League of Legends Map Checker

By
Dean Garofalo
Ramin Ikhiilov
Billy Lin
Simon Yip

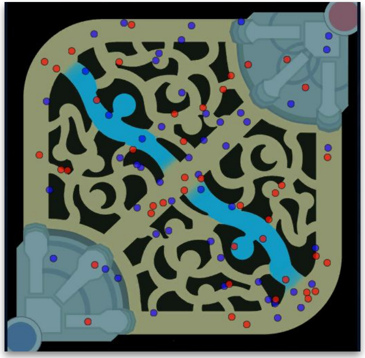
What is League of Legends?

- League of Legends is a multiplayer online battle arena video game developed and published by Riot Games for Microsoft Windows and macOS.

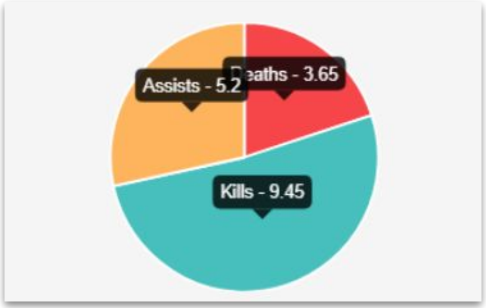
- In *League of Legends*, players assume the role of an unseen "summoner" that controls a "champion" with unique abilities and battle against a team of other players or computer-controlled champions. The goal is usually to destroy the opposing team's "nexus", a structure which lies at the heart of a base protected by defensive structures.

- In September 2016 the company estimated that there are over 100 million active players each month

So what are we doing?



- Illustrating useful statistics for players to view in a informative format
  - Heatmaps, Graphs, Kill/Death Ratios, Minion Kill rates



- Providing recent match history for personal reference

- The goal is to aid players and help them understand analytically their faults in map rotations.

Technology used

Coding using

- HTML/CSS for website design

- Javascript for the algorithms which power the website

- NodeJS for server side communication

- Riot API for retrieving data from LoL servers

Work Distribution

- Billy - HTML/CSS

- Dean - JavaScript Programming

- Ramin - JavaScript Programming

- Simon - Server side API /Database

Architecture

- The significant operating portion of our project relies on Javascript code which in essence fetches data from Riot and processes for display on our website

- The data we obtain from Riot is quite extensive so we spend a substantial amount of our algorithms finding our relevant data to display

- Once we find our data we either plot it in our heatmap, or display it in interactive graphs to display to the user

The Riot API - The Basics

- Riot stores substantial amounts of data from each user and provides all of it through their custom API

- Riot assigns developers API Keys which serve as a security measure

The Riot API - Getting Started

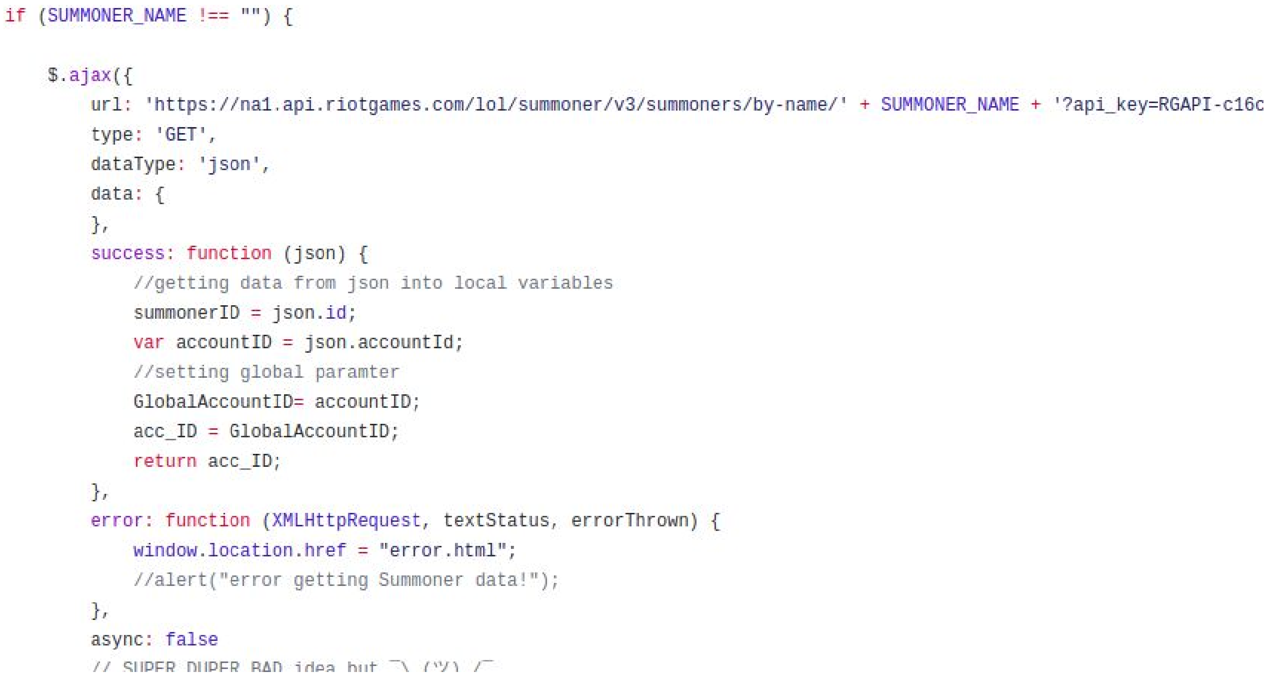
- Each developer receives a unique API key which is tied to their account and has certain limits
- These limits are, Expiration dates for keys, Rate limits, and Region limits
- We then applied for an advanced license which gives us far less limits, and are given out on a case by case basis
  - For example normal developer accounts are limited to 20 requests every 1 second, 100 requests every 2 minutes, and key expiration every 48 hours
- With our advanced license we can have higher rate limits and have non-expiring API keys

The Riot API - Structure

- There are 7 types of API calls Riot provides. Each retrieves specific information separated into relevant categories

