .data(data);
// what to do with NEW elements
selection.enter()
    .append('circle')
    .attr({
        'fill-opacity': 0.5,
        cx: function(d) {return projection([d.lon, d.lat])[0]},
        cy: function(d) {return projection([d.lon, d.lat])[1]}
    });
// what to do with ALL elements
selection.attr({
        r: radiusFunction
});
```

(d3places_03.html)

Two things are missing:

- we don't have a map initially
- maybe we want to add animations

Drawing the Initial Map

```
d3.csv('places-AT-cleaned.csv', function(data) {
    d3.select('#populationButton').on('click', function() {
        draw_map(data, populationRadius);
    });
    d3.select('#areaButton').on('click', function() {
        draw_map(data, areaRadius);
    });
    draw_map(data, populationRadius);
});
```

Animating Transitions

```
// in draw_map()
// ...
// what to do with ALL elements
selection.attr({
    r: radiusFunction
});
```

Animating Transitions

```
// in draw_map()
// ...
// what to do with ALL elements
selection.transition()
    .duration(800)
    .attr({
       r: radiusFunction
    });
```

duration... duration in milliseconds

(d3places_04.html)



Scales

```
var populationRadius = function(d) {
    return Math.sqrt(parseFloat(d.population)/500);
};

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
18
```

Scales

Scales

```
// var areaRadius = function(d) {
// return Math.sqrt(parseFloat(d.area)/10);
// };

var areaScale = d3.scale.sqrt()
    .domain([0,500])
    .range([0,10])
;

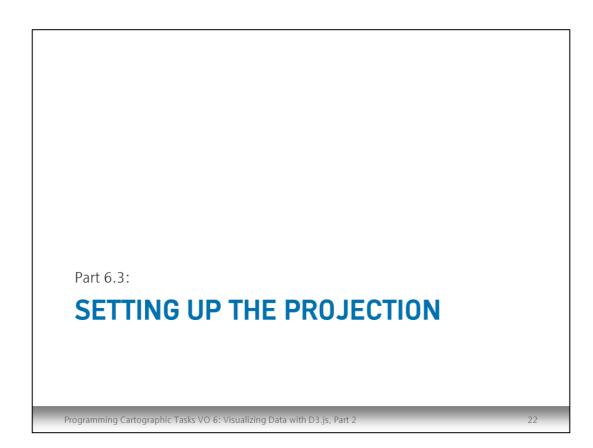
var areaRadius = function(d) {
    return areaScale(d.area);
};

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
```

More Scale Examples

```
var colorScale = d3.scale.linear()
    .domain([0,100])
    .range(['#e0ecf4','#8856a7'])
;
var thresholdColors = d3.scale.quantize()
    .domain([0,100])
    .range(['#edf8fb','#b3cde3','#8c96c6','#8856a7','#810f7c'])
;
var ordinalColors = d3.scale.ordinal()
    .domain(['rural','mixed','urban'])
    .range(['#99d8c9','#bdbdbd','#636363'])
;

Programming Cartographic Tasks VO 6: Visualizing Data With D3.js, Part 2
21
```



```
// so far, we set up our projection like this:
var projection = d3.geo.mercator()
    .translate([-950, 5710])
    .scale(5800);
;
```

Can we calculate these numbers?

```
function setup_projection(projection, geometry) {
    // reset projection
    projection.scale(1).translate([0,0]);

    // TODO: calculate projection parameters from geometry
    var scale = ???
    var translate = ???

    // apply the new parameters
    projection
        .scale(scale)
        .translate(translate);
}

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js. Part 2
24
```

This is the overall idea.

 \dots how can we calculate the projection parameters from our geometry? \dots

```
// we use a path generator to convert geometry into pixels
var pathGenerator = d3.geo.path().projection(projection);
var bounds = pathGenerator.bounds(geometry);

// bounds will now contain projected coordinates:
// [[left, top], [right, bottom]]

// TODO: set up projection parameters
var scale = ???
var translate = ???
```

```
// bounds: [[left, top], [right, bottom]]

// TODO: set up projection parameters
var scale = 0.95 / Math.max(
        (bounds[1][0] - bounds[0][0]) / width,
        (bounds[1][1] - bounds[0][1]) / height
);

var translate = [
        (width / 2 - (bounds[0][0] + bounds[1][0]) / 2 * scale),
        (height / 2 - (bounds[0][1] + bounds[1][1]) / 2 * scale)
];

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
26
```

We need to add the global variables width and height!

```
function setup_projection(projection, geometry) {
   projection.translate([0,0]).scale(1);
   var bounds = d3.geo.path().projection(projection).bounds(geometry);

   var scale = 0.95 / Math.max((bounds[1][0] - bounds[0][0]) / width,
        (bounds[1][1] - bounds[0][1]) / height);

   var translate = [
        (width / 2 - (bounds[0][0] + bounds[1][0]) / 2 * scale),
        (height / 2 - (bounds[0][1] + bounds[1][1]) / 2 * scale)
   ];
   projection
        .scale(scale)
        .translate(translate);
}

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
```

```
d3.json('bezirke.geojson', function(error, geometry) {
    // set up the projection as soon as geometry has loaded
    setup_projection(projection, geometry);
    // ...
});
Programming Cartographic Tasks VO 6: Visualizing Data with D3.js. Part 2
```

(d3places_06.html)

There is one problem remaining: we set up the projection after the geometry has loaded, but the dots may have been rendered earlier!

```
d3.csv('places-AT-cleaned.csv', function(data) {
    // PROBLEM: projection may not have been set up!
    // ...
});
d3.json('bezirke.geojson', function(error, geometry) {
    // set up the projection as soon as geometry has loaded
    setup_projection(projection, geometry);
    // ...
});
   Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
```

```
d3.json('bezirke.geojson', function(error, geometry) {
    // set up the projection as soon as geometry has loaded
    setup_projection(projection, geometry);
    // ...
    d3.csv('places-AT-cleaned.csv', function(data) {
         // SOLUTION: load csv data after geometry has loaded
    });
});
   Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2
```

Assignment 3

- Create an interactive choropleth map of Austria
 - Use D3.js to create a SVG map from geodata
 - Visualize income data of Austrian districts, available as fields of the properties of each GeoJSON feature

• income med Overall median income

income_med_m Median income of male workersincome_med_f Median income of female workers

- Provide 3 buttons to switch the visualization between these values
- Use a single color scale to set the fill color of each district, depending on the selected value.
 - Choose a suitable domain for the color scale
 - Choose a range of color values from colorbrewer2.org

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2

Assignment 3

- Bonus Points
 - Create a legend for the map using D3.js, using the *color* values as data to create the legend entries
 - Add a 'click' event handler to the district geometries that shows information (Name, data values) next to the map

Due date: May 22

Programming Cartographic Tasks VO 6: Visualizing Data with D3 is Part 2

Next Week

No lecture, only (final) programming tutorial (at 3PM)

Programming Cartographic Tasks VO 6: Visualizing Data with D3.js, Part 2