# exercise session 3

# pointing stick

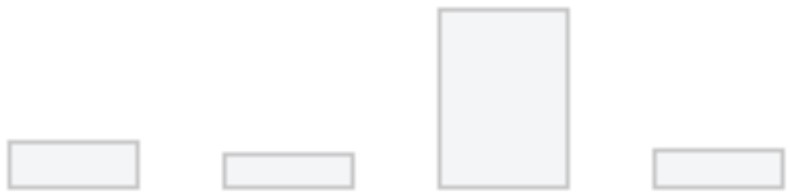
- visually describe the pointing stick in card’s design space

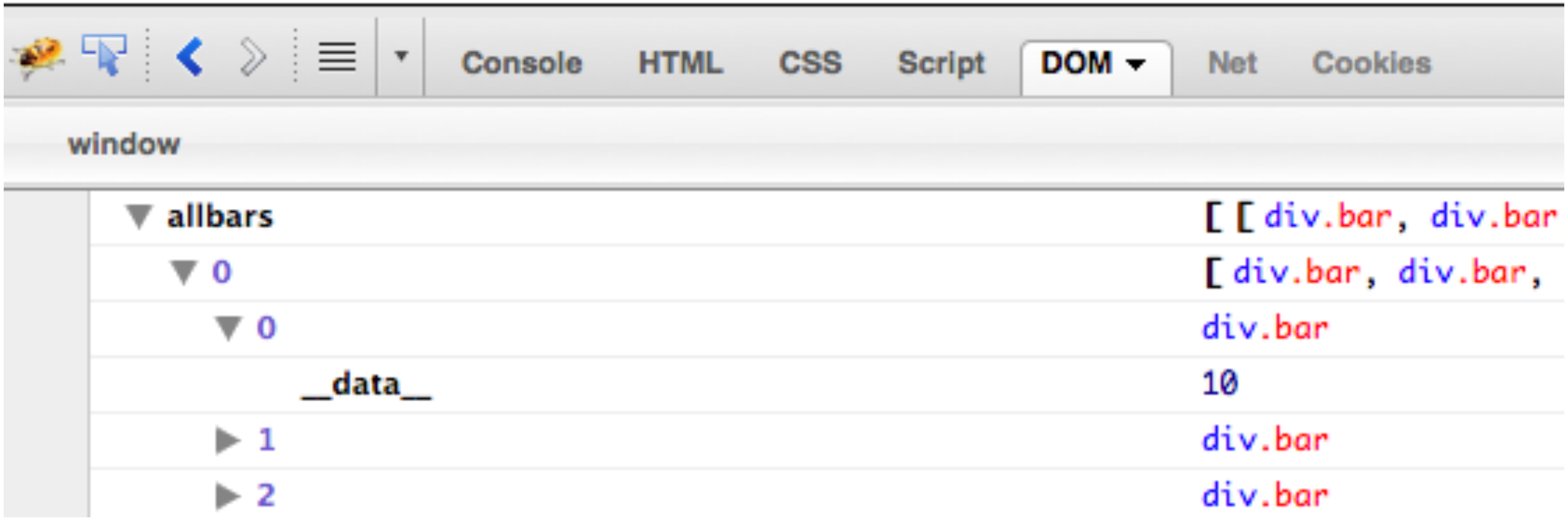
- effectiveness: *the input conveys the intended meaning with felicity*
  - how well does an input device convey users’ intention in terms of speed, errors, or other qualities.
    - **desk footprint**, **pointing precision**, errors, user preference, cost, ...

Literature: Card, S. et al. A morphological analysis of the design space of input devices, April 1991, ACM Trasnaction on Information systems

# data joins

- command graphical elements to enter, update and exit.
- data is connected to elements (data binding)