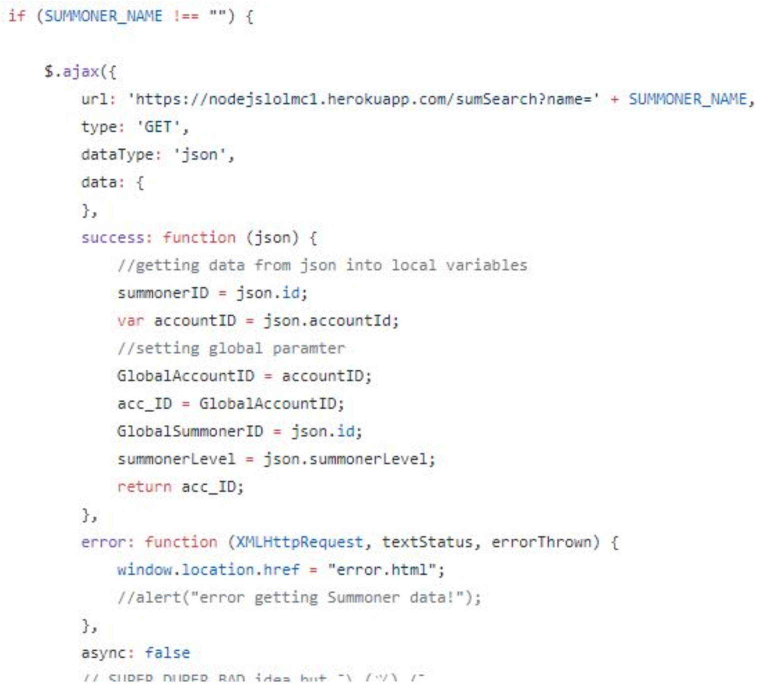
  CHAMPION-MASTERY- Mastery Lvl/ Experience with Champions
  CHAMPION- Retrieve all champions
  LEAGUE- Get leagues in all queues for a given summoner ID
  MATCH- Get match timeline by match ID
  SPECTATOR-Get current game information for the given summoner ID
  STATIC-DATA- Retrieves champion list, Retrieves item list, etc.
  SUMMONER- Get a user's Account info

Riot API - Making a API Call



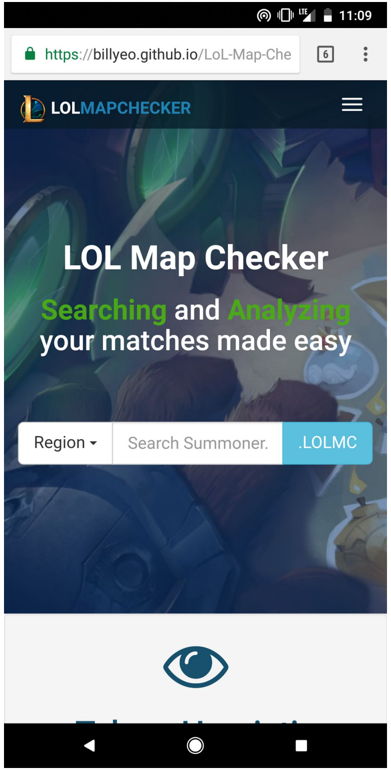
```
if (SUMMONER_NAME !== "") {

    $.ajax({
        url: 'https://na1.api.riotgames.com/lol/summoner/v3/summoners/by-name/' + SUMMONER_NAME + '?api_key=RGAPI-c16c
        type: 'GET',
        dataType: 'json',
        data: {
        },
        success: function (json) {
            //getting data from json into local variables
            summonerID = json.id;
            var accountID = json.accountId;
            //setting global paramter
            GlobalAccountID= accountID;
            acc_ID = GlobalAccountID;
            return acc_ID;
        },
        error: function (XMLHttpRequest, textStatus, errorThrown) {
            window.location.href = "error.html";
            //alert("error getting Summoner data!");
        },
        async: false
        // SUPER DUPER BAD idea but ¯\_(ツ)_/¯
```

Riot API - Making a API Call



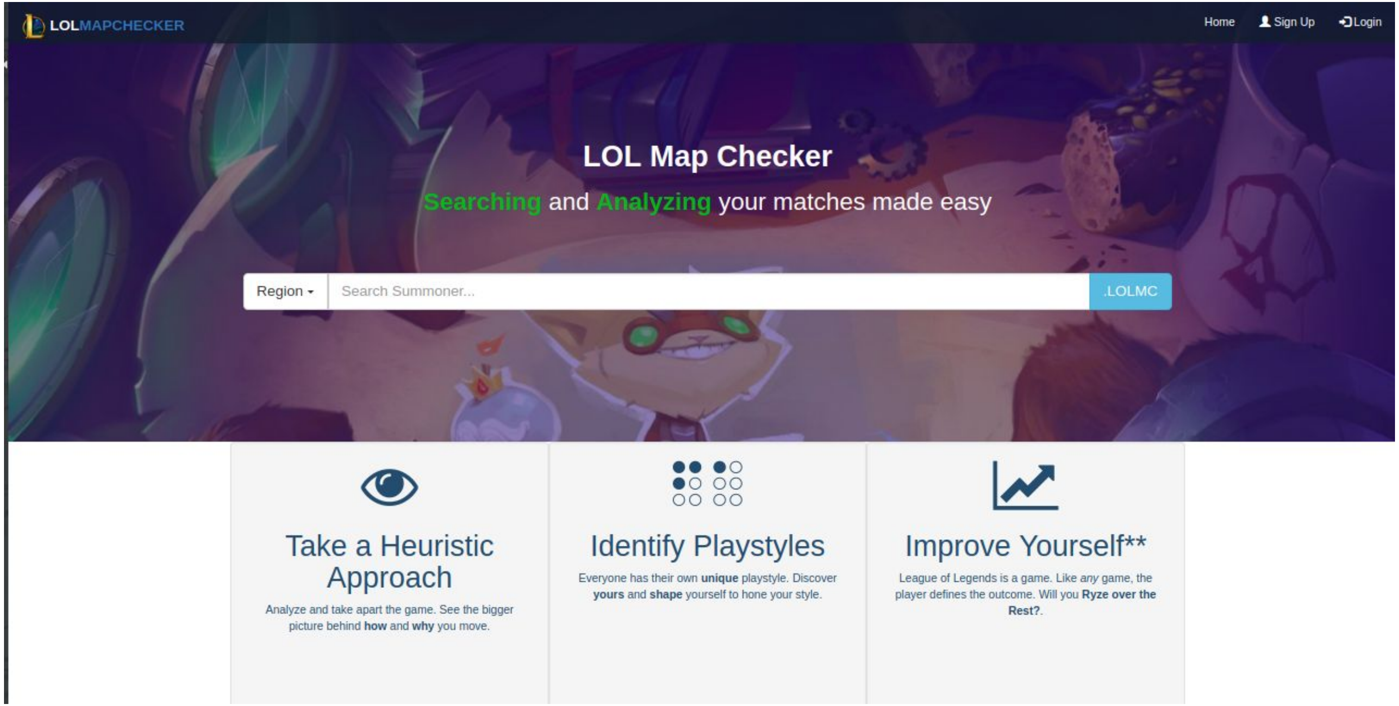
```
if (SUMMONER_NAME !== "") {

    $.ajax({
        url: 'https://nodejslolmc1.herokuapp.com/sumSearch?name=' + SUMMONER_NAME,
        type: 'GET',
        dataType: 'json',
        data: {
        },
        success: function (json) {
            //getting data from json into local variables
            summonerID = json.id;
            var accountID = json.accountId;
            //setting global paramter
            GlobalAccountID = accountID;
            acc_ID = GlobalAccountID;
            GlobalSummonerID = json.id;
            summonerLevel = json.summonerLevel;
            return acc_ID;
        },
        error: function (XMLHttpRequest, textStatus, errorThrown) {
            window.location.href = "error.html";
            //alert("error getting Summoner data!");
        },
        async: false
        // SUPER DUPER BAD idea but ¯\_(ツ)_/¯
```

