
COMP 431/531: Web Development

Lecture 7: Modern JavaScript

Mack Joyner (mjoyner@rice.edu)

<https://www.clear.rice.edu/comp431>

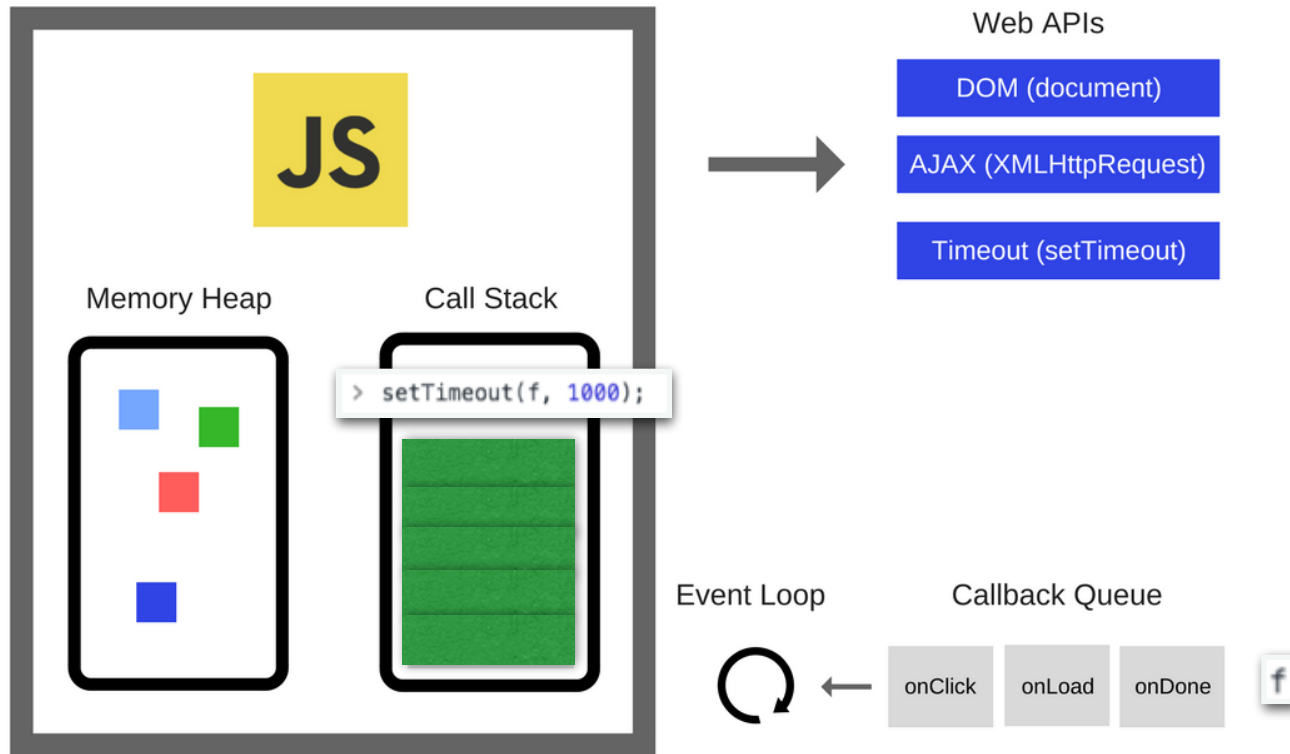


Announcements & Reminders

- Quiz #1 (JavaScript) is due Thursday, Sept. 18th at 11:59pm
- HW #3 (JavaScript Game) is due Thursday, Sept. 25th at 11:59pm
classroom hw3 repo: <https://classroom.github.com/a/wZT10Yqv>

