# D3.js Cheatsheet

## d3.js quick reference sheet

### Table of contents



### Attributes/Styles

#### Attributes/Styles - initialization

```
var svgWrapper = d3.select("body")
  .append("svg")
  .attr("id", "viz")
  .attr("width", width + margin.left + margin.right)
  .attr("height", height + margin.top + margin.bottom);

var container = svgWrapper.append("g")
  .attr("transform", "translate(" + margin.left + "," + margin.top + ")")
  .style("pointer-events", "all");
```

#### Attributes/Styles - circle

```
selection.attr("r", style.r)
  .attr("fill", style.fill)
  .attr("stroke", style.color)
  .attr("stroke-width", style["stroke-width"])
```

### Classes

```
selection.classed('my-class', true);
```

### Link/Href

#### In SVG

```
selection.select('text')
  .text(function(d) {
    return d.name;
  })
  .attr("xlink:href", function(d){
    return d.url;});
```

#### In HTML

```
selection
  .append('a')
  .attr('href', function(d) {
```

```
    return d.url;
  })
  .text(function(d) {
    return d.name;
  });
```

## Properties

```
var id = d3.select("#id").property("value");
d3.select("input").property("value", d3.event.keyCode);
```

## Axis

```
var xAxis = d3.svg.axis()
  .scale(_xScale)
  .orient("bottom")
  .ticks(4)
  .tickFormat(d3.time.format('%d %b %I%p')) //14 Feb 12AM
  .tickSize(5);
```

## Time parsing/formatting

```
var parseDate = d3.time.format("%Y-%m-%dT%H:%M:%SZ").parse,
  formatDate = d3.time.format("%d %b %H:%M:%S"),
  formatDateForQuery = d3.time.format("%Y-%m-%dT%H:%M:%SZ"),
  formatTime = d3.time.format("%H:%M:%S");
```

## Color

### Custom color ordinal scale

```
var myCategory20Colors = [
  0x1f77b4, 0xaec7e8,
  0xff7f0e, 0xffbb78,
  0x2ca02c, 0x98df8a,
  0xd62728, 0xff9896,
  0x9467bd, 0xc5b0d5,
  0x8c564b, 0xc49c94,
  0xe377c2, 0xf7b6d2,
  0xbcbd22, 0xdbdb8d,
  0x17becf, 0x9edae5
].map(function(x) {
  var value = x + "";
  return d3.rgb(value >> 16, value >> 8 & 0xff, value & 0xff).toString();
});

var myCategory20 = d3.scale.ordinal().range(myCategory20Colors);
console.log(myCategory20("x"), myCategory20("y"));
// #1f77b4 #aec7e8
```

## General Update Pattern

```
function update(data) {

  // DATA JOIN
  // Join new data with old elements, if any.
  var text = svg.selectAll("text")
    .data(data);

  // UPDATE
  // Update old elements as needed.
  text.attr("class", "update");

  // ENTER
  // Create new elements as needed.
  text.enter().append("text")
```

```
    .attr("class", "enter")
    .attr("x", function(d, i) { return i * 32; })
    .attr("dy", ".35em");

  // ENTER + UPDATE
  // Appending to the enter selection expands the update selection to include
  // entering elements; so, operations on the update selection after appending to
  // the enter selection will apply to both entering and updating nodes.
  text.text(function(d) { return d; });

  // EXIT
  // Remove old elements as needed.
  text.exit().remove();
}
```

## Transition

### Chaining transition

```
function endall(transition, callback) {
  if (transition.size() === 0) {
    callback()
  }
  var n = 0;
  transition
    .each(function() {
      ++n;
    })
    .each("end", function() {
      if (!--n) callback.apply(this, arguments);
    });
}

selection.transition()
  .attr("cx", xMap)
  .attr("cy", yMap)
  .call(endall, function() {
    console.log("all loaded");
    // do your next transition
  });
```