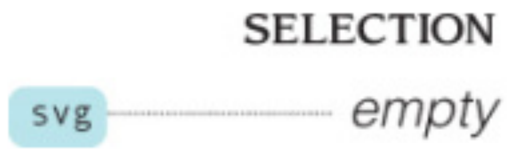




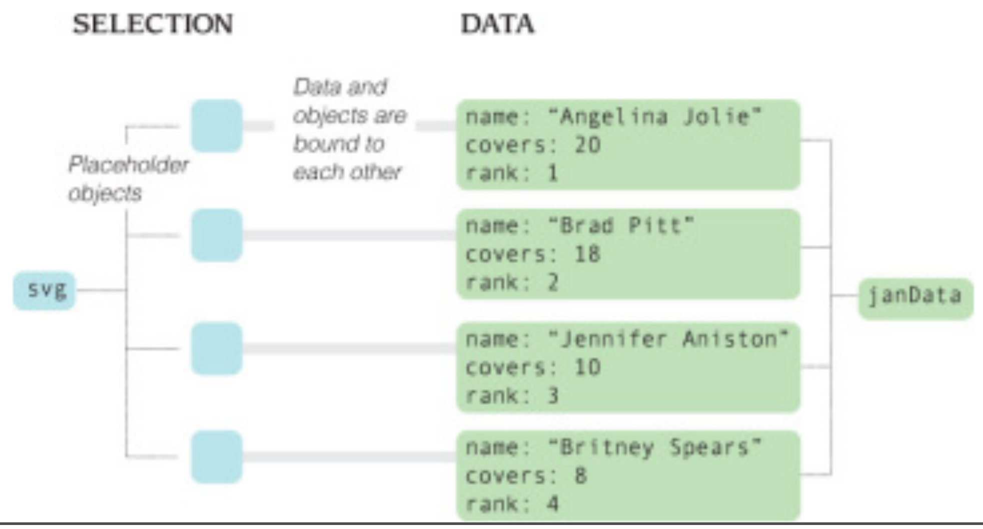
```
▼ allbars                                    [ [ div.bar, div.bar
    ▼ 0                                      [ div.bar, div.bar,
        ▼ 0                                  div.bar
              __data__                       10
        ▶ 1                                  div.bar
        ▶ 2                                  div.bar
```

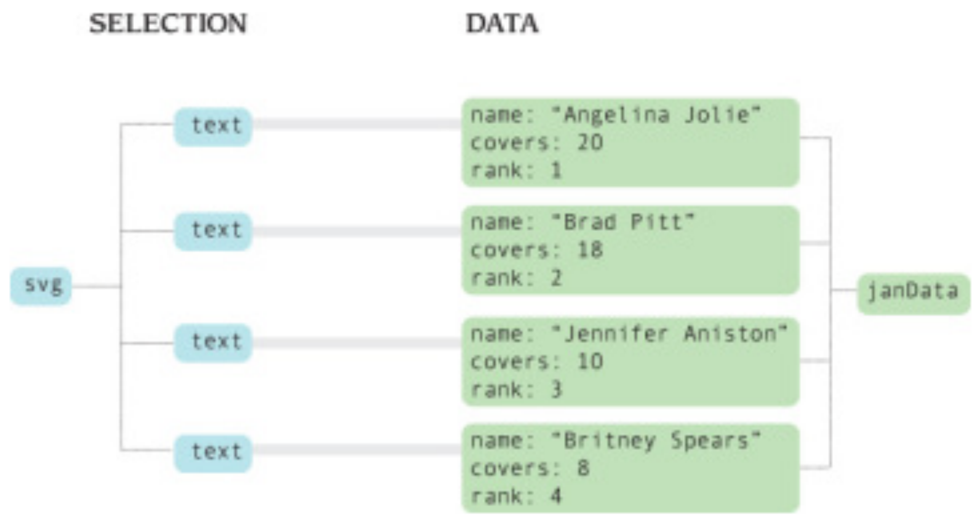
# enter phase

- create a selection of elements that do not exist yet (e.g. d3.selectAll(“p”))

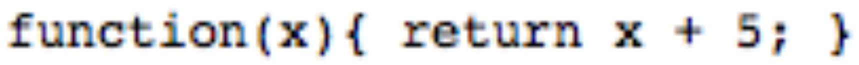


- use two methods: data(), enter()
  - placeholders, need to append() objects



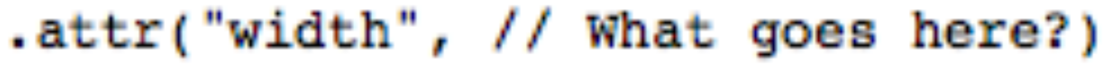


# anonymous functions

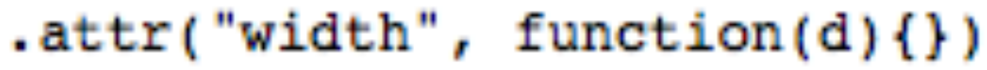


```
function(x){ return x + 5; }
```

- to access bound data
- anonymous functions had not been bound to an identifier (no name)