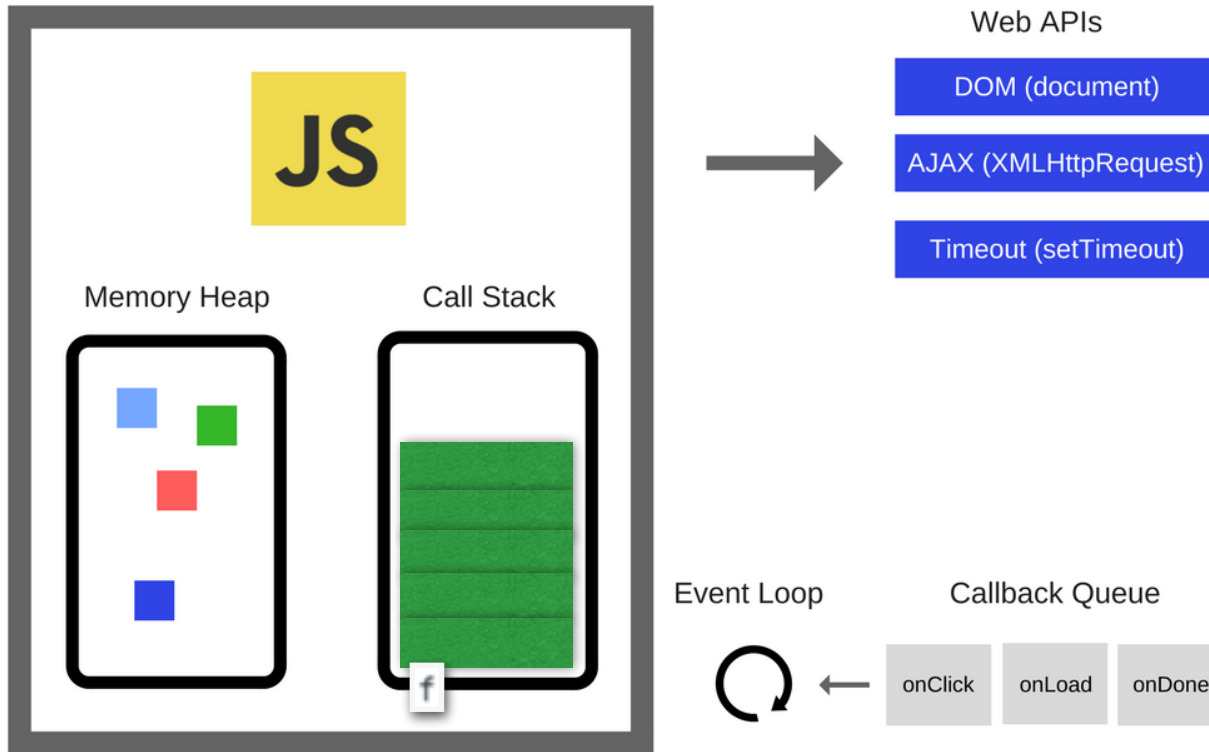


How does JavaScript work?



Source: <https://blog.sessionstack.com/how-does-javascript-actually-work-part-1-b0bacc073cf>

How does JavaScript work?



Source: <https://blog.sessionstack.com/how-does-javascript-actually-work-part-1-b0bacc073cf>

jQuery Callback vs Chaining

```
$("#div3").click(function() {  
    $(this)  
        .animate( {opacity: 0}, 2500)  
        .animate( {opacity: 1, fontSize: '1em' }, 500 )  
        .hide(1000, function() {  
            $(this).css({ backgroundColor: "blue" })  
        })  
        .show(1000)  
        .animate( { fontSize: '2em' }, function() {  
            $(this).css({ backgroundColor: "green" })  
        } )  
    })
```



