**Notes on D3 for Data Visualization**

It is possible to connect to a firestone database on the web (see another course if you want to do this.

3 Types of data visualizations: Explore (find an unknown), analyze (test a hypothesis), present (tell a story)

D3 blows out of the water the other chart in browser visualization tools because of

* Flexibility: load your own data from any source. It matches data to shapes. Make completely new visualizations.
* Elegance: Handles huge volumes of data efficiently, easily add smooth transitions for updating elements, and works with clean, well-designed code.
  + It’s like building a house out of clay instead of legos. Or making a custom, from scratch, home made pizza instead of a frozen pizza or one ordered from Dominoes.
* Community: Find pre-bulid code to use in oyour own projects. Everyting is open source. Wide user base makes it easy to collaborate.

You get to do 4 different projects. I really like the 3rd and 4th ones.

Section 1-6: Dynamic and interactive visualizations

Section 7 – 9: 3 Sections

Section 10: Complex linked views, updates for several charts at once.

Or you can use php and node with mysql.

**Understanding SVGs (2.10)**

Scalable Vector Graphics (SVGs) allow us to draw shapes in a browser window. Use familiar html syntax to do it. D3 uses SVGs rather than images to display data because svgs don’t lose quality if they’re resized, and they don’t take up much space/memory. You can have thousands of svgs on the screen inseconds, and there won’t be a significant lag when we want to move them all around at once. For example, Google’s new svg logo takes up 1/70 of the space of the old jpeg logo.

You can either include svgs directly in html or in javasript files.

Use the svg tag. You can specify width & height.

X, y coordinates: Remember that the coordinates (the top left) of your shape is set with x and y, and if you set them outside of the area of the svg, they won’t appear on the screen.

<rect

*x*="0" *y*="0" *width*="50" *height*="50" *fill*="green" *stroke*="grey" *stroke-width*="5px"> </rect>

Stroke: Give shapes outline, it’s a border on the outside of the shape.

Cirlce and ellipse: use cy and cx, and rx and ry Coordinates for circles are based on the center of the circle.

<circle *cx*="95" *cy*="25" *r*="25" *fill*="red"></circle>

<ellipse *cx*="145" *cy*="25" *rx*="15" *ry*="25" *fill*="grey"></ellipse>

Line: use y1, y2, x1, x2

(see mark-index.html)

Path: A path is the most powerful kind of SVG that we have available, and it can be used to create lines, curves, arcs and all kinds of complex shapes. You can use them for maps and other things.

**D3 Select** – Grab hold of element on the screen.

const svg = d3.select(“#canvas”)

D3 Append – add svgs onto your selection

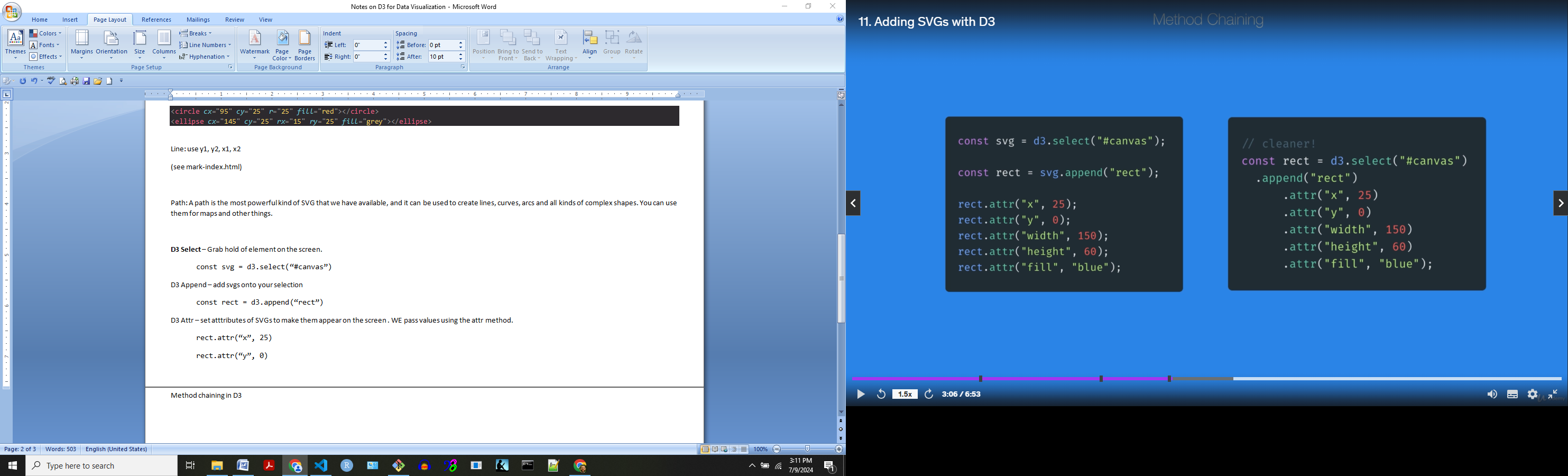
const rect = d3.append(“rect”)