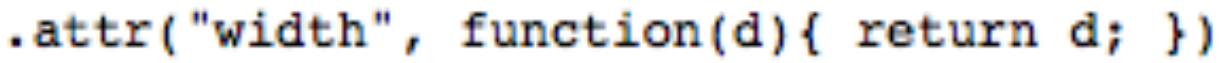


```
.attr("width", // What goes here?)
```



```
.attr("width", function(d){})
```

- d represents the bound data point

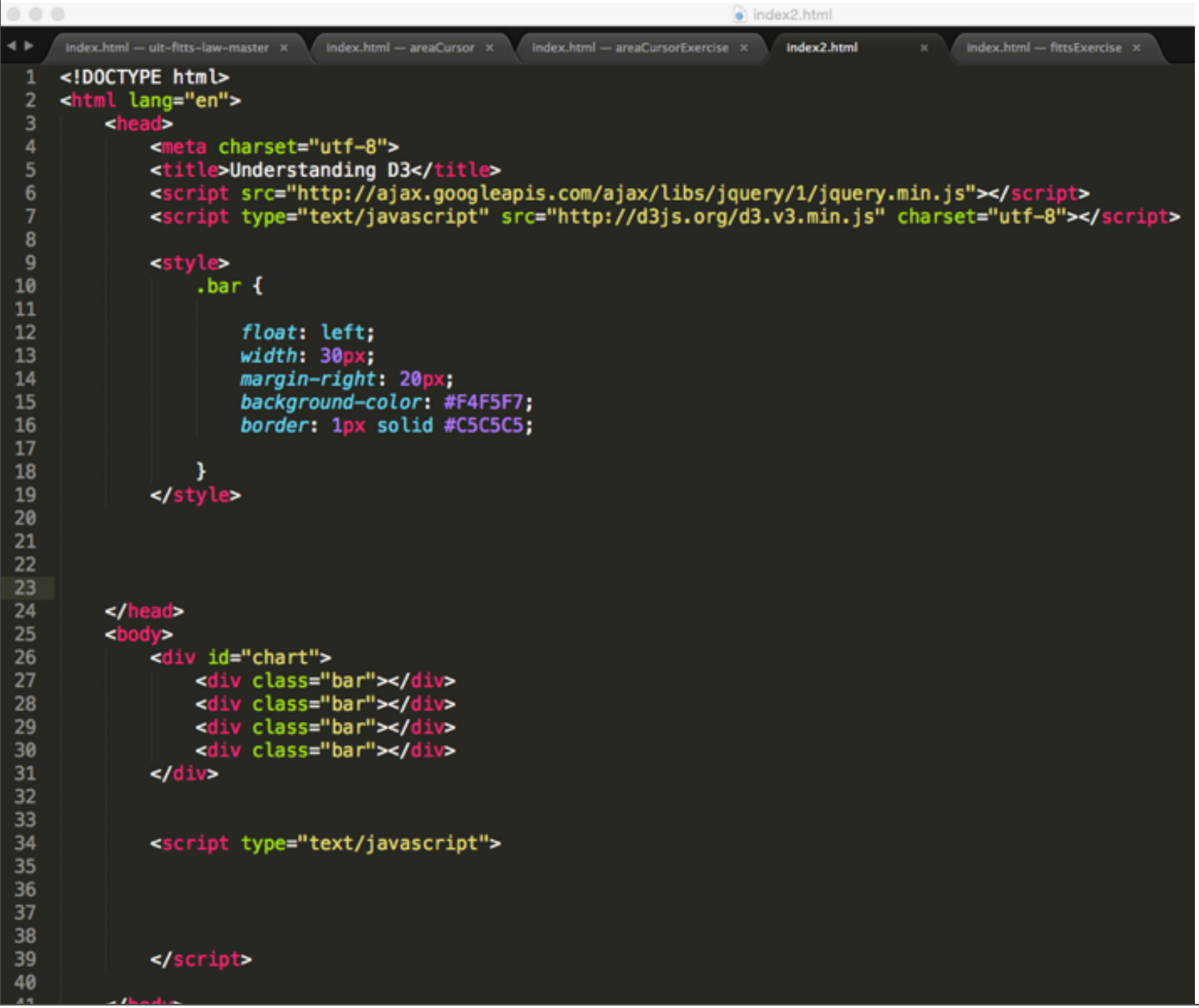


```
.attr("width", function(d){ return d; })
```

# update and exit phase

- update: use data() method, no enter()
  - update attributes as needed
- remove: use method exit() and remove()
  - selection.exit().remove();