Riot API - Benefits of server



LOL Map Checker

Searching and Analyzing your matches made easy



LOL Map Checker

Searching and Analyzing your matches made easy

Take a Heuristic Approach

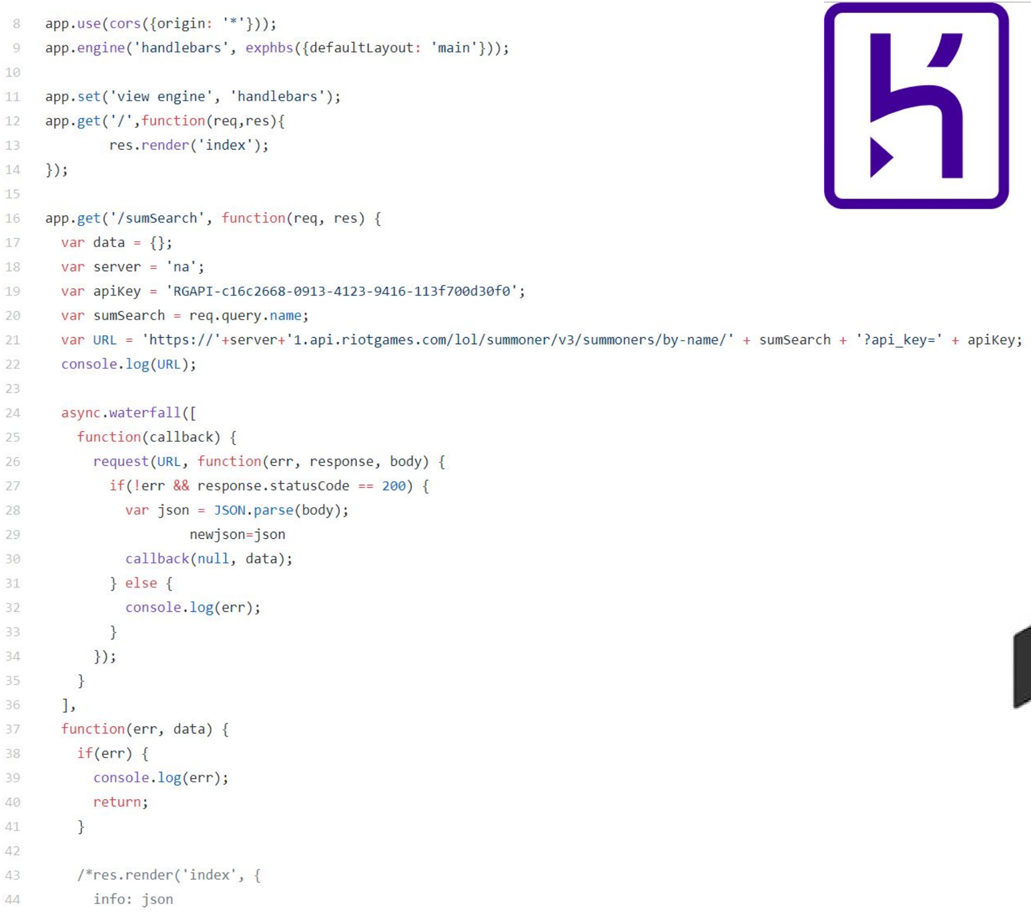
Analyze and take apart the game. See the bigger picture behind **how** and **why** you move.

Identify Playstyles

Everyone has their own **unique** playstyle. Discover **yours** and **shape** yourself to hone your style.

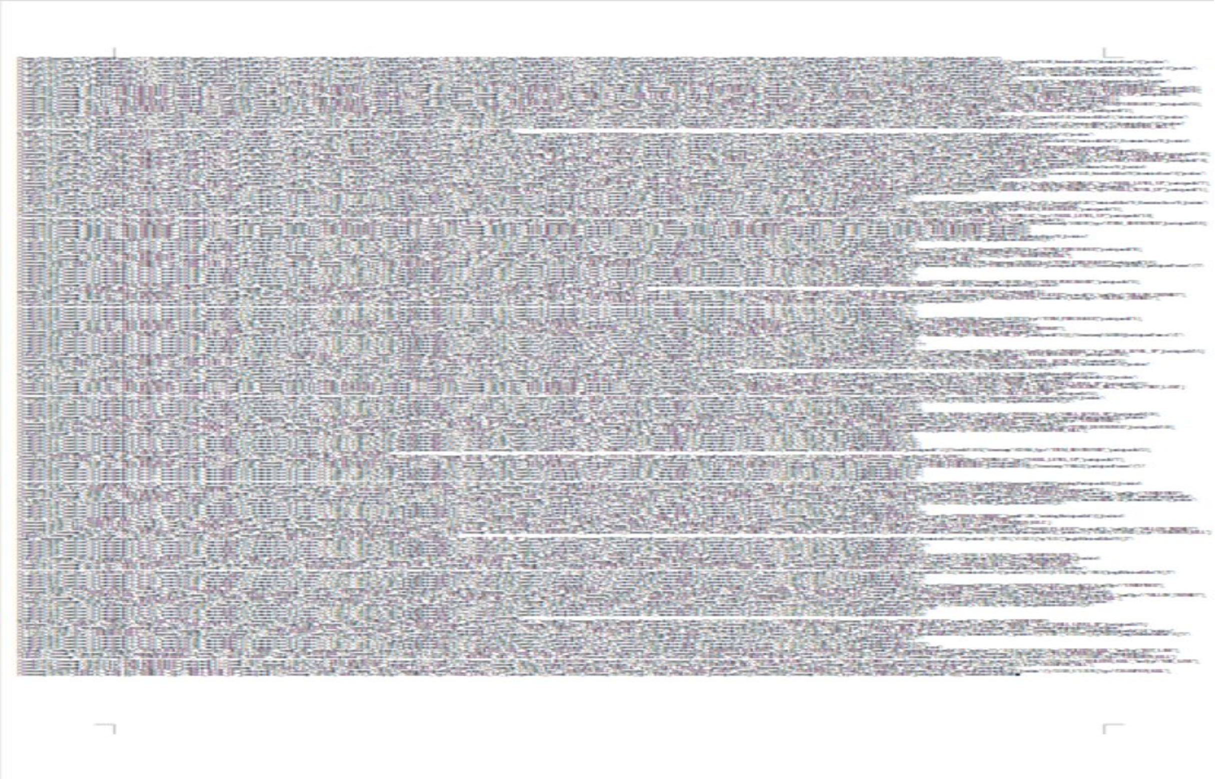
Improve Yourself**

League of Legends is a game. Like *any* game, the player defines the outcome. Will you **Ryze over the Rest?**.

