

# D3 Workshop

Nami Sumida  
February 17, 2019

# Examples of what you can create with D3

## Nexus Blitz play and win rates

Compared to other game modes during the time of the alpha

Show:

Play count

Win rate

Sort by:

Play count

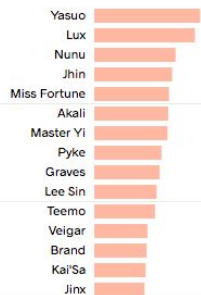
Champion name

Filter:

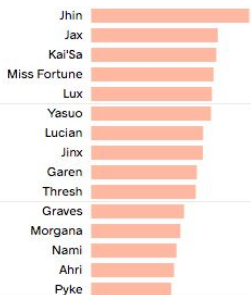
All classes

Click on each champion for details.

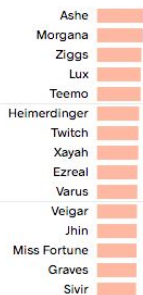
### Nexus Blitz



### Ranked 5v5



### ARAM



## Nexus Blitz jungle duos

Explore win and play rates for different combinations of your favorite junglers.

Nunu



Search for a champion:

Sort by: Win rate

# games

Champion name

Show pairs with at least 200 games played:



# HTML, CSS & JavaScript

# 1) HTML

— — —

- Hypertext Markup Language
- A tool for specifying *semantic structure* of your content (i.e. attaching hierarchy and relationships)

## **Without structure:**

How to create interactive data visualizations for the web You'll need the following: HTML CSS JavaScript

## **With structure:**

**How to create interactive data visualizations for the web**

You'll need the following:

- HTML
- CSS
- JavaScript

## Without structure:

How to create interactive data visualizations for the web You'll need the following: HTML CSS JavaScript

## With structure:

**How to create interactive data visualizations for the web**

You'll need the following:

- HTML
- CSS
- JavaScript

Headline

Paragraph text

Unordered list with three items

## 2) CSS

— — —

- Cascading Style Sheets
- Used to style the visual presentation of your content

## Without CSS styles:

How to create interactive data visualizations for the web

You'll need the following:

- HTML
- CSS
- JavaScript

## With CSS styles:

How to create interactive data visualizations for the web

You'll need the following:

- HTML
- CSS
- *JavaScript*

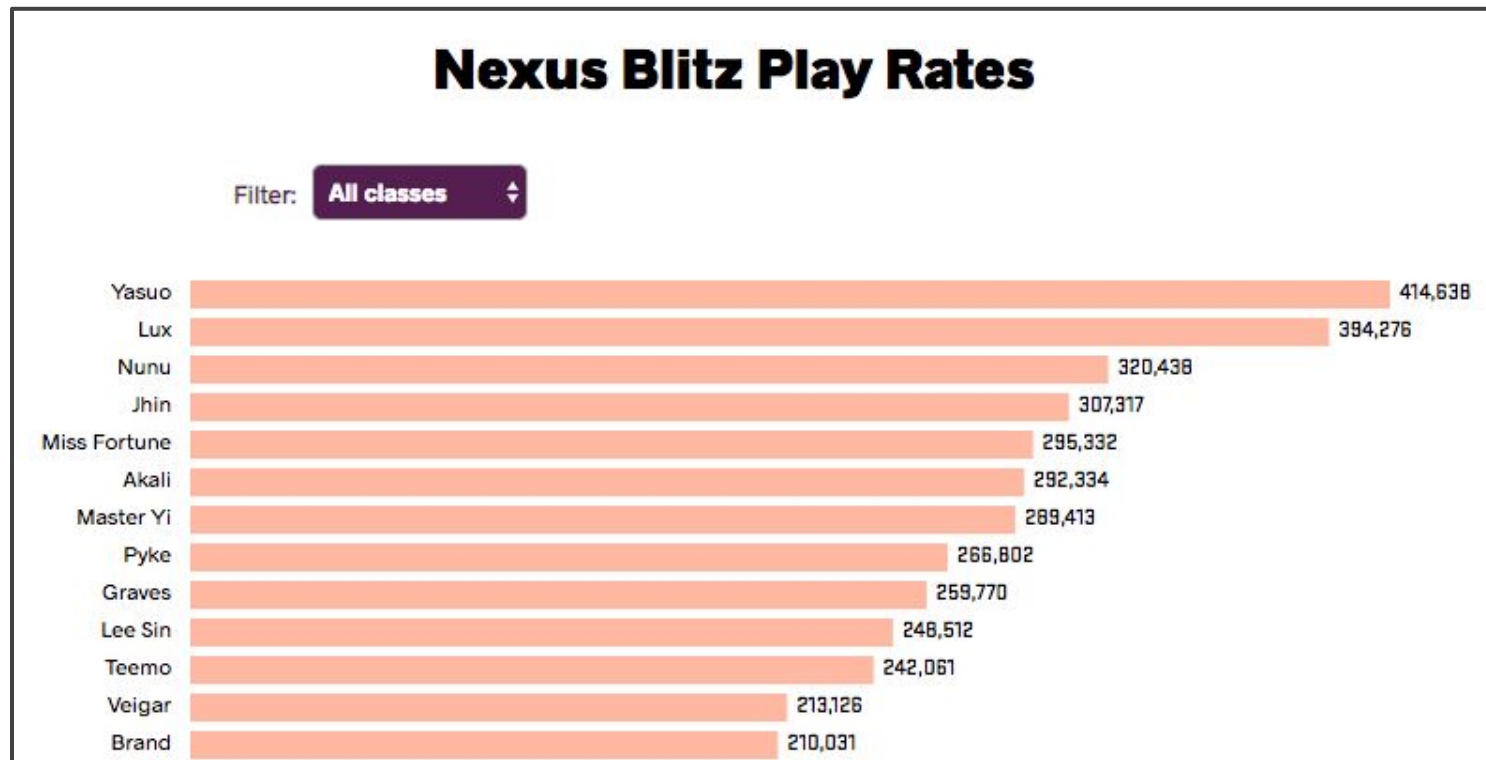


# 3) JavaScript

— — —

- Scripting language that can make pages dynamic by manipulating your content (after a page has already loaded in your browser)
- Examples:
  - Mouseover
  - Click
  - Filtering