ECMAScript

<div> <div>COMPAT</div> <div>ES</div> </div> <div>ECMAScript</div> <div>5</div> <div>6</div> <div>2016+</div> <div>next</div> <div>intl</div> <div>non-standard</div>																			
		Compilers/polyfills					Desktop browsers												
100%		61%	44%	58%	6%	1%	66%	87%	100%	100%	100%	100%	87%	100%	100%	100%	33%	87%	87%
Current browser		Babel 7 ± core-js 3	Closure 2020.05	Type-Script ± core-js 3	es7-shim	IE 11	FF 68 ESR	FF 78 ESR	FF 79	FF 80	FF 81 Beta	FF 82 Nightly	CH 84	CH 85	CH 86	CH 87	Edge 18	Edge 83	Edge 84
3/3		3/3	3/3	2/3	0/3	0/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
3/3		3/3	2/3	3/3	2/3	0/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
Yes		?	?	?	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		?	Yes	Yes ^[9]	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		?	?	?	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		Yes	Yes	Yes	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		Yes	Yes	Yes	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		?	?	?	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes		?	?	?	?	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

<https://compat-table.github.io/compat-table/es2016plus/>

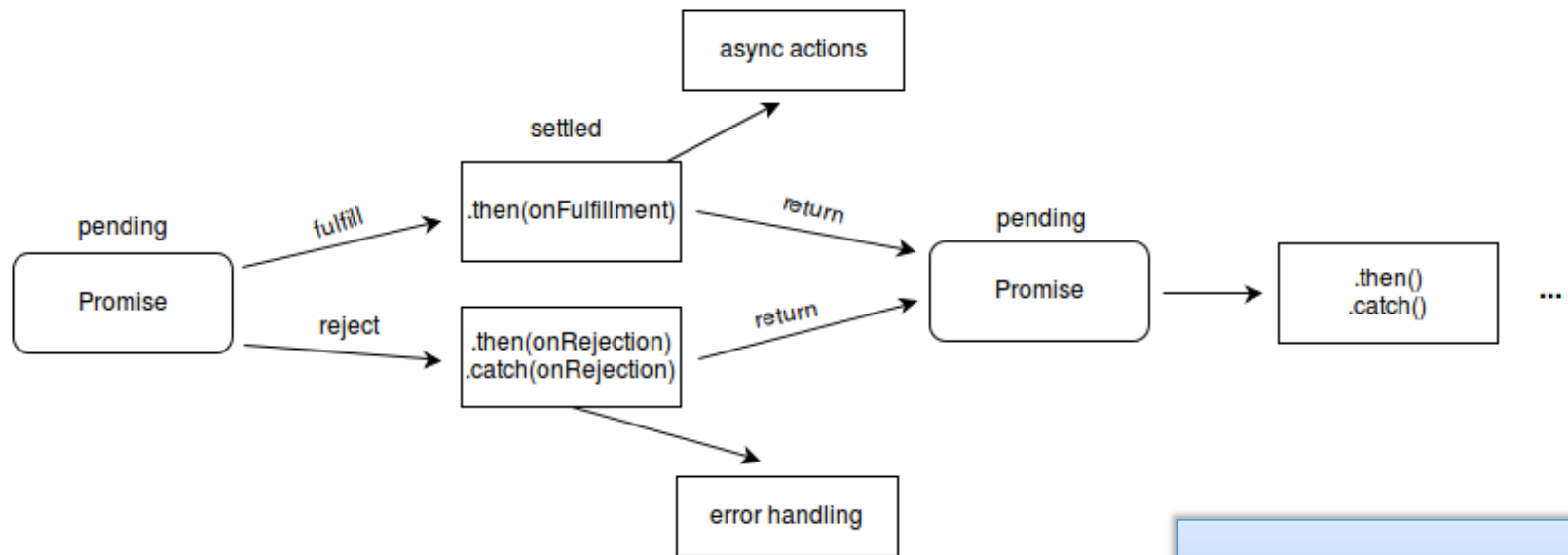


Recent Evolution of JavaScript

- Jun 1997 - ECMAScript as ECMA-262 specification
- Dec 1999 - ECMAScript 3 = *JavaScript*
- Dec 2009 - ECMAScript 5 = *strict mode*
- Jun 2015 - ECMAScript 6 (aka ES2015) = *Harmony*
 - classes, modules, arrow functions, block scope let & const, promises, generators, ...
- Jun 2017 - ES2017
 - concurrency, atomics, async/await
-