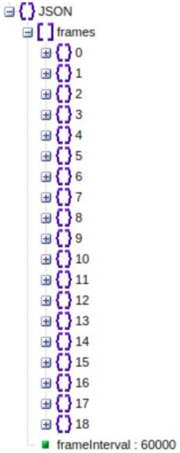


```
app.use(cors({origin: '*'}));
app.engine('handlebars', exphbs({defaultLayout: 'main'}));

app.set('view engine', 'handlebars');
app.get('/',function(req,res){
        res.render('index');
});

app.get('/sumSearch', function(req, res) {
  var data = {};
  var server = 'na';
  var apiKey = 'RGAPI-c16c2668-0913-4123-9416-113f700d30f0';
  var sumSearch = req.query.name;
  var URL = 'https://'+server+'1.api.riotgames.com/lol/summoner/v3/summoners/by-name/' + sumSearch + '?api_key=' + apiKey;
  console.log(URL);

  async.waterfall([
    function(callback) {
      request(URL, function(err, response, body) {
        if(!err && response.statusCode == 200) {
          var json = JSON.parse(body);
                  newjson=json
          callback(null, data);
        } else {
          console.log(err);
        }
      });
    }
  ],
  function(err, data) {
    if(err) {
      console.log(err);
      return;
    }

    /*res.render('index', {
      info: json
```

Riot API - JSON Data

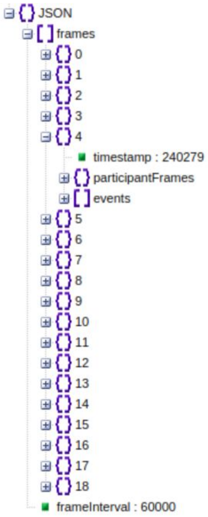


Riot API - JSON stuff continued



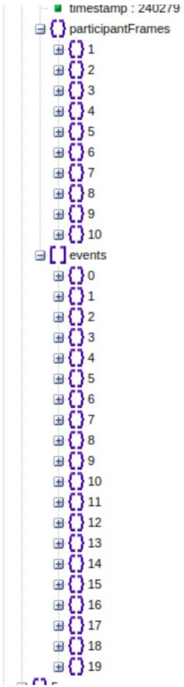
```
⊟ { } JSON
  ⊟ [ ] frames
    ⊞ { } 0
    ⊞ { } 1
    ⊞ { } 2
    ⊞ { } 3
    ⊞ { } 4
    ⊞ { } 5
    ⊞ { } 6
    ⊞ { } 7
    ⊞ { } 8
    ⊞ { } 9
    ⊞ { } 10
    ⊞ { } 11
    ⊞ { } 12
    ⊞ { } 13
    ⊞ { } 14
    ⊞ { } 15
    ⊞ { } 16
    ⊞ { } 17
    ⊞ { } 18
    ■ frameInterval : 60000
```

Riot API - JSON stuff continued



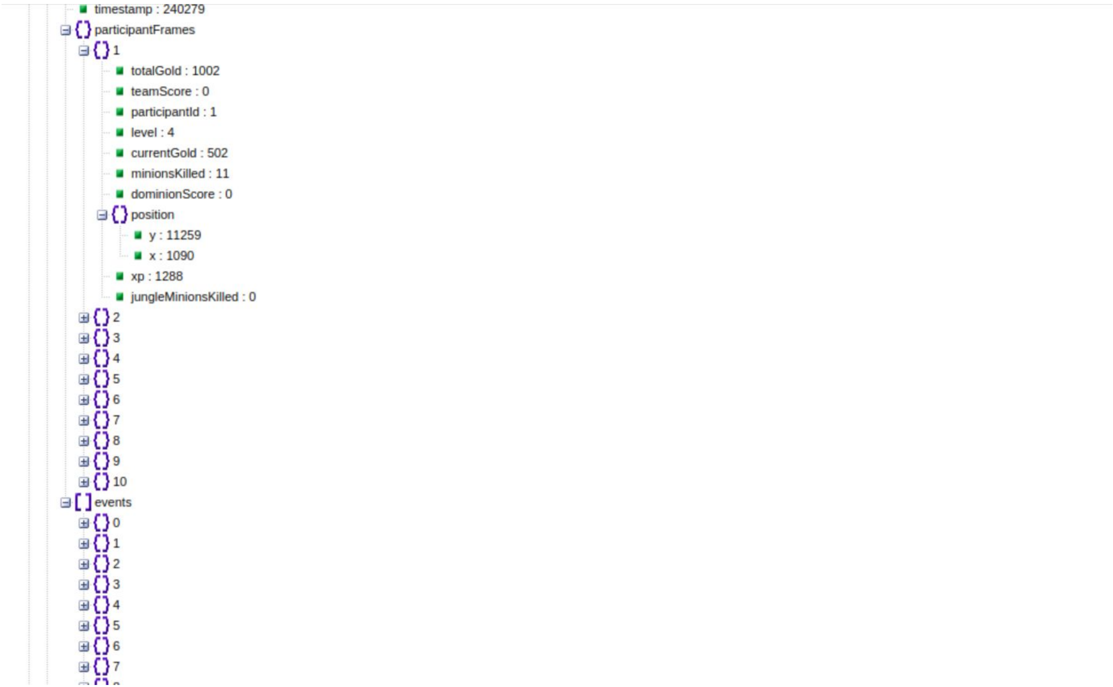
```
⊟ {} JSON
  ⊟ [] frames
    ⊞ {} 0
    ⊞ {} 1
    ⊞ {} 2
    ⊞ {} 3
    ⊟ {} 4
         ■ timestamp : 240279
       ⊞ {} participantFrames
       ⊞ [] events
    ⊞ {} 5
    ⊞ {} 6
    ⊞ {} 7
    ⊞ {} 8
    ⊞ {} 9
    ⊞ {} 10
    ⊞ {} 11
    ⊞ {} 12
    ⊞ {} 13
    ⊞ {} 14
    ⊞ {} 15
    ⊞ {} 16
    ⊞ {} 17
    ⊞ {} 18
    ■ frameInterval : 60000
```

Riot API - JSON stuff continued



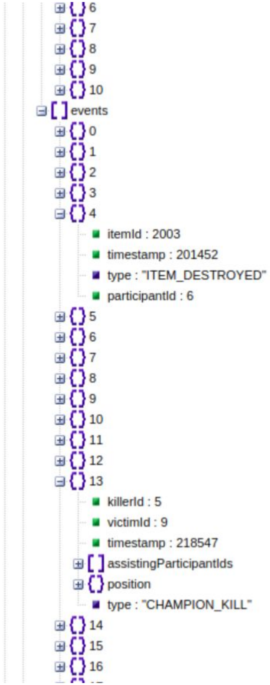
```
      timestamp : 240279
  ⊟ { } participantFrames
      ⊞ { } 1
      ⊞ { } 2
      ⊞ { } 3
      ⊞ { } 4
      ⊞ { } 5
      ⊞ { } 6
      ⊞ { } 7
      ⊞ { } 8
      ⊞ { } 9
      ⊞ { } 10
  ⊟ [ ] events
      ⊞ { } 0
      ⊞ { } 1
      ⊞ { } 2
      ⊞ { } 3
      ⊞ { } 4
      ⊞ { } 5
      ⊞ { } 6
      ⊞ { } 7
      ⊞ { } 8
      ⊞ { } 9
      ⊞ { } 10
      ⊞ { } 11
      ⊞ { } 12
      ⊞ { } 13
      ⊞ { } 14
      ⊞ { } 15
      ⊞ { } 16
      ⊞ { } 17
      ⊞ { } 18
      ⊞ { } 19
⊟ { } 5
```