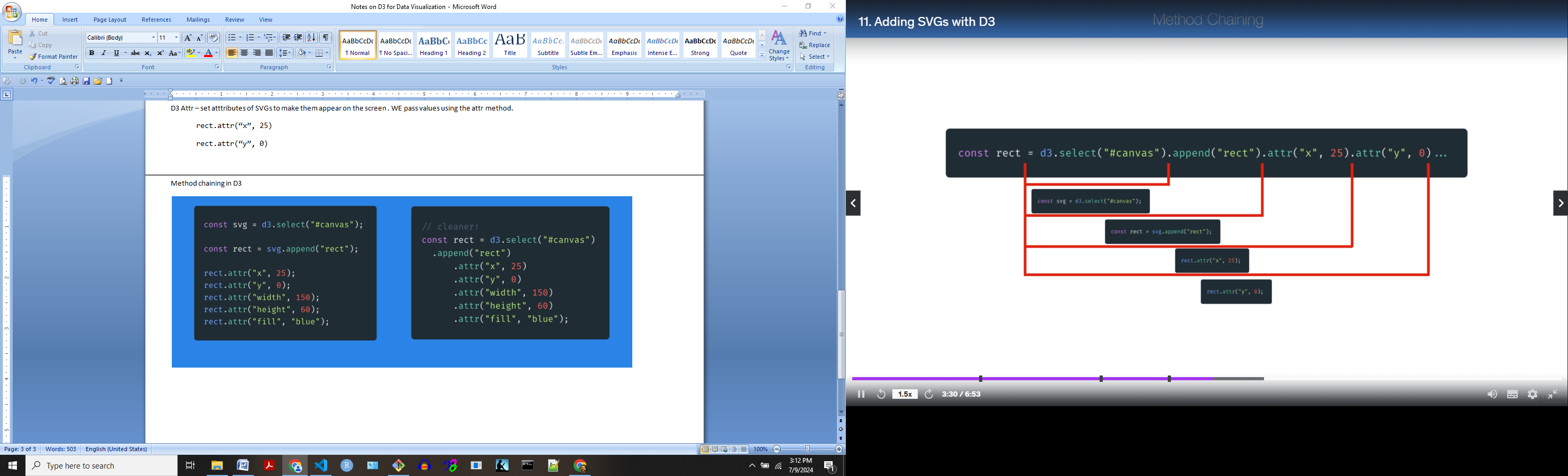
D3 Attr – set atttributes of SVGs to make them appear on the screen . WE pass values using the attr method.

rect.attr(“x”, 25)

rect.attr(“y”, 0)

Method chaining in D3





I got to where I can draw an svg shape on the screen. I need to start of tomorrow on 2.13 “Selections and data joins”