# What we'll be creating today



What is HTML doing? Defining the elements on the page

Header

## Nexus Blitz Play Rates

Filter:

All classes



Dropdown + what goes in the dropdown

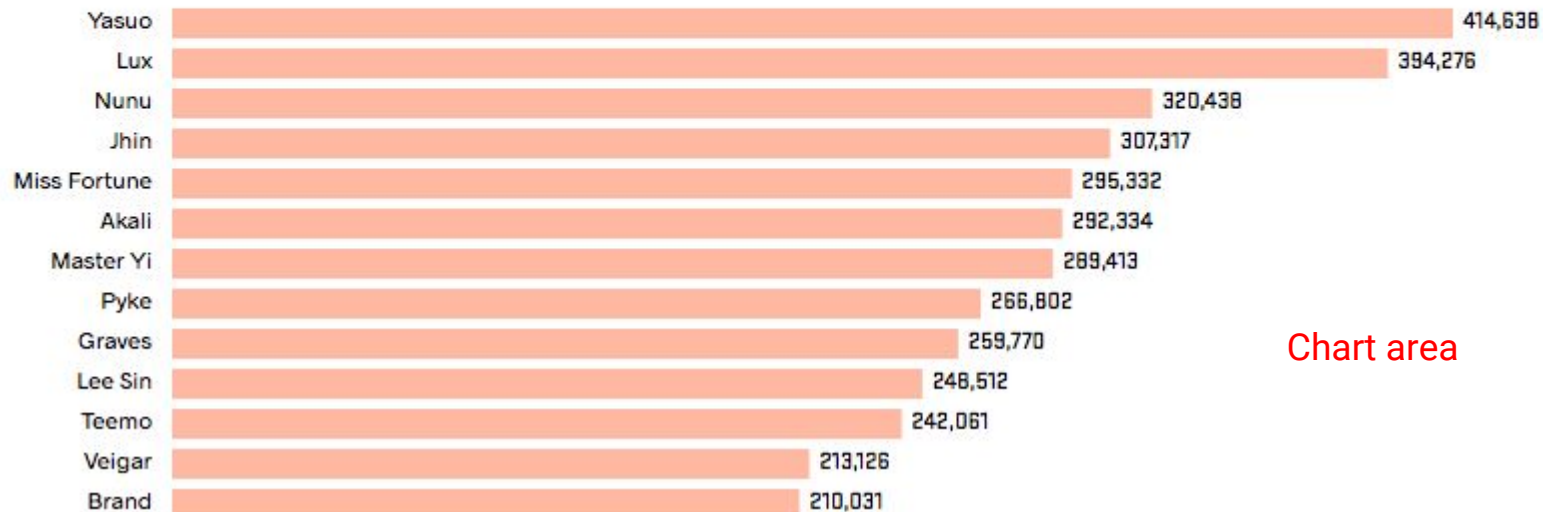


Chart area

What is CSS doing? Defining the styles for each element

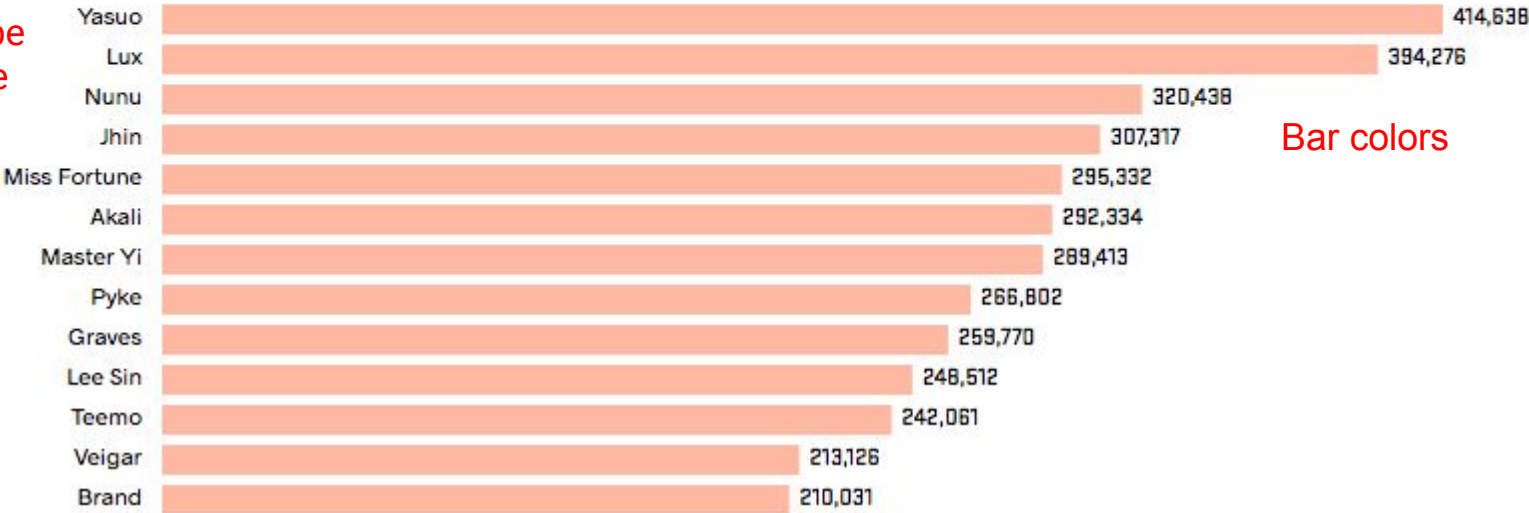
Font type, size,  
placement of text

Nexus Blitz Play Rates

Filter: All classes

Width of dropdown, font type, size, background color

Font type  
and size



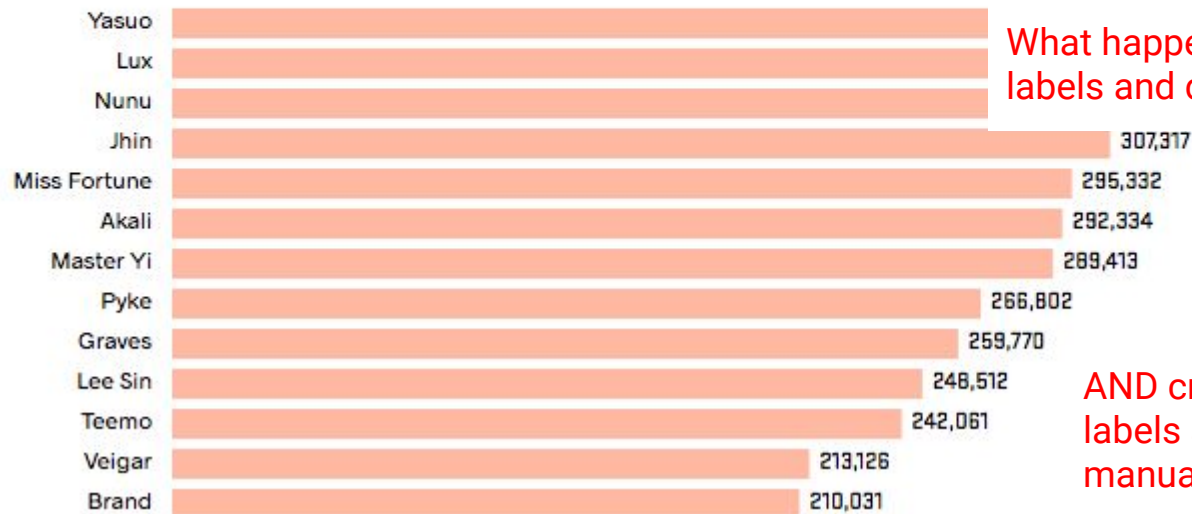
Bar colors