Riot API - JSON stuff continued

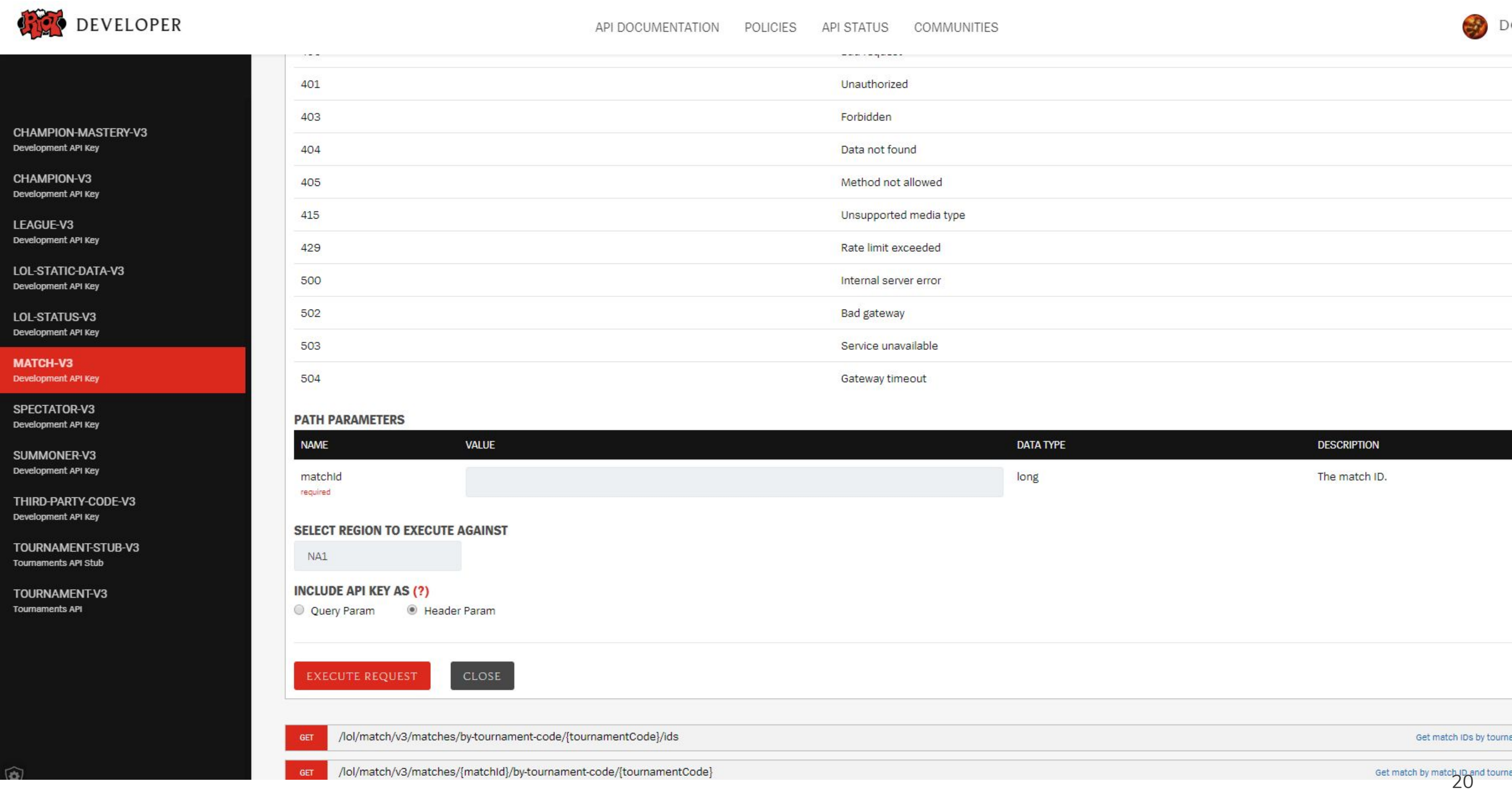


```
timestamp : 240279
{} participantFrames
  {} 1
     totalGold : 1002
     teamScore : 0
     participantId : 1
     level : 4
     currentGold : 502
     minionsKilled : 11
     dominionScore : 0
     {} position
        y : 11259
        x : 1090
     xp : 1288
     jungleMinionsKilled : 0
  {} 2
  {} 3
  {} 4
  {} 5
  {} 6
  {} 7
  {} 8
  {} 9
  {} 10
[] events
  {} 0
  {} 1
  {} 2
  {} 3
  {} 4
  {} 5
  {} 6
  {} 7
  {} 8
```

Riot API - JSON stuff continued



```
    ⊞ { } 6
    ⊞ { } 7
    ⊞ { } 8
    ⊞ { } 9
    ⊞ { } 10
  ⊟ [ ] events
    ⊞ { } 0
    ⊞ { } 1
    ⊞ { } 2
    ⊞ { } 3
    ⊟ { } 4
          itemId : 2003
          timestamp : 201452
          type : "ITEM_DESTROYED"
          participantId : 6
    ⊞ { } 5
    ⊞ { } 6
    ⊞ { } 7
    ⊞ { } 8
    ⊞ { } 9
    ⊞ { } 10
    ⊞ { } 11
    ⊞ { } 12
    ⊟ { } 13
          killerId : 5
          victimId : 9
          timestamp : 218547
      ⊞ [ ] assistingParticipantIds
      ⊞ { } position
          type : "CHAMPION_KILL"
    ⊞ { } 14
    ⊞ { } 15
    ⊞ { } 16
```



401	Unauthorized
403	Forbidden
404	Data not found
405	Method not allowed
415	Unsupported media type
429	Rate limit exceeded
500	Internal server error
502	Bad gateway
503	Service unavailable
504	Gateway timeout

PATH PARAMETERS

NAME	VALUE	DATA TYPE	DESCRIPTION
matchId required		long	The match ID.

