(30%) Tool Explanation

Your video explains the tool and its purpose. Your video explains what problem it solves, as well as the limitations of the tool for solving problems.

(30%) Tool Demonstration

Your video includes a demonstration of the tool's usage in a worked example. This demonstration is effective and showing how the tool is used in practice.

(10%) Video Engagement

Your video is watchable and engaging, and effective and conveying the information you present. That is, you have paid attention to the "user experience" of your tutorial.

(15%) Demonstration Code

You have included both "starter" and "completed" version of your demonstration code. This code is well organized and readable, provided a good reference and example for how to use the tool.

(15%) Peer Review

You have completed another students tutorial, and pushed your version of the code to a repository. Your review demonstrates that you understand the tool that was taught, and includes constructive feedback.

-Why:

The justification for the tool. What problem does it solve?

You should also consider the limitations of the tools—what problem(s) does it not solve (or solve poorly?) In short: when would someone want to use the tool, and when would they not?

-What:

A description of the tool—what is it? You might also include some context or history for the tool (e.g., if it was created by a particular company, or if it interacts with the web ecosystem in a particular way).

-How:

A demonstration of the tool in action. This should be a worked example, similar to the demos that we've done in class (that means working code!). Show us exactly how to use the tool to solve its particular problem.

You are welcome to adapt a demo from the tool's documentation, but don't copy it straight out! Personalize it in some way (so there are more examples in the world). For example, since the Angular.js tutorial builds a phone catalog, yours might build a coffee catalog.

-Your video should also include name (as the author)—take credit for your work!

Hello, everyone. This is Huijie. Today I will introduce you a cool JavaScript library called D3.

This teaching video will be formed by 3 parts:

1. what is d3.
2. what problems do d3 solve and what limitation it has.
3. a demonstration of d3.

(slide)

First, what is D3?

The full name of D3 is Data-Driven Documents. It’s a JavaScript library for producing dynamic, interactive data visualizations in web browsers. (based on wikipedia)

D3’s emphasis on web standards gives user the full capabilities of modern browsers without tying themselves to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

The first edition V1.0 of D3 was released by Michael Bostock on February 18th 2011. The latest edition is v3.4.11. Michael used to make a javascript library called Protovis to realize data visualization, now it’s replaced by D3.

(slide)

Now let’s talk about what can you do with D3.

Users can use D3 to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use an array of number to create an interactive SVG bar chart with smooth transitions and interaction.

Some advantages D3 have.

D3 efficiently manipulate documents based on data, which avoids proprietary representation and affords extraordinary flexibility, exposing the full capabilities of web standards such as HTML, SVG, and CSS.

D3 is extremely fast, supporting large datasets and dynamic behaviors for interaction and animation. D3’s functional style allows code reuse through a diverse collection of components and plugins.

(slide)

However, it also has limitations.

-D3 doesn't always try to support older browsers. If you want users on those browsers to see your visualization, you may have to use a static placeholder (in the case it doesn't load).-

-D3 has some data-source limitations.

-D3 cannot easily conceal original data. If you're using data that you don't want shared, it can be challenging to use D3.

(slide)

Now, let me teach you how to realize data visualization by using d3 to draw a chord.

We should first understand what a chord do.

A chord shows the relation between two elements.

if two elements are connected by line, it means they are related.

the (degree of) thickness stands for weight of relationship.

after understanding what is chord, we are ready to draw it.

first, please open the starter code in my public GitHub repository called d3.

you can find the link of my github below my video.

okay, then you will find some original data source, including city name and sample

let me explain the data a little bit.

the variable city name includes four main cities of the china

which are beijing shanghai shenzhen and hongkong, and other.

the variable sample shows how the sample of one city formed by people from other cities.

take the first line as example, the sample in beijing was formed by 3000 people from beijing, 1045 from shanghai, 504 from shenzhen, 327 from hk, and 3714 from other cities.

after explaining the data source. let’s visualize the data!

we divide the code to 4 part

1. set layout and transform data
2. set svg, chord and color definition
3. draw the element group for each city and the city name
4. draw the inner chords

here is the layout of the chord.

we set the gap of the elements,sort it, and input the matrix.

then transform data to apply this layout.

after transformation, sample becomes groups and chords

group is the elements, chord is the link

chords can be divided to source and target, which are the two ends of the link.

we can look at the console to see what data do we have now.

now let’s set the svg chord and color definition

we define width,height, inner radius and outer radius

and color, which use d3 scale category.

we also define the attributes of svg using d3

this part is pretty easy so i don’t plan to explain too much.

okay, now it’s time to draw elements group for each city and city name.

the elements are in the outside of the chord. each element group has start angle and end angle, thus the elements can be shown by arc. we build the arc by providing inner and outer radius.

inside the svg, we can create variable g-outer to select path and text.

here, we make the text in the middle of the arc

then we use transform attribute to make the text look better in the graph using mathematical way.

at last, let’s draw inner chord. not like circle, we don’t have chord element in svg, thus we need use path to create. d3 provides d3.svg.chord(). we only need to pass the obeject of the chord, then we can get the path.

here is a bonus, let’s make some interactive function.

how about change the color of the chord when we touch it.

we can use mouseover and mouseout.

okay, here is our works

we just drew a pretty cool chord.

(slide)

d3 is a very simple and useful javascript library, and it’s getting more and more popular.

today we use 10 mins to learn a very simple drawing using d3

if you like it, i provided some link below the video for you to go deeper and make more complicated drawings.

thank you for watching my video, and this is huijie.

Thanks byebye.