**What is JavaScript doing?** Defining how the page elements can change

## Nexus Blitz Play Rates

Filter: All classes

What happens when you click on a class



What happens to the bars, axis labels and data labels on a filter

AND creating the bars, axis labels and data labels (NOT manually in HTML)

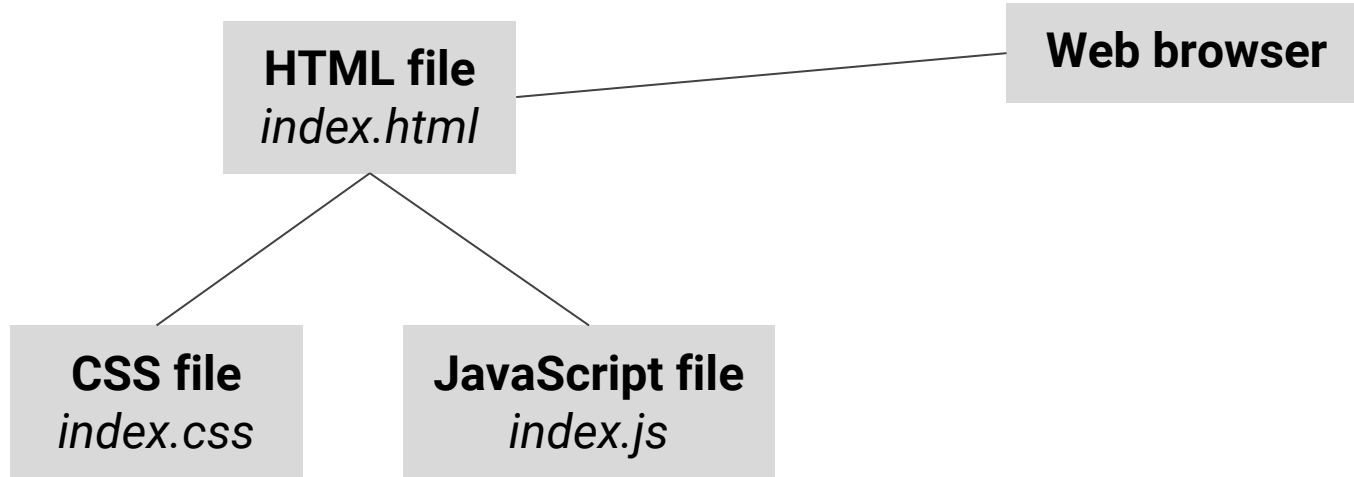
# Putting it all together

HTML file

CSS file

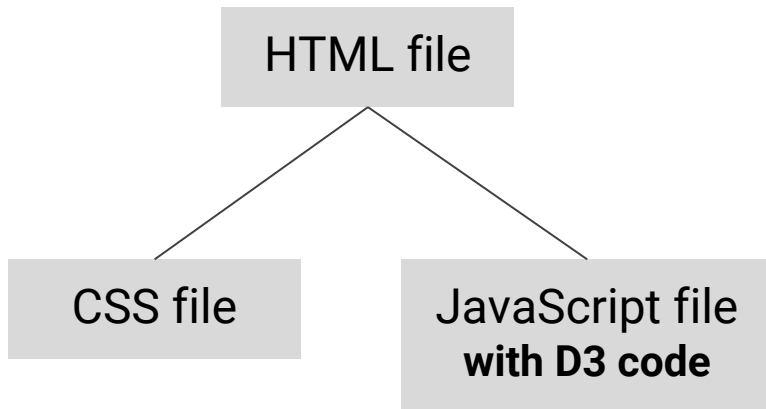
JavaScript file

# Putting it all together



# What about D3?

- D3 is a JavaScript library for loading data into a web page and generating visuals from that data