SELECT REGION TO EXECUTE AGAINST

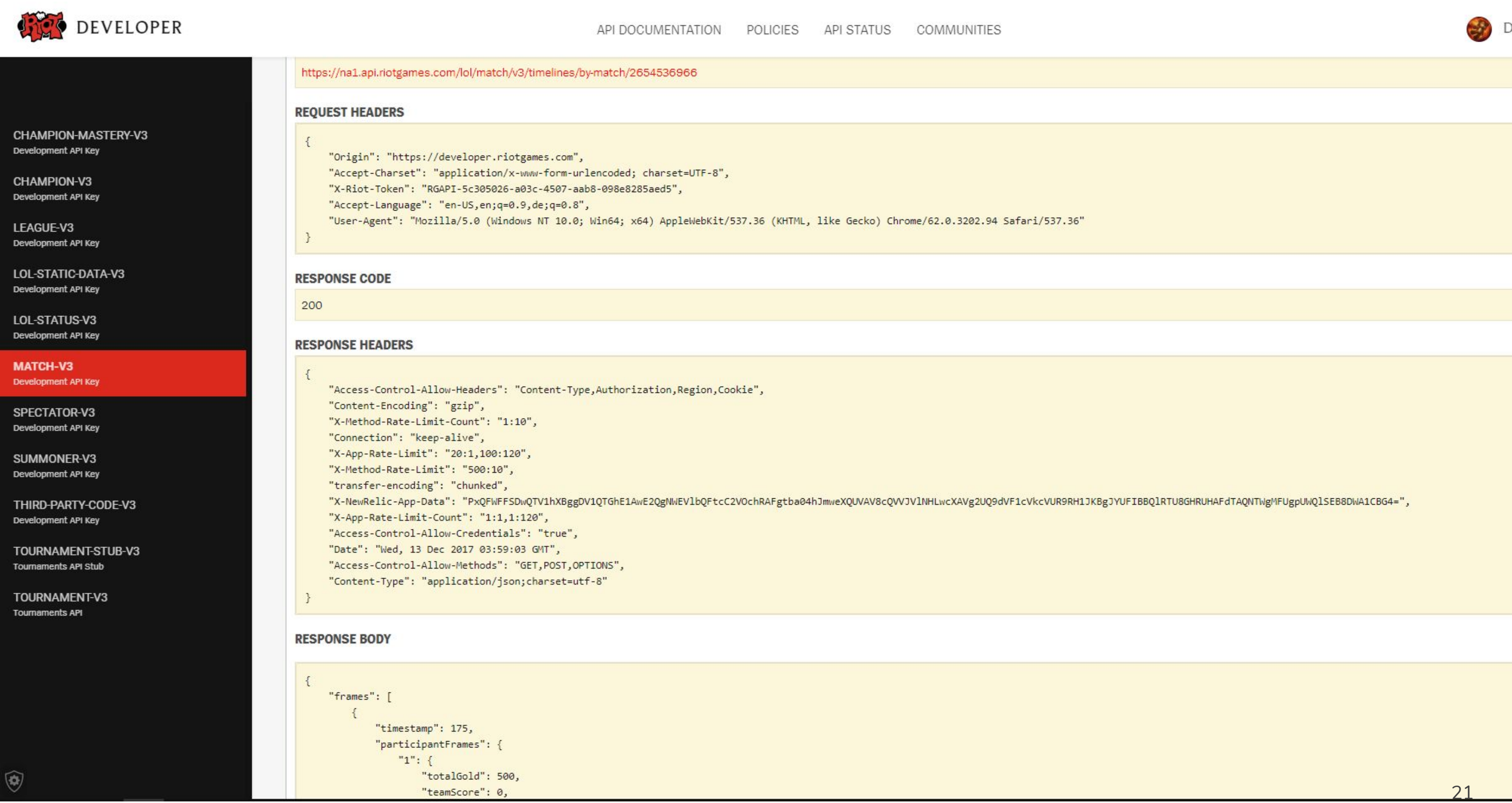
NA1

INCLUDE API KEY AS (?)

○ Query Param ● Header Param

GET /lol/match/v3/matches/by-tournament-code/{tournamentCode}/ids Get match IDs by tourne

GET /lol/match/v3/matches/{matchId}/by-tournament-code/{tournamentCode} Get match by match ID and tourne



https://na1.api.riotgames.com/lol/match/v3/timelines/by-match/2654536966

REQUEST HEADERS

```
{
    "Origin": "https://developer.riotgames.com",
    "Accept-Charset": "application/x-www-form-urlencoded; charset=UTF-8",
    "X-Riot-Token": "RGAPI-5c305026-a03c-4507-aab8-098e8285aed5",
    "Accept-Language": "en-US,en;q=0.9,de;q=0.8",
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/62.0.3202.94 Safari/537.36"
}
```

RESPONSE CODE

200

RESPONSE HEADERS

```
{
    "Access-Control-Allow-Headers": "Content-Type,Authorization,Region,Cookie",
    "Content-Encoding": "gzip",
    "X-Method-Rate-Limit-Count": "1:10",
    "Connection": "keep-alive",
    "X-App-Rate-Limit": "20:1,100:120",
    "X-Method-Rate-Limit": "500:10",
    "transfer-encoding": "chunked",
    "X-NewRelic-App-Data": "PxQFWFFSDwQTV1hXBggDV1QTGhE1AwE2QgNWEVlbQFtcC2VOchRAFgtba04hJmweXQUVAV8cQVVJVlNHLwcXAVg2UQ9dVF1cVkcVUR9RH1JKBgJYUFIBBQlRTU8GHRUHAFdTAQNTWgMFUgpUWQlSEB8DWA1CBG4=",
    "X-App-Rate-Limit-Count": "1:1,1:120",
    "Access-Control-Allow-Credentials": "true",
    "Date": "Wed, 13 Dec 2017 03:59:03 GMT",
    "Access-Control-Allow-Methods": "GET,POST,OPTIONS",
    "Content-Type": "application/json;charset=utf-8"
}
```

RESPONSE BODY

```
{
    "frames": [
        {
            "timestamp": 175,
            "participantFrames": {
                "1": {
                    "totalGold": 500,
                    "teamScore": 0,
```

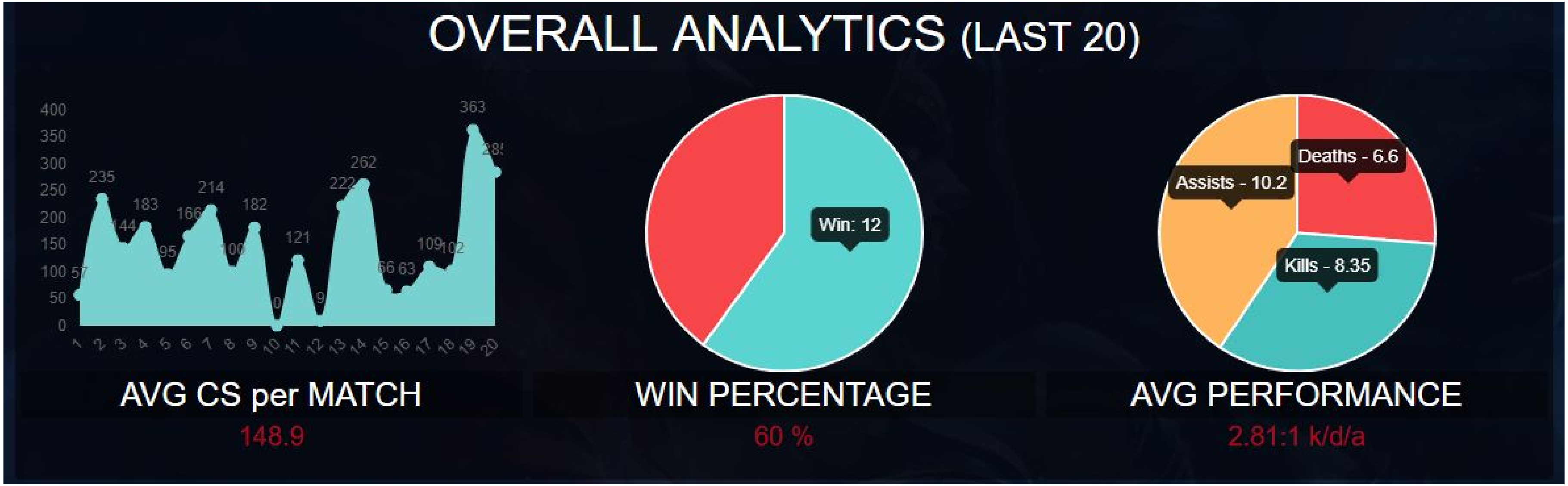
Chart.js and D3.js Libraries

- Chart.js allows developers to easily create JavaScript charts

- All that's required is the script included in your page along with a single <canvas> node to render the chart.