Promises are Better than Callbacks



https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

See also <http://www.html5rocks.com/en/tutorials/es6/promises/>

Promise States
Pending
Fulfilled / Rejected
Settled

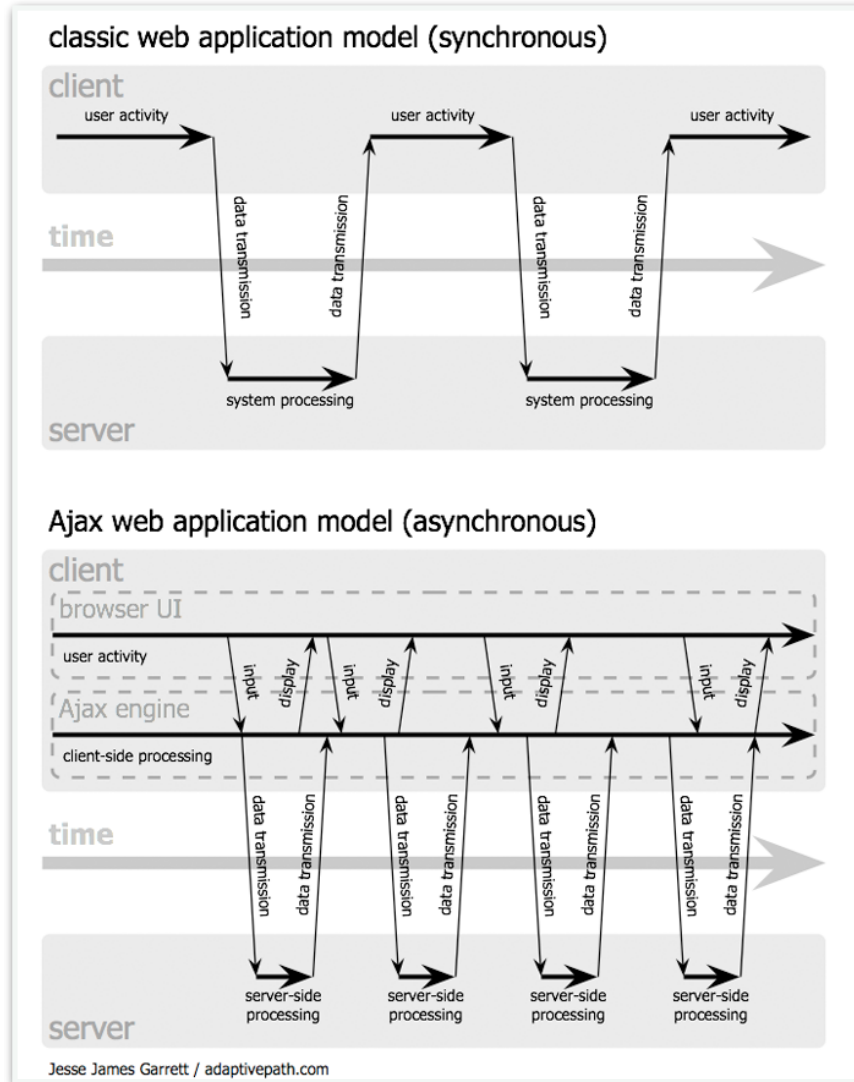


Communicating with the Server

- GET, POST, etc...
- Always instantiated by the browser (i.e. user)
 - using a link
 - clicking on a button (e.g. submitting a form)
- Convenient to use JavaScript to ask the server for data



The first “A” in AJAX



JSONPlaceholder (Server)

{JSON} Placeholder

Free fake and reliable API for testing and prototyping.

Powered by JSON Server + LowDB.

Serving ~3 billion requests each month.

