

D3 Cheat Sheet

Scott [Murray](#) (@d3js) d3js.org — January 6, 2014

Selections

Explanation

<code>d3.select()</code>	Returns the element found
<code>d3.selectAll()</code>	Returns all found elements
<code>selection.append()</code>	Creates a new element inside the selection
<code>selection.remove()</code>	Removes the selection from the DOM
<code>selection.text()</code>	Sets the text content of the selection
<code>selection.attr()</code>	Set an HTML attribute value on the selection
<code>selection.style()</code>	Set an inline CSS style on the selection
<code>selection.classed()</code>	Adds or removes a class from the selection

Example

```
d3.select("svg")
d3.selectAll("circle")
d3.select("svg").append("circle")
d3.select("rect").remove()
d3.select("#tooltip").text("")
d3.selectAll("circle").attr("r", 10)
d3.selectAll("circle").style("fill", "teal")
d3.select("circle").classed("highlight", true)
```

Data

<code>selection.data()</code>	Binds an array of data values to the selection
<code>selection.enter()</code>	Returns a selection of “new” placeholder elements

Use anonymous functions to access data values bound to elements via `d`.

Optionally, include `i` to get the index value of each element in the selection.

```
d3.selectAll("circle").data(dataset).enter()...
d3.selectAll("circle").data(dataset).enter()...
d3.selectAll("rect")
  .attr("height", function(d) {
    return d.value; // Set the height to 'value'
  });
d3.selectAll("rect")
  .attr("x", function(d, i) {
    return i * 10; // Move each rect to the right
  });
```

Transitions

<code>selection.transition()</code>	Initiates a new transition
<code>selection.duration()</code>	Sets the transition duration, in milliseconds

```
d3.selectAll("circle").transition().attr("cx", ...)
d3.selectAll("circle").transition().duration(2000)...
```

Scales

<code>d3.scale.linear()</code>	Creates a new linear scale function
<code>scale.domain()</code>	Sets the scale's input domain
<code>scale.range()</code>	Sets the scale's output range
<code>d3.min()</code>	Returns the smallest value in an array
<code>d3.max()</code>	Returns the largest value in an array

```
var xScale = d3.scale.linear()
                        .domain([ 0, 2000 ])
                        .range([ 0, width ]);
d3.min([ 10, 20, 70, 35 ]) // Returns 10
d3.max([ 10, 20, 70, 35 ]) // Returns 70
```

Axes

<code>d3.svg.axis()</code>	Creates a new axis generator function
<code>axis.scale()</code>	Specifies the scale to be used with this axis
<code>axis.orient()</code>	Specifies the orientation for this axis
<code>axis.ticks()</code>	Suggests a number of ticks for this axis
<code>selection.call()</code>	Calls a method; used to generate an axis

```
var xAxis = d3.svg.axis()
                .scale(xScale)
                .orient("bottom")
                .ticks(5);
svg.append("g").call(xAxis);
```

Interactivity

<code>selection.on()</code>	Binds an event listener to a selection
-----------------------------	--

Within an anonymous function, **this** refers to “the element being acted upon.”

```
d3.select("#button").on("click", function() { ... });
d3.selectAll("rect")
    .on("mouseover", function() {
        d3.select(this).classed("highlight", true);
    });
```

Other Useful JavaScript

<code>Math.random()</code>	Returns a random value between 0.0 and 1.0
<code>Math.floor()</code>	Rounds down to the nearest integer
<code>array.push()</code>	Appends a new value to an existing array

```
Math.random() * 100 // Could return 61.87844036612...
Math.floor(61.87844036612) // Returns 61
var numbers = [ 2, 3, 4, 5 ];
numbers.push(6); // Now numbers is [ 2, 3, 4, 5, 6 ]
```