- Type of charts include: Line, Bar, Radar, Doughnut & Piee, Polar Area, Bubble, Scatter, Area, mixed, and more.

How did we use Chart.js?



OVERALL ANALYTICS (LAST 20)

AVG CS per MATCH

148.9

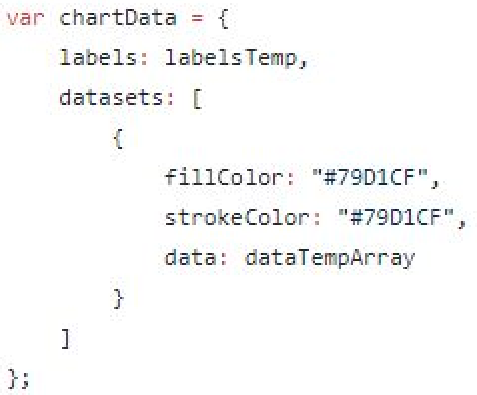
WIN PERCENTAGE

60 %

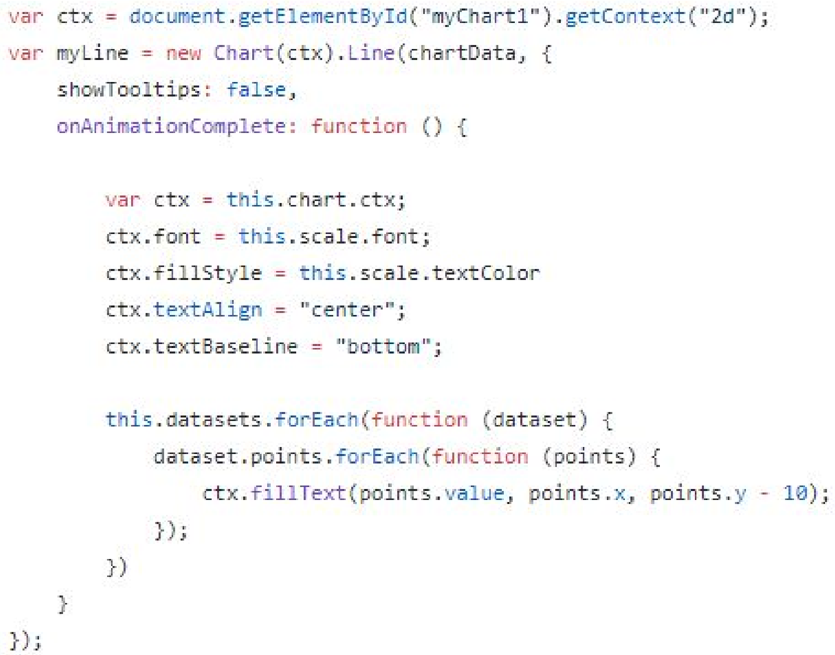
AVG PERFORMANCE

2.81:1 k/d/a

Code behind Line Graph

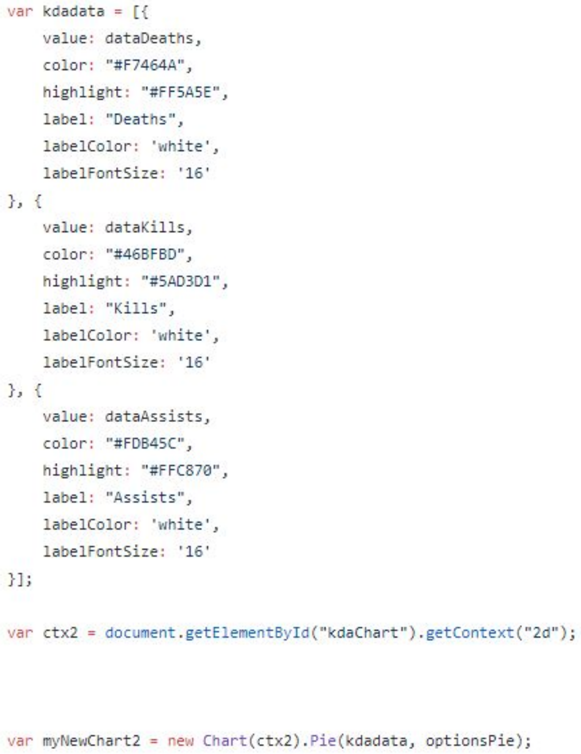


```
var chartData = {
    labels: labelsTemp,
    datasets: [
        {
            fillColor: "#79D1CF",
            strokeColor: "#79D1CF",
            data: dataTempArray
        }
    ]
};
```

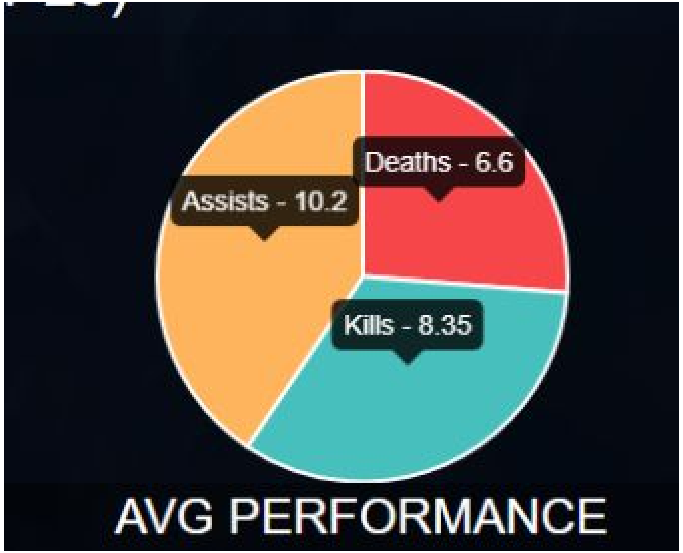


```
var ctx = document.getElementById("myChart1").getContext("2d");
var myLine = new Chart(ctx).Line(chartData, {
    showTooltips: false,
    onAnimationComplete: function () {

        var ctx = this.chart.ctx;
        ctx.font = this.scale.font;
        ctx.fillStyle = this.scale.textColor
        ctx.textAlign = "center";
        ctx.textBaseline = "bottom";

        this.datasets.forEach(function (dataset) {
            dataset.points.forEach(function (points) {
                ctx.fillText(points.value, points.x, points.y - 10);
            });
        })
    }
});
```