<https://jsonplaceholder.typicode.com/>



Fetch API

```
> fetch
< f fetch() { [native code] }
> fetch('https://jsonplaceholder.typicode.com/posts').then(r => r.json())
```

Inline function callbacks
with arrow notation

```
< ▼ Promise {<pending>} 1
  ▶ __proto__: Promise
    [[PromiseStatus]]: "resolved"
    ▼ [[PromiseValue]]: Array(100)
      ▶ 0: {userId: 1, id: 1, title: "sunt aut facere repellat provident", body: "quia molestias reprehenderit id ut nihil quas temporibus voluptate"}
      ▶ 1: {userId: 1, id: 2, title: "qui est esse", body: "est rerum tempore vero, et ut in quo, excepteur dolor enim"}
      ▶ 2: {userId: 1, id: 3, title: "ea molestias quasi exercitationem repellat qui ipsa sit", body: "autem est delectatibus sunt, ad autem nostrum rationibus"}
      ▶ 3: {userId: 1, id: 4, title: "eum et est occaecati", body: "ullam et saepe reiciendis voluptatem adipisci"}
      ▶ 4: {userId: 1, id: 5, title: "nesciunt quas odio", body: "repudiandae veniam quaerat sunt sed"}
      ▶ 5: {userId: 1, id: 6, title: "dolorem eum magni eos aperiam quia", body: "ut enim aut sit nesciunt"}
      ▶ 6: {userId: 1, id: 7, title: "magnam facilis autem", body: "dolore placeat quibusdam doloribus voluptas"}
      ▶ 7: {userId: 1, id: 8, title: "dolorem dolore est ipsam", body: "dignissimos ad eum ipsa sapiente excepteur dolor"}
      ▶ 8: {userId: 1, id: 9, title: "nesciunt iure omnis dolorem tempora et accusantium", body: "consectetur ut nulla in"}
      ▶ 9: {userId: 1, id: 10, title: "optio molestias id quia eum", body: "quo et ex distinctio"}
      ▶ 10: {userId: 2, id: 11, title: "et ea vero quia laudantium autem", body: "consectetur ut nulla in"}
      ▶ 11: {userId: 2, id: 12, title: "in quibusdam tempore odit est dolorem", body: "voluptatem"}
      ▶ 12: {userId: 2, id: 13, title: "dolorum ut in voluptas mollitia et saepe quo", body: "voluptas"}
      ▶ 13: {userId: 2, id: 14, title: "voluptatem eligendi optio", body: "fuga et voluptatem"}
      ▶ 14: {userId: 2, id: 15, title: "eveniet quod temporibus", body: "reprehenderit et dolor eum"}
      ▶ 15: {userId: 2, id: 16, title: "sint suscipit perspiciatis velit dolorum rerum ipsa", body: "tempore"}
      ▶ 16: {userId: 2, id: 17, title: "fugit voluptas sed molestias voluptatem pro", body: "voluptate"}
      ▶ 17: {userId: 2, id: 18, title: "voluptate et itaque vero tempora molestiae", body: "tempore"}
      ▶ 18: {userId: 2, id: 19, title: "adipisci placeat illum aut reiciendis qui", body: "voluptas"}
      ▶ 19: {userId: 2, id: 20, title: "doloribus ad provident suscipit at", body: "voluptatem"}
      ▶ 20: {userId: 3, id: 21, title: "asperiores ea ipsam voluptatibus modi minima", body: "enim aut sit"}
      ▶ 21: {userId: 3, id: 22, title: "ut qui aut aut", body: "voluptatem"}
      ▶ 22: {userId: 3, id: 23, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 23: {userId: 3, id: 24, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 24: {userId: 3, id: 25, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 25: {userId: 3, id: 26, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 26: {userId: 3, id: 27, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 27: {userId: 3, id: 28, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 28: {userId: 3, id: 29, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 29: {userId: 3, id: 30, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 30: {userId: 3, id: 31, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 31: {userId: 3, id: 32, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 32: {userId: 3, id: 33, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 33: {userId: 3, id: 34, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 34: {userId: 3, id: 35, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 35: {userId: 3, id: 36, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 36: {userId: 3, id: 37, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 37: {userId: 3, id: 38, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 38: {userId: 3, id: 39, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 39: {userId: 3, id: 40, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 40: {userId: 3, id: 41, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 41: {userId: 3, id: 42, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 42: {userId: 3, id: 43, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 