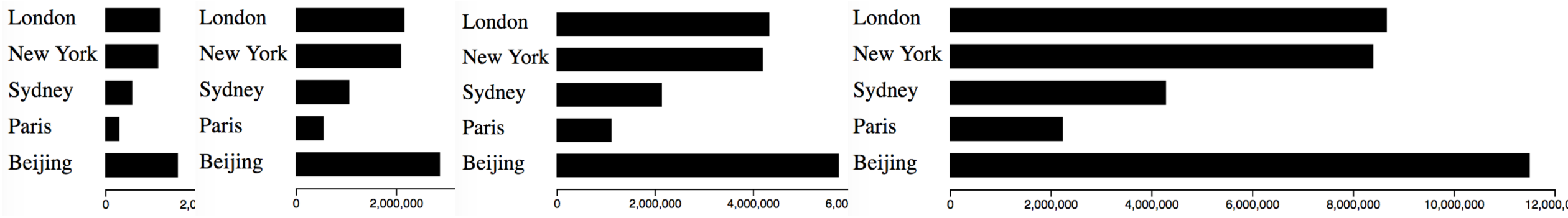
D3 Tutorial

Transitions

Edit by Jiayi Xu and Han-Wei Shen, The Ohio State University

Transitions

- When the state of an element changes from current state to desired state, transition helps to apply the change smoothly by interpolating the states between the two end states
  - E.g. gradually growing bars



Transitions

- Applications
  - Assist interactions
  - Story-telling
    - A series of visualizations connected by interactions and animations
    - A story about phones
      - https://www.visualcinnamon.com/2014/12/using-data-storytelling-with-chord.html

Populations of Cities - Slowly growing bars

```
var rects =  barChartG.selectAll("rect")
    .data(cities)
    .enter().append("rect")
    .attr("x", 80)
    .attr("y", function(d) {
        return bandScale(d.name);
    })
    .attr("height", bandScale.bandwidth())
    .style("fill", "black");
```

```
var t = d3.transition()
    .delay(200)
    .duration(1000);

rects
    .transition(t)
    .attr("width", function(d, i) {
        return pop2width(d.population);
    });
```

- Create bars and specify their attributes.
- By default, width is 0

Populations of Cities - Slowly growing bars

- After creating bars, we create a transition

```
var rects =  barChartG.selectAll("rect")
    .data(cities)
    .enter().append("rect")
    .attr("x", 80)
    .attr("y", function(d) {
        return bandScale(d.name);
    })
    .attr("height", bandScale.bandwidth())
    .style("fill", "black");

var t = d3.transition()
    .delay(200)
    .duration(1000);

rects
    .transition(t)
    .attr("width", function(d, i) {
        return pop2width(d.population);
    });
```

- Create a *transition* behavior
- delay(*milliseconds*): the transition will happen after a certain milliseconds
- duration(*milliseconds*): the transition will last a certain milliseconds

Populations of Cities - Slowly growing bars

```
var rects =  barChartG.selectAll("rect")
    .data(cities)
    .enter().append("rect")
    .attr("x", 80)
    .attr("y", function(d) {
        return bandScale(d.name);
    })
    .attr("height", bandScale.bandwidth())
    .style("fill", "black");

var t = d3.transition()
    .delay(200)
    .duration(1000);

rects
    .transition(t)
    .attr("width", function(d, i) {
        return pop2width(d.population);
    });
```

- Bind the *transition* behavior with bars
- Then, the bars will gradually grow to the final width

Populations of Cities - Slowly growing bars

- Two alternative ways of coding have the same effect

```
var t = d3.transition()
    .delay(200)
    .duration(1000);

rects
    .transition(t)
    .attr("width", function(d, i) {
        return pop2width(d.population);
    });
```

```
rects
    .transition()
    .delay(200)
    .duration(1000)
    .attr("width", function(d, i) {
        return pop2width(d.population);
    });
```

Events

- We can bind listeners to events of *transition* behaviors
- d3.transition().on(*EventType*, *listener*)
- Events
  - start
    - Be triggered at the beginning of the *transition* behavior
  - end
    - After the *transition* behavior ends

```
var t = d3.transition()
    .delay(200)
    .duration(1000)
    .on('start', started)
    .on('end', ended);
```

Transition Chaining

- We can create multiple transitions
  - When one transition finishes, next transition in the chain takes off

```
<svg width="1000" height="1000">
    <circle r="100" fill="blue" transform="translate(100, 100)"></circle>
</svg>
<script type="text/javascript">
    d3.select("circle")
        // Transition 1
        .transition()
            .duration(2000)
            .attr("fill", "red")
        // Transition 2
        .transition()
            .duration(2000)
            .attr("fill", "blue")
            .attr('transform', 'translate(600, 300)');
</script>
```

- Transition 1: The color of the circle changes from blue to red;
- Transition 2: The color of the circle changes back to blue, and the circle is moved to (600, 300).