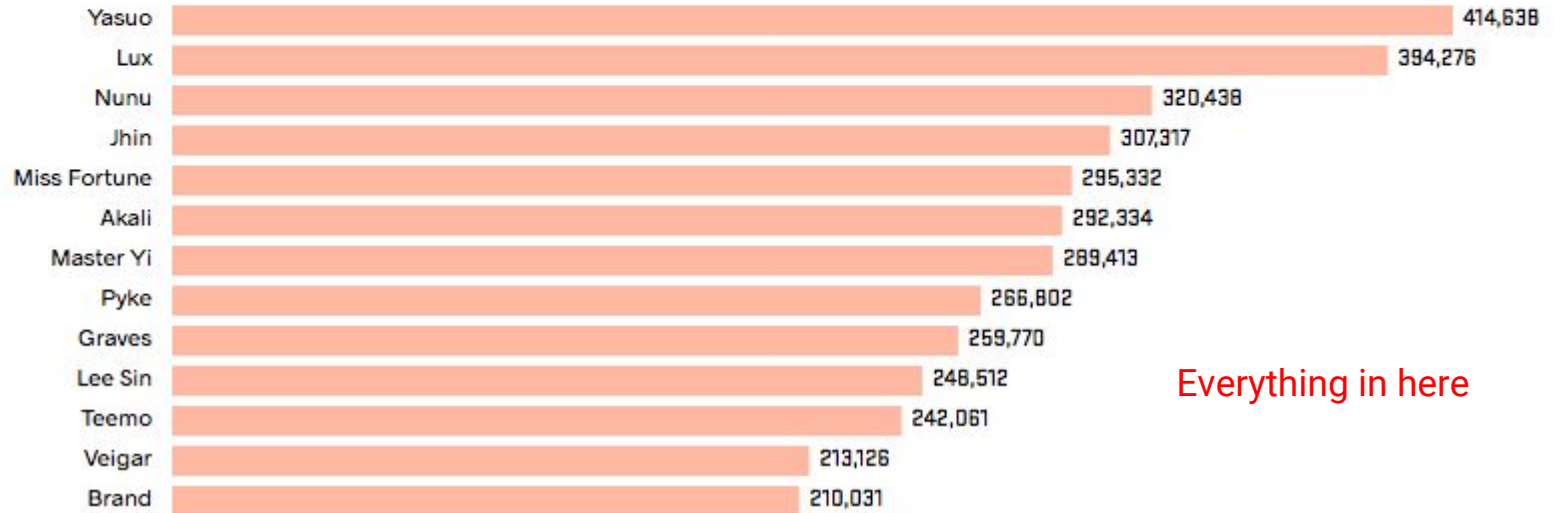




**What is D3 doing?** Everything in the chart area: creating the bars, axis labels, data labels AND manipulating all of these elements on a filter

## Nexus Blitz Play Rates

Filter: All classes



Everything in here

# Setup

# General file structure

- Download GitHub repo: <https://github.com/namisumida/d3-workshop>
- Open up the **template** folder
  - index.html
  - index.js
  - CSS (folder)
    - index.css

# Setting up a Python server

- Open a terminal window and navigate to the directory that you want served. For now, navigate to the ***template*** folder.
- For Python version 2.x, enter:

```
python -m SimpleHTTPServer 8888
```

- For Python version 3.x, enter:

```
python -m http.server 8888
```

- Switch to your web browser and enter the URL:

```
http://localhost:8888/
```

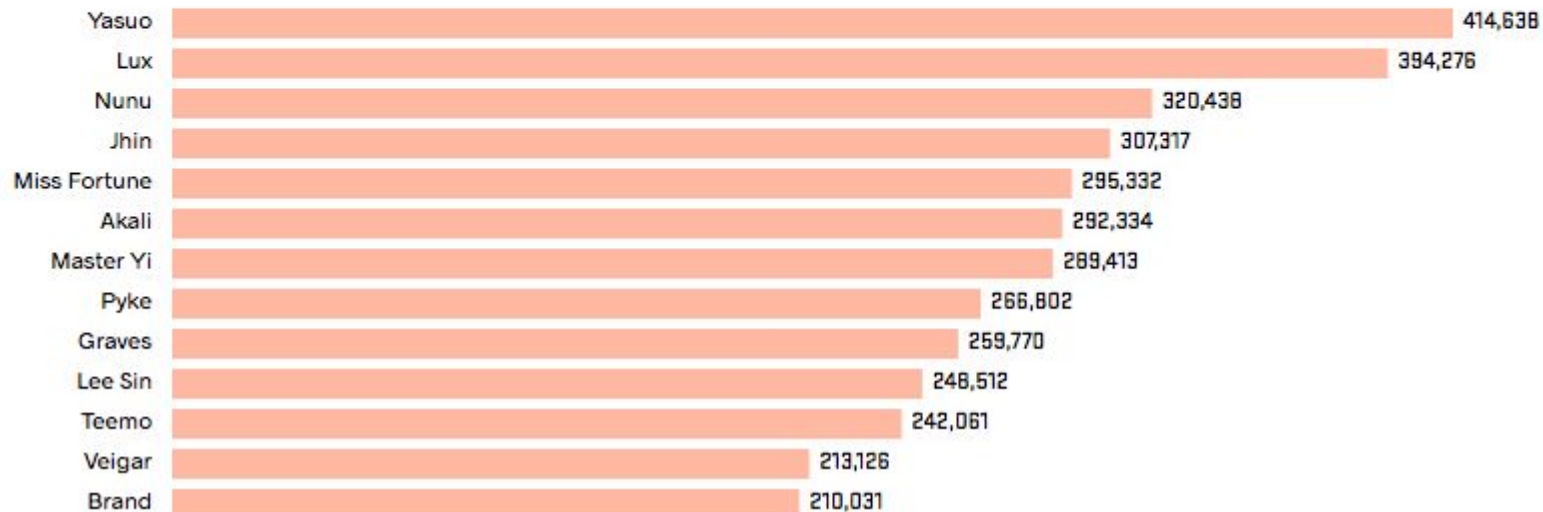
# Overview of how we'll create our data visualization

## Step 1: Create the structure of the page using HTML

### Nexus Blitz Play Rates

Filter:

All classes

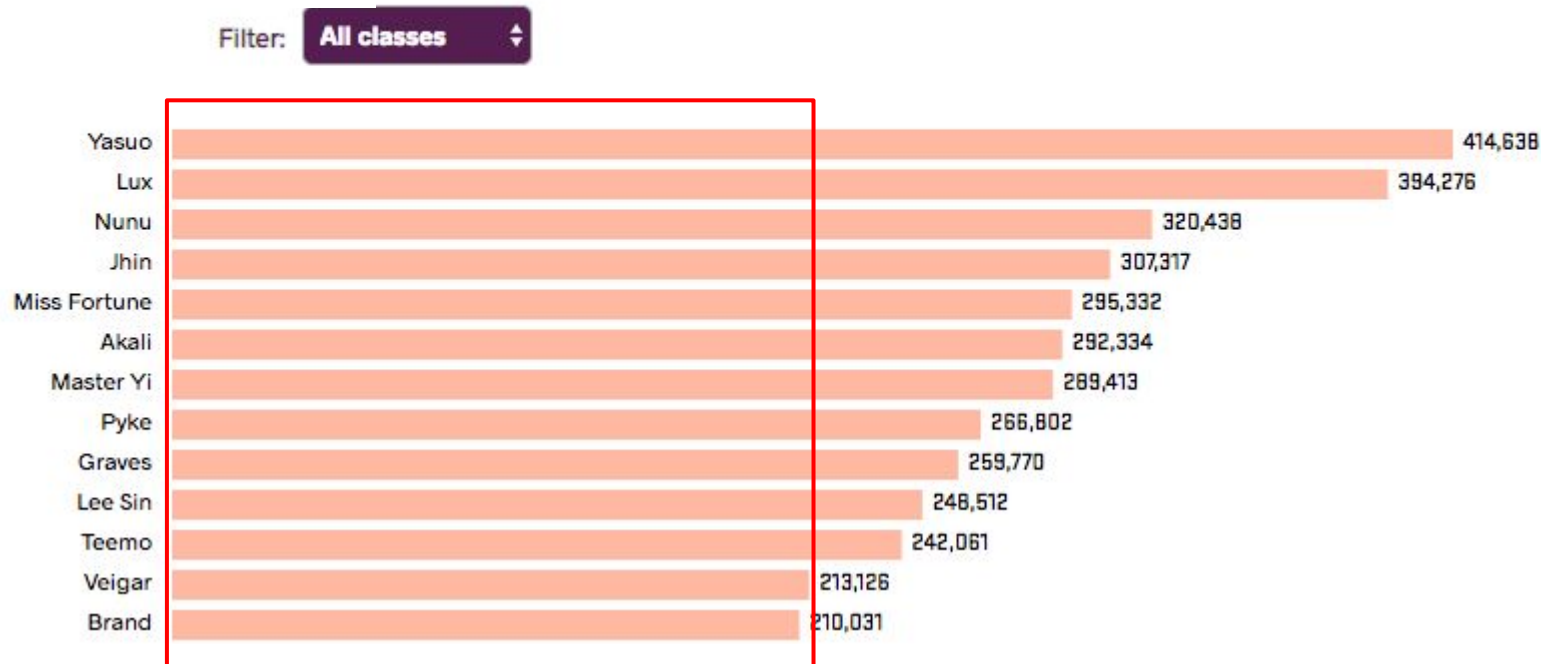


## Step 2: Create bars

For each data point...

- Create a rectangle
- Define x y position
- Define the height and width (based on data)

## Nexus Blitz Play Rates



### Step 3: Create axis labels

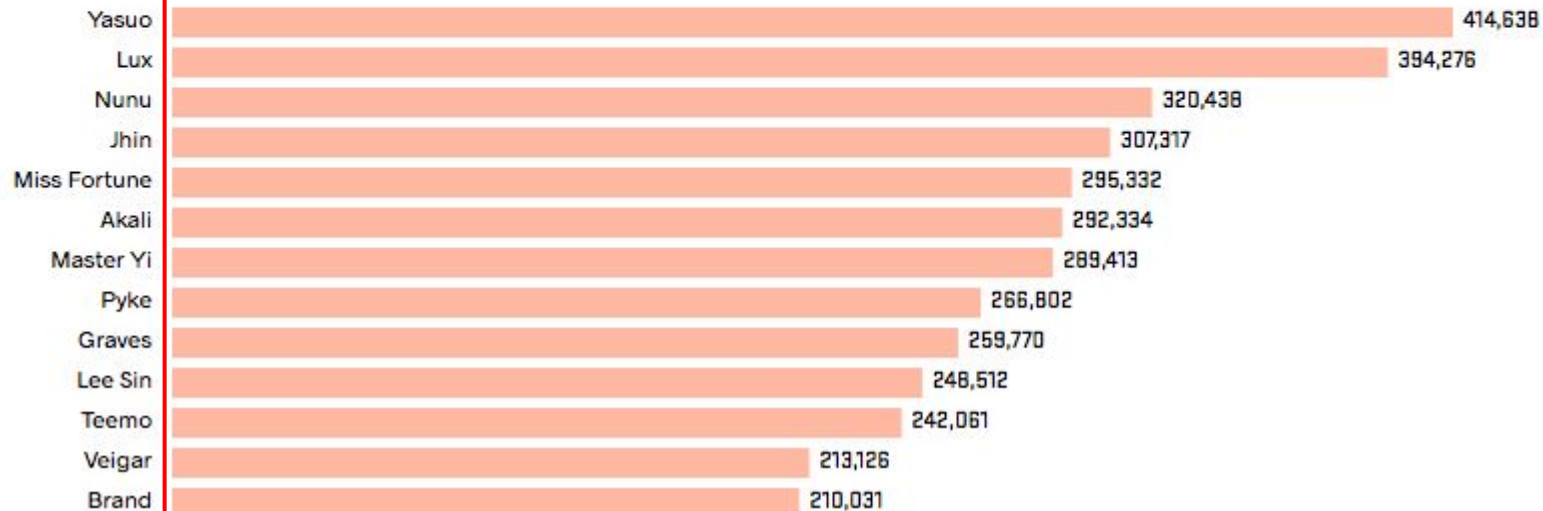
For each data point...