43: {userId: 3, id: 44, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 44: {userId: 3, id: 45, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 45: {userId: 3, id: 46, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 46: {userId: 3, id: 47, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 47: {userId: 3, id: 48, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 48: {userId: 3, id: 49, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 49: {userId: 3, id: 50, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 50: {userId: 3, id: 51, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 51: {userId: 3, id: 52, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 52: {userId: 3, id: 53, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 53: {userId: 3, id: 54, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 54: {userId: 3, id: 55, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 55: {userId: 3, id: 56, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 56: {userId: 3, id: 57, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 57: {userId: 3, id: 58, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 58: {userId: 3, id: 59, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 59: {userId: 3, id: 60, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 60: {userId: 3, id: 61, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 61: {userId: 3, id: 62, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 62: {userId: 3, id: 63, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 63: {userId: 3, id: 64, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 64: {userId: 3, id: 65, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 65: {userId: 3, id: 66, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 66: {userId: 3, id: 67, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 67: {userId: 3, id: 68, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 68: {userId: 3, id: 69, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 69: {userId: 3, id: 70, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 70: {userId: 3, id: 71, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 71: {userId: 3, id: 72, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 72: {userId: 3, id: 73, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 73: {userId: 3, id: 74, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 74: {userId: 3, id: 75, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 75: {userId: 3, id: 76, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 76: {userId: 3, id: 77, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 77: {userId: 3, id: 78, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 78: {userId: 3, id: 79, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 79: {userId: 3, id: 80, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 80: {userId: 3, id: 81, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 81: {userId: 3, id: 82, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 82: {userId: 3, id: 83, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 83: {userId: 3, id: 84, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 84: {userId: 3, id: 85, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 85: {userId: 3, id: 86, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 86: {userId: 3, id: 87, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 87: {userId: 3, id: 88, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 88: {userId: 3, id: 89, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 89: {userId: 3, id: 90, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 90: {userId: 3, id: 91, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 91: {userId: 3, id: 92, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 92: {userId: 3, id: 93, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 93: {userId: 3, id: 94, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 94: {userId: 3, id: 95, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 95: {userId: 3, id: 96, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 96: {userId: 3, id: 97, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 97: {userId: 3, id: 98, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 98: {userId: 3, id: 99, title: "autem aut aut sit", body: "voluptatem"}
      ▶ 99: {userId: 3, id: 100, title: "autem aut aut sit", body: "voluptatem"}
    
```