# let’s try it …

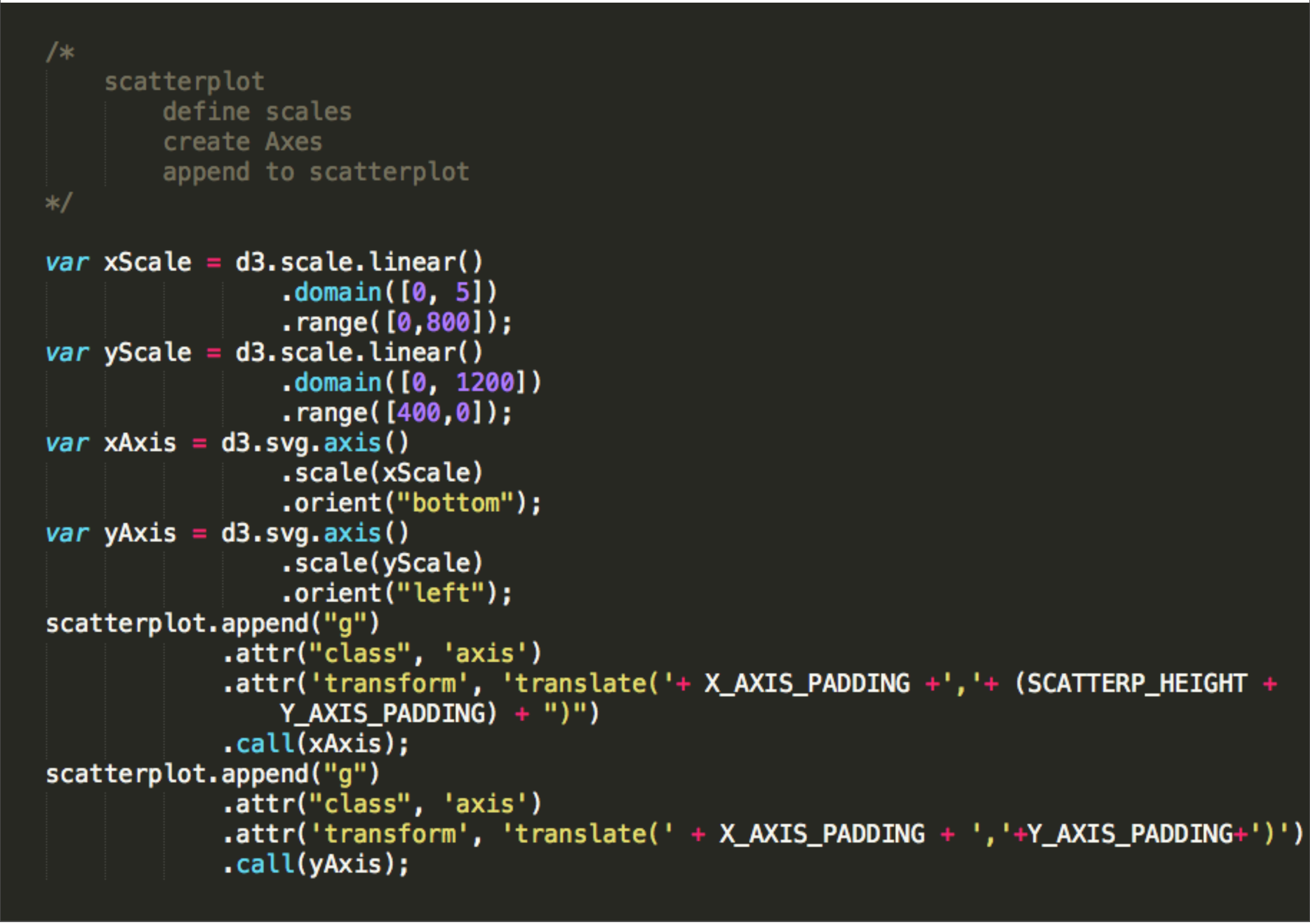


index2.html
index.html — uit-fitts-law-master
index.html — areaCursor
index.html — areaCursorExercise
index2.html
index.html — fittsExercise