- Create a text element
- Define x y position
- Fill in the text element with champion name

## Nexus Blitz Play Rates

Filter:

All classes



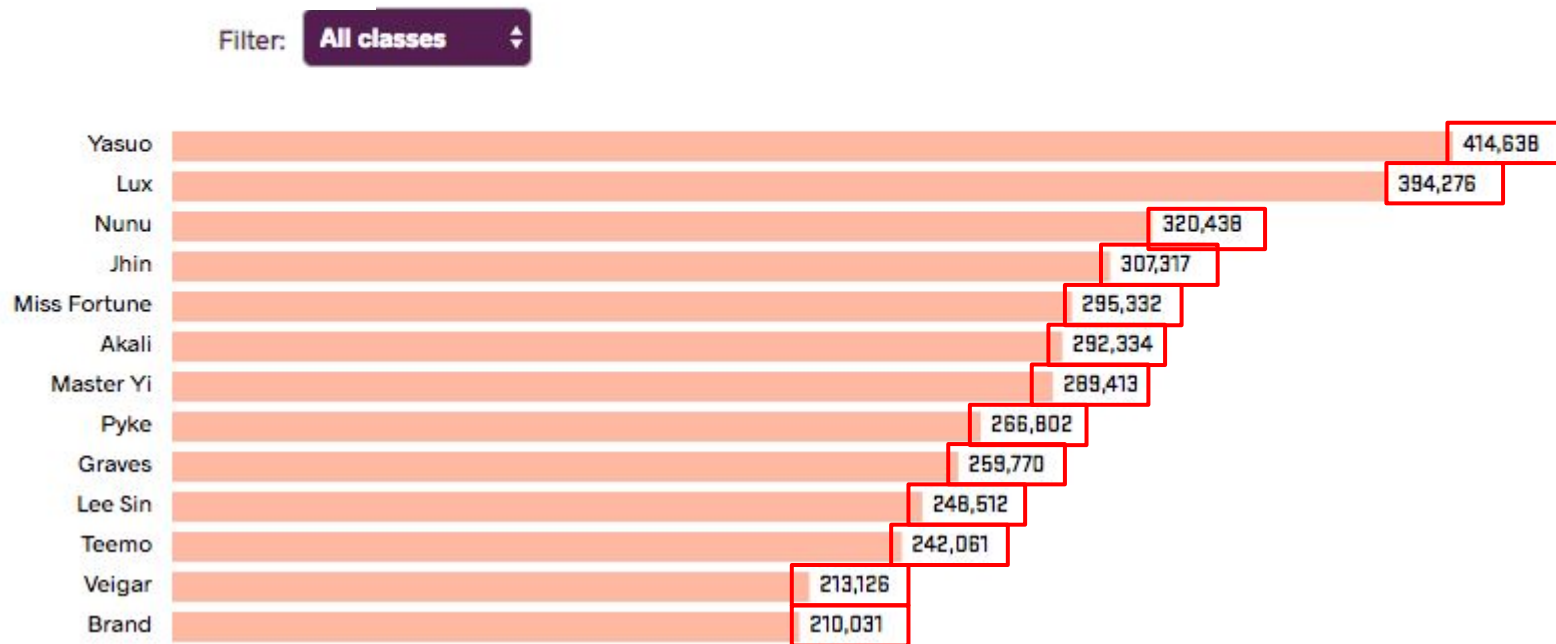


## Step 4: Create data labels

For each data point...

- Create a text element
- Define x y position
- Fill in the text element with play rate

## Nexus Blitz Play Rates



## Step 5: Filtering

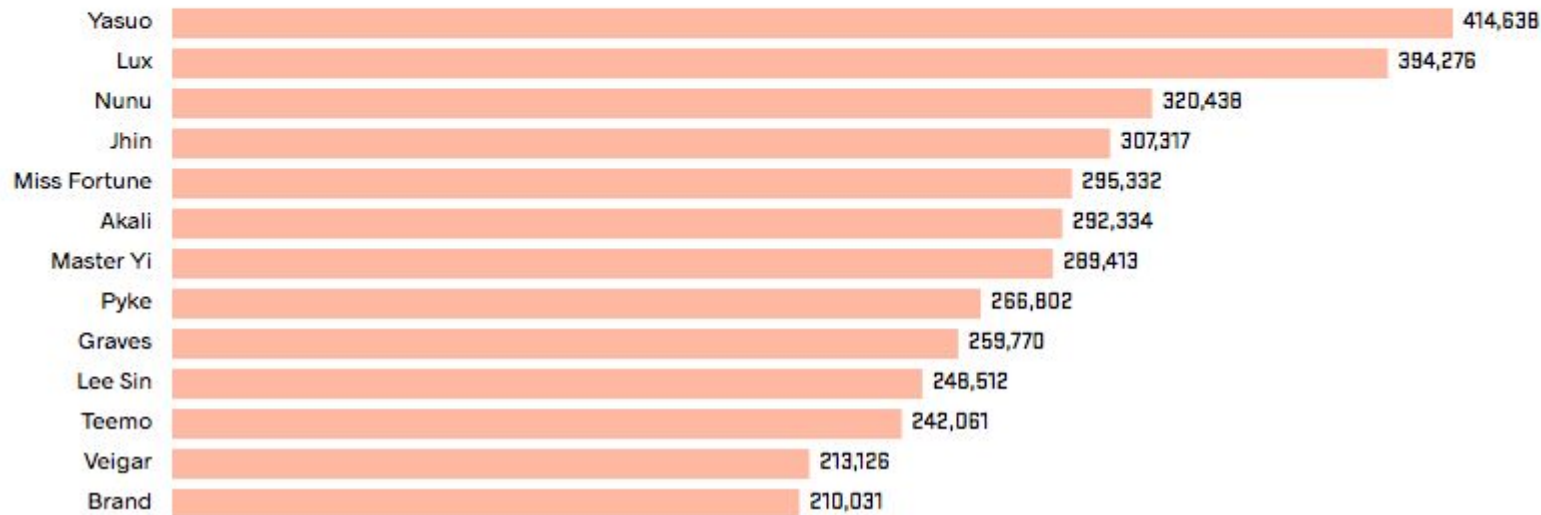
For each filter/class...