Same as:

```
.then(function(r) {
  return r.json()
})
```

`r.json()` returns a *Promise*, the next `then()` is called when `json()` resolves.



JavaScript Object Notation (JSON)

Be careful when building string to convert to JSON

Client and Server use JSON to communicate data to/from persistent storage (e.g. database)

```
> total = JSON.parse('{age:30}')
```

✖ ▶ Uncaught SyntaxError: Unexpected token a in JSON at position 1
at JSON.parse (<anonymous>)
at <anonymous>:1:14

```
> total = JSON.parse('{"age":30}')
```


```
< ▶ {age: 30}
```



JavaScript Object Notation (JSON)

Object.keys(), Object.values()
work for JSON

Error occurs when passing Object.keys()
a non-JSON argument



```
function parseJSON(url) {  
  
    //print out the JSON key, value pairs (assuming input is json here!)  
    return fetch(url)  
        .then(res => res.json())  
        .then(res => console('keys: ' + Object.keys(res) + ', values: ' + Object.values(res)));  
}
```



Jasmine

- Jasmine is a test framework for JavaScript
- SpecRunner is the main engine for running tests
- Browser displays test results

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="utf-8">
5    <title>Jasmine Spec Runner v5.10.0</title>
6
7    <link rel="shortcut icon" type="image/png" href="lib/jasmine-5.10.0/jasmine_favicon.png">
8    <link rel="stylesheet" href="lib/jasmine-5.10.0/jasmine.css">
9
10   <script src="lib/jasmine-5.10.0/jasmine.js"></script>
11   <script src="lib/jasmine-5.10.0/jasmine-html.js"></script>
12   <script src="lib/jasmine-5.10.0/boot0.js"></script>
13   <!-- optional: include a file here that configures the Jasmine env -->
14   <script src="lib/jasmine-5.10.0/boot1.js"></script>
15
16   <!-- include source files here... -->
17   <script src="src/Player.js"></script>
18   <script src="src/Song.js"></script>
19
20   <!-- include spec files here... -->
21   <script src="spec/SpecHelper.js"></script>
22   <script src="spec/PlayerSpec.js"></script>
```



Jasmine Test Structure (spec)

Jasmine test description

Jasmine test to run

Jasmine test assertion

```
6 describe("Jasmine Inclass Fetch Exercises", () => {
7     const baseUrl = 'https://jsonplaceholder.typicode.com';
8     const posts = `${baseUrl}/posts`;
9
10
11     it('countWords should count the number of words in each post', (done) => {
12         countWords(posts)
13         .then(res => {
14             let wordCounts = Object.values(res);
15             expect(wordCounts[0]).toEqual(20);
16             expect(wordCounts[1]).toEqual(28);
17             expect(wordCounts[2]).toEqual(23);
18             done();
19         });
20     });
```



Jasmine Source: countWords, getLastLargest

- Jasmine src directory contains JavaScript code
- Implement functionality to pass spec tests

```
1 // Inclass Fetch Exercise
2 // =====
3
4 function countWords(url) {
5   return fetch(url)
6     .then(res => res.json()).then(res => {
7     let postWordCounts = {};
8
9     // TODO: get the word count for each post
10    //postWordCounts[postId] = postSize;
11
12
13    // return an object { postId: wordCount }
14    return postWordCounts;
15  });
16 }
17
18 function getLastLargest(url) {
19   return countWords(url)
20     .then(postWordCounts => {
21     let largestCountSoFar = 0;
22     let largestCountPostId = 0;
23
24     // TODO: get all the post id keys
25
26     // TODO: now find the post id with longest post
27     return largestCountPostId;
28   });
29 }
```



Jasmine Test Failures

```
Jasmine 3.1.0
XX

2 specs, 2 failures, randomized with seed 50617
Spec List | Failures

Jasmine Inclass Fetch Exercises > countWords should count the number of words in each post

Expected 2 to equal 20.
Error: Expected 2 to equal 20.
    at <Jasmine>
    at countWords.then.res (file:///Users/mjoyner/comp431/webdev/inclass_exercises/ic8/jasmine/spec/jasmine-inclass-fetch.spec.js:15:36)

Expected 4 to equal 28.
Error: Expected 4 to equal 28.
    at <Jasmine>
    at countWords.then.res (file:///Users/mjoyner/comp431/webdev/inclass_exercises/ic8/jasmine/spec/jasmine-inclass-fetch.spec.js:16:36)

Expected 6 to equal 23.
Error: Expected 6 to equal 23.
    at <Jasmine>
    at countWords.then.res (file:///Users/mjoyner/comp431/webdev/inclass_exercises/ic8/jasmine/spec/jasmine-inclass-fetch.spec.js:17:36)

Jasmine Inclass Fetch Exercises > getLastLargest should return the id of the last post with the most words

Expected 5 to equal 97.
Error: Expected 5 to equal 97.
    at <Jasmine>
    at getLastLargest.then.res (file:///Users/mjoyner/comp431/webdev/inclass_exercises/ic8/jasmine/spec/jasmine-inclass-fetch.spec.js:25:36)
```