Code Behind Pie Chart



```
var kdadata = [{
    value: dataDeaths,
    color: "#F7464A",
    highlight: "#FF5A5E",
    label: "Deaths",
    labelColor: 'white',
    labelFontSize: '16'
}, {
    value: dataKills,
    color: "#46BFBD",
    highlight: "#5AD3D1",
    label: "Kills",
    labelColor: 'white',
    labelFontSize: '16'
}, {
    value: dataAssists,
    color: "#FDB45C",
    highlight: "#FFC870",
    label: "Assists",
    labelColor: 'white',
    labelFontSize: '16'
}];

var ctx2 = document.getElementById("kdaChart").getContext("2d");



var myNewChart2 = new Chart(ctx2).Pie(kdadata, optionsPie);
```



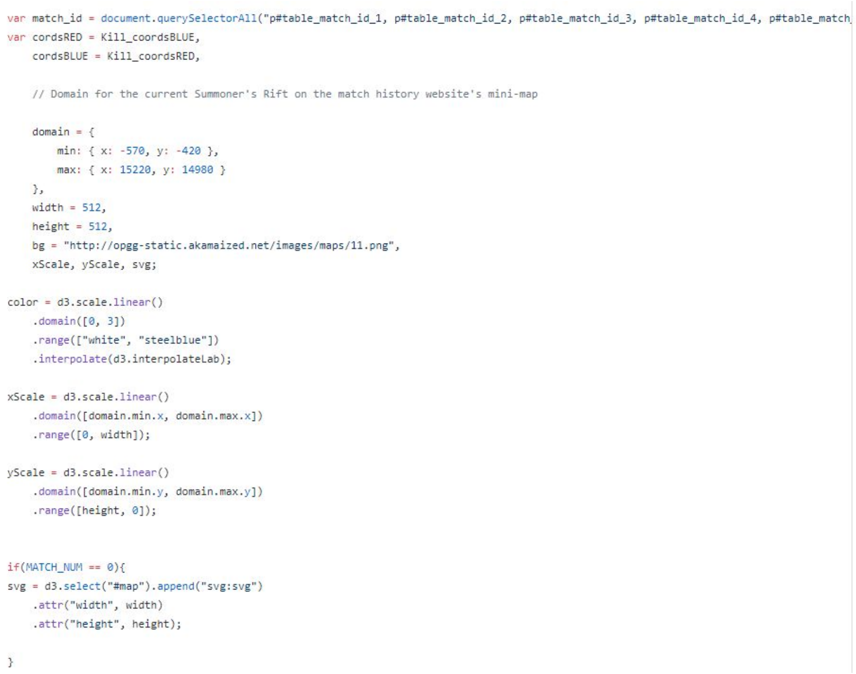
What is D3.js?

- D3.js is a JavaScript library for manipulating documents based on data.

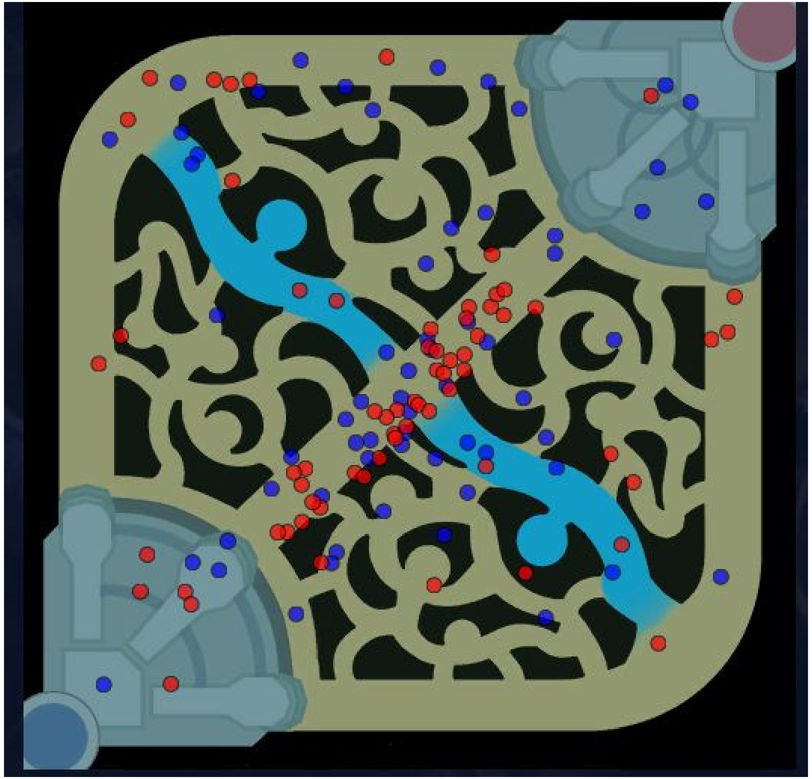
- D3 helps you bring data to life using HTML, SVG, and CSS.

- D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document.
  - For example, you can use D3 to generate an HTML table from an array of numbers

How did we use D3.js?



```
var match_id = document.querySelectorAll("p#table_match_id_1, p#table_match_id_2, p#table_match_id_3, p#table_match_id_4, p#table_match
var cordsRED = Kill_coordsBLUE,
    cordsBLUE = Kill_coordsRED,

    // Domain for the current Summoner's Rift on the match history website's mini-map

    domain = {
        min: { x: -570, y: -420 },
        max: { x: 15220, y: 14980 }
    },
    width = 512,
    height = 512,
    bg = "http://opgg-static.akamaized.net/images/maps/11.png",
    xScale, yScale, svg;

color = d3.scale.linear()
    .domain([0, 3])
    .range(["white", "steelblue"])
    .interpolate(d3.interpolateLab);

xScale = d3.scale.linear()
    .domain([domain.min.x, domain.max.x])
    .range([0, width]);

yScale = d3.scale.linear()
    .domain([domain.min.y, domain.max.y])
    .range([height, 0]);


if(MATCH_NUM == 0){
svg = d3.select("#map").append("svg:svg")
    .attr("width", width)
    .attr("height", height);

}
```



D3.js Examples

https://bl.ocks.org/mbostock/7280327

http://ghv.artzub.com/#repo=LoL-Map-Checker.github.io&climit=500&user=billyeo

How can other developers use this?

- Make visual representation of large complex data sets look simple

- Showcase animations to convey data as a living object instead of numerical statistics

- Visual representation is easier for humans to quantify and reason with versus spreadsheets and lists of data

Final Reflections

- Good free web hosting is hard to find
- Massive amounts of data limits our flexibility
- Reliability needs to be at the beginning of development architecture

https://billyeo.github.io/LoL-Map-Checker.github.io/