- Filter the dataset
- Redraw bars
- Redraw axis labels
- Redraw data labels

## Nexus Blitz Play Rates

Filter:

All classes



**Let's start coding!**

# What is an SVG?

- Scalable Vector Graphics
- Most often used with D3
- SVGs are better than normal *div* elements since they're more reliable, visually consistent and faster
- Think of it as a canvas for all your visuals

# Chaining methods

- Chain syntax which allows you to “chain” methods together with periods to perform several actions in a single line of code
- Type in the following code:

```
svg.append(“text”).text(“My first text element!”);
```

- **svg**: refers to and returns our SVG element. We set that element to a variable called `svg` in line 1 of our JavaScript file

# Chaining methods

- Chain syntax which allows you to “chain” methods together with periods to perform several actions in a single line of code
- Type in the following code:

```
svg.append(“text”).text(“My first text element!”);
```

- `.append()`: creates whatever new element you specify and appends it to your previous selection. It also hands off a reference to the new element it just created.

# Chaining methods

- Chain syntax which allows you to “chain” methods together with periods to perform several actions in a single line of code
- Type in the following code:

```
svg.append(“text”).text(“My first text element!”);
```

- `.text()`: takes a string and inserts it as the text that will be displayed in this text element

# Setting attributes

```
svg.append("text")  
    .text("My first text element!")  
    .attr("x", 30)  
    .attr("y", 10);
```

- `.attr()`: sets an HTML attribute and value on an element
- Here, we're setting the x position of the text element to 30 pixels and the y position to 30 pixels.
- On a webpage, the point (0,0) is the top-left corner of the page

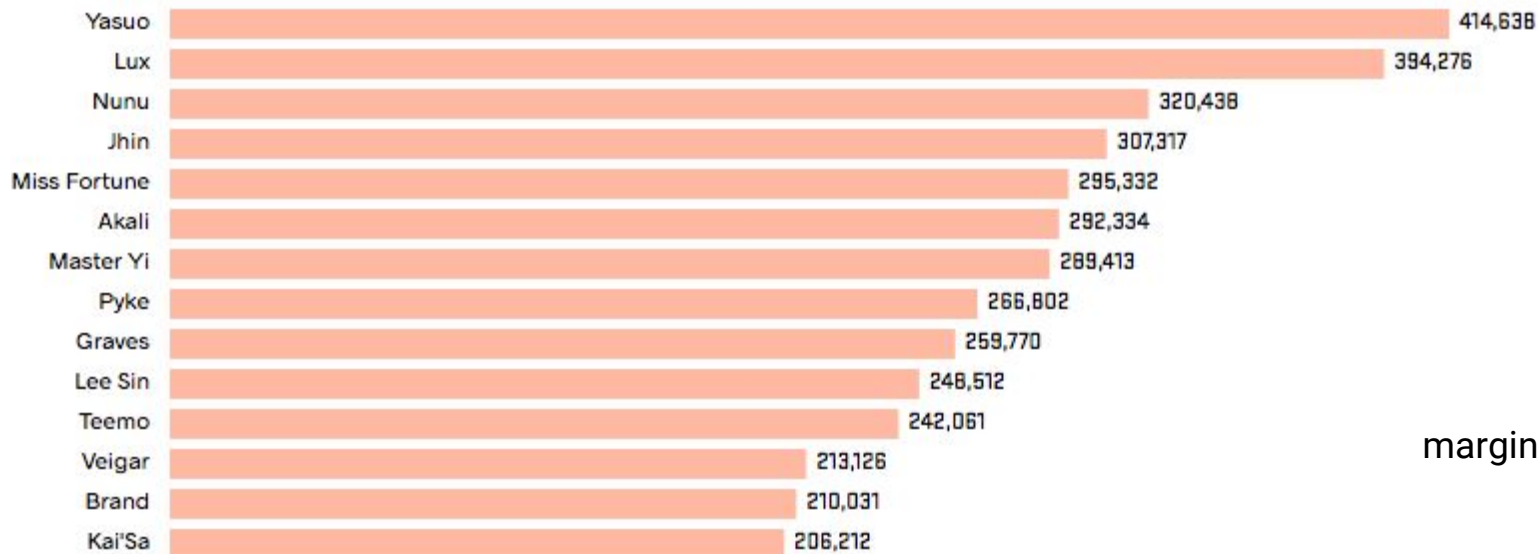


# Other attributes

- ID (unique name of an element)
- Class (group that an element belongs to)
- Width/height