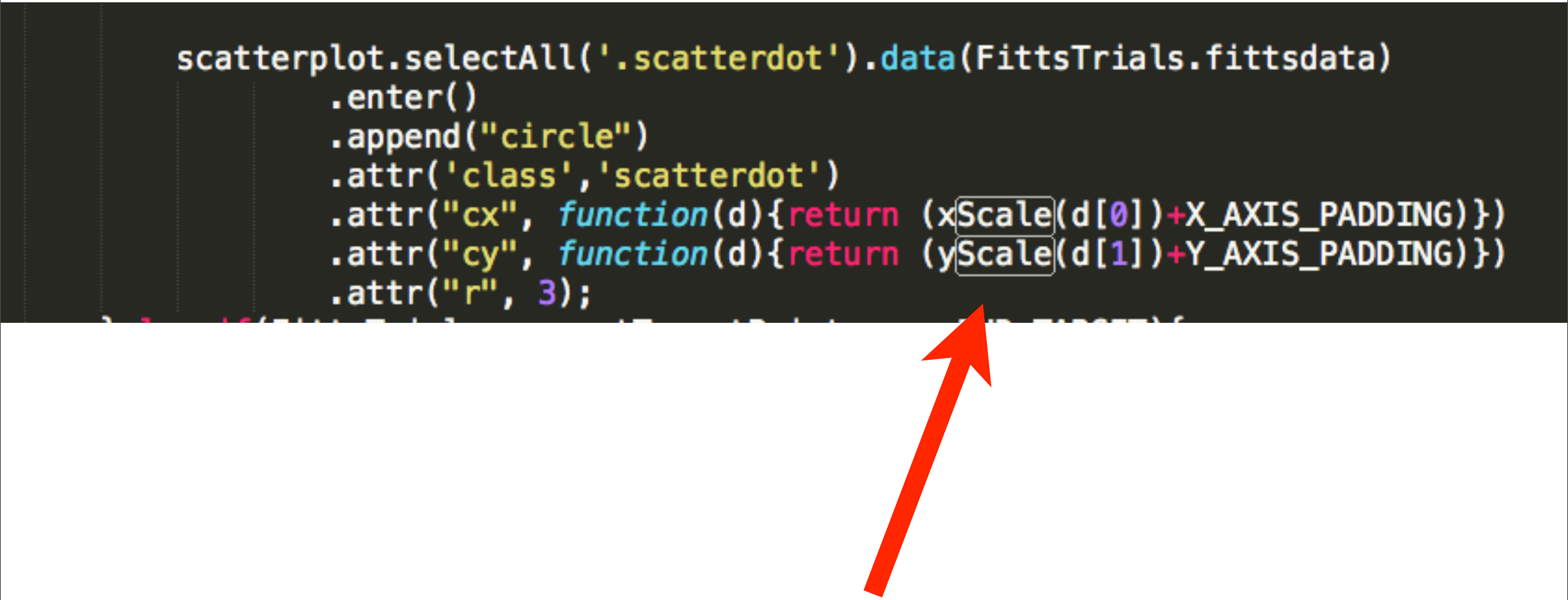
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>Understanding D3</title>
        <script src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.min.js"></script>
        <script type="text/javascript" src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>

        <style>
            .bar {

                float: left;
                width: 30px;
                margin-right: 20px;
                background-color: #F4F5F7;
                border: 1px solid #C5C5C5;

            }
        </style>




    </head>
    <body>
        <div id="chart">
            <div class="bar"></div>
            <div class="bar"></div>
            <div class="bar"></div>
            <div class="bar"></div>
        </div>


        <script type="text/javascript">




        </script>

    </body>
```

# scales and axes

- Scales are functions that map from an input domain to an output domain.
    - we just look at linear scales here, since most common
- input domain: range of possible input data values.
- output range: range of possible output values, display values in pixel units



```
/*
    scatterplot
        define scales
        create Axes
        append to scatterplot
*/

var xScale = d3.scale.linear()
                .domain([0, 5])
                .range([0,800]);
var yScale = d3.scale.linear()
                .domain([0, 1200])
                .range([400,0]);
var xAxis = d3.svg.axis()
                .scale(xScale)
                .orient("bottom");
var yAxis = d3.svg.axis()
                .scale(yScale)
                .orient("left");
scatterplot.append("g")
            .attr("class", 'axis')
            .attr('transform', 'translate('+ X_AXIS_PADDING +','+ (SCATTERP_HEIGHT +
                Y_AXIS_PADDING) + ")")
            .call(xAxis);
scatterplot.append("g")
            .attr("class", 'axis')
            .attr('transform', 'translate(' + X_AXIS_PADDING + ','+Y_AXIS_PADDING+')')
            .call(yAxis);
```



```
        scatterplot.selectAll('.scatterdot').data(FittsTrials.fittsdata)
                .enter()
                .append("circle")
                .attr('class','scatterdot')
                .attr("cx", function(d){return (xScale(d[0])+X_AXIS_PADDING)})
                .attr("cy", function(d){return (yScale(d[1])+Y_AXIS_PADDING)})
                .attr("r", 3);
```