# Planning the chart: margins

margin\_left



margin\_right

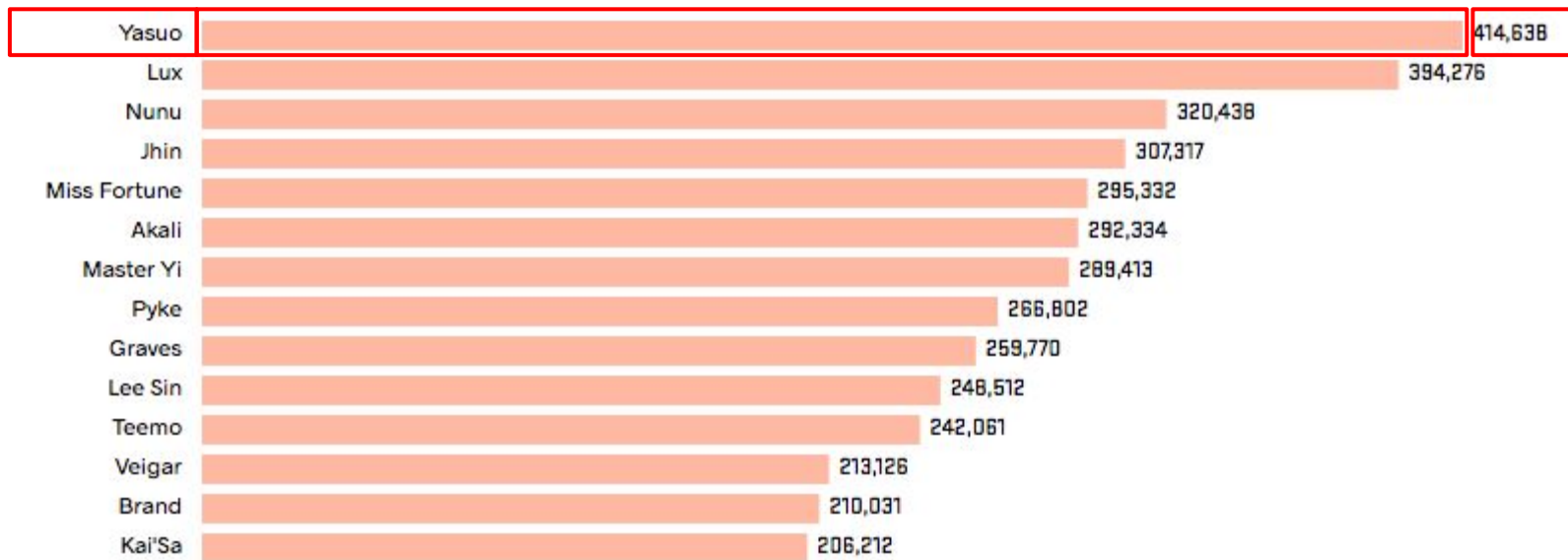
margin\_bottom

# Planning the chart: determining max width of elements

axisLabelWidth

maxBarWidth

dataLabelWidth



# Creating an x scale for bar widths

```
var xScale = d3.scaleLinear()  
              .domain([minDomain, maxDomain])  
              .range([minRange, maxRange]);
```

- A function that will take input values and output a range of values
- In our case, input values are our data values and output values are bar widths

# Creating an x scale for bar widths

```
var xScale = d3.scaleLinear()  
    .domain([minDomain, maxDomain])  
    .range([minRange, maxRange]);
```

- Your input values
- You'll typically use the minimum and maximum values of the data you're visualizing

# Creating an x scale for bar widths

```
var xScale = d3.scaleLinear()  
                .domain([minDomain, maxDomain])  
                .range([minRange, maxRange]);
```

- Output range
- For our bar chart, think of the smallest bar width you want to use and the widest bar width (maxBarWidth variable)

# Creating bars

```
svg.selectAll("myBars")  
  .data(dataset)  
  .enter()  
  .append("rect");
```

- Start with `svg` because that's where we want these bars to go into
- Selecting all elements on the page that are of type "myBars". None exist right now, but they will soon in the next line

# Creating bars

```
svg.selectAll("myBars")  
  .data(dataset)  
  .enter()  
  .append("rect");
```

- Parses data values and binds our elements with data values
- Everything past this point is going to be executed for each data value



# Creating bars

```
svg.selectAll("myBars")  
  .data(dataset)  
  .enter()  
  .append("rect");
```

- Creates new, data-bound elements
- This method looks at the “myBars” elements that are being selected, compares them to the dataset being called. If there are more data values than corresponding elements, then `enter()` creates new elements for them.

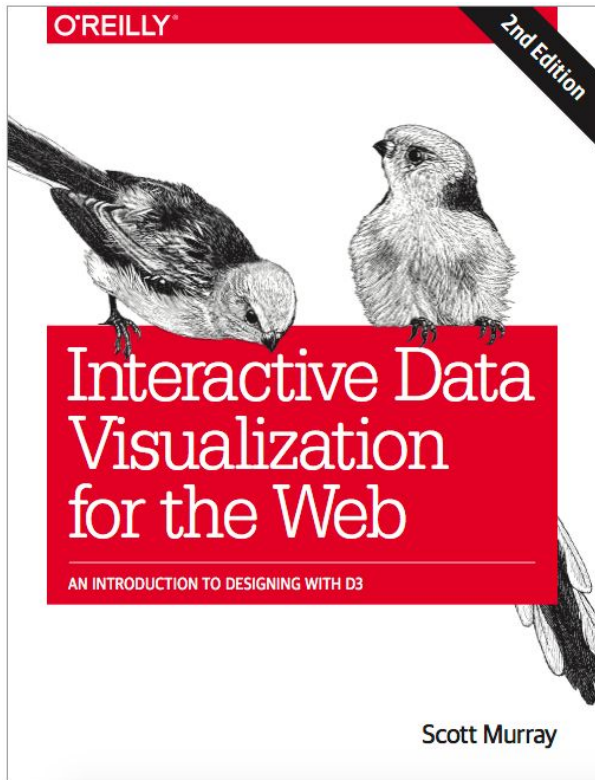
# Creating bars

```
svg.selectAll("myBars")  
  .data(dataset)  
  .enter()  
  .append("rect");
```

- Specifies what type of element you want appended
- In our case, we want rectangles

# Resources

# Resources



## For learning D3:

Interactive Data Visualization for the Web by Scott Murray

# Resources

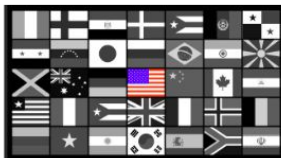
TOPICS  
Music  
Social Issues  
Entertainment  
People & Culture  
Sports



## The Sexualized Messages Dress Codes are Sending to Students

What we learned about sexualization from analyzing 481 high school dress codes

By AMBER THOMAS



## The World through the Eyes of the US

The countries that have preoccupied Americans since 1900

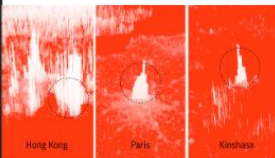
By RUSSELL GOLDENBERG



## A Brief History of the Past 100 Years

An analysis of 12 decades of New York Times headlines

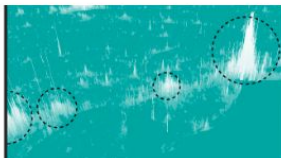
By ILIA BLINDERMAN, JAN DIEHM



## Population Mountains

This is a story about how to perceive the population size of cities.

By MATT DANIELS



## Human Terrain: Population in 3D

Visualizing the World's Population as a Terrain

By MATT DANIELS



## Tech Jobs may not Solve America's Looming Automation Crisis

Tech retraining programs are becoming more popular. They may not be a solution.

For inspiration & learning tricks/tools  
(once you've learned/mastered D3):

The Pudding ([www.pudding.cool](http://www.pudding.cool))