IN4MATX 133: User Interface Software

Bootstrap and JavaScript

Goals for this Lecture

By the end of this lecture, you should be able to…

- Explain the basic functionality of CSS frameworks like Bootstrap
- Explain the different roles HTML, CSS, and JavaScript play
- Describe how JavaScript standards evolved
- Follow JavaScript syntax for traditional programming concepts like typing, variable assignment, loops, and conditionals

Opposition to Grid-based frameworks

Can stifle creativity

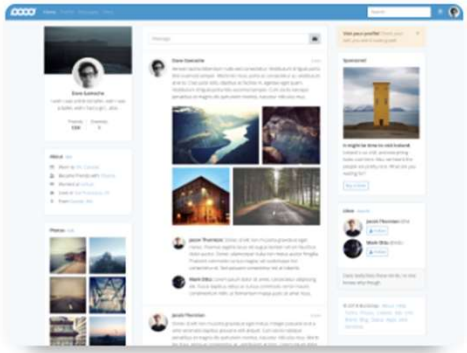
Themes built by or reviewed by Bootstrap's creators.

Why our themes?

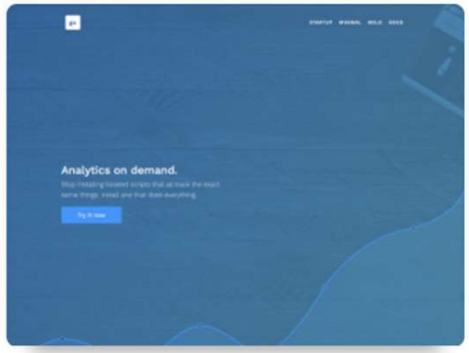
Built by Bootstrap Team

Component-based frameworks designed, built, and supported by the Bootstrap Team.

Dashboard $49.00
Admin & Dashboard ★★★★★



Application $49.00
Application ★★★★★

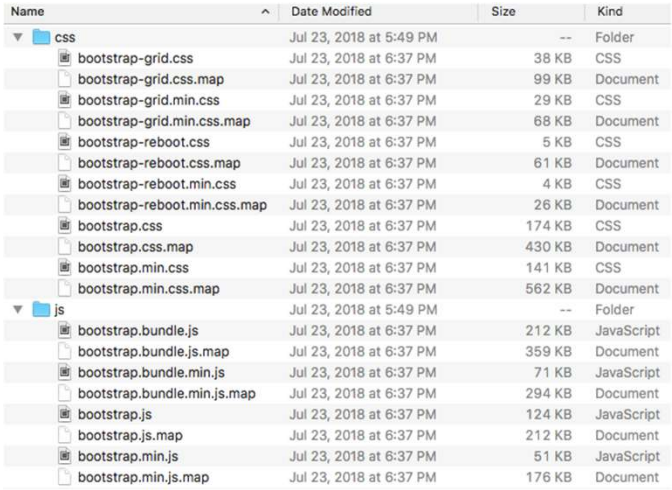


Marketing $49.00
Landing & Corporate ★★★★★

Bootstrap Grid Framework

Bootstrap

- Direct download
  - https://getbootstrap.com/docs/5.3/getting-started/introduction/
- CSS and JavaScript files
- Minified files are compressed, will load faster
- .map files support editing preprocessed files
  - We won't really touch on those in this class
  - For more: https://developer.chrome.com/blog/sourcemaps/
- We'll use bootstrap.min.css for now



Name	Date Modified	Size	Kind
css	Jul 23, 2018 at 5:49 PM	--	Folder
bootstrap-grid.css	Jul 23, 2018 at 6:37 PM	38 KB	CSS
bootstrap-grid.css.map	Jul 23, 2018 at 6:37 PM	99 KB	Document
bootstrap-grid.min.css	Jul 23, 2018 at 6:37 PM	29 KB	CSS
bootstrap-grid.min.css.map	Jul 23, 2018 at 6:37 PM	68 KB	Document
bootstrap-reboot.css	Jul 23, 2018 at 6:37 PM	5 KB	CSS
bootstrap-reboot.css.map	Jul 23, 2018 at 6:37 PM	61 KB	Document
bootstrap-reboot.min.css	Jul 23, 2018 at 6:37 PM	4 KB	CSS
bootstrap-reboot.min.css.map	Jul 23, 2018 at 6:37 PM	26 KB	Document
bootstrap.css	Jul 23, 2018 at 6:37 PM	174 KB	CSS
bootstrap.css.map	Jul 23, 2018 at 6:37 PM	430 KB	Document
bootstrap.min.css	Jul 23, 2018 at 6:37 PM	141 KB	CSS
bootstrap.min.css.map	Jul 23, 2018 at 6:37 PM	562 KB	Document
js	Jul 23, 2018 at 5:49 PM	--	Folder
bootstrap.bundle.js	Jul 23, 2018 at 6:37 PM	212 KB	JavaScript
bootstrap.bundle.js.map	Jul 23, 2018 at 6:37 PM	359 KB	Document
bootstrap.bundle.min.js	Jul 23, 2018 at 6:37 PM	71 KB	JavaScript
bootstrap.bundle.min.js.map	Jul 23, 2018 at 6:37 PM	294 KB	Document
bootstrap.js	Jul 23, 2018 at 6:37 PM	124 KB	JavaScript
bootstrap.js.map	Jul 23, 2018 at 6:37 PM	212 KB	Document
bootstrap.min.js	Jul 23, 2018 at 6:37 PM	51 KB	JavaScript
bootstrap.min.js.map	Jul 23, 2018 at 6:37 PM	176 KB	Document

Bootstrap

- Load bootstrap

```
<link rel="stylesheet" href="css/bootstrap.min.css">

<link rel="stylesheet" href="css/override.css">
```

Bootstrap

- Content Delivery Networks (CDN)
- Browser-side caching reduces burdens of loading files
- Integrity: hashes to ensure the downloaded file matches what's expected
    - Protects against server being compromised
- Crossorigin: some imports require credentials, anonymous requires none

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
```

Bootstrap

Specifying a viewport

- In page's `head`
- Sets device width and scale level (for zooming)

```
<head>
  <meta name="viewport" content="width=device-
width,initial-scale=1">
</head>
```

Bootstrap

Designating a container

- All bootstrap content lives in a container

```
<div class="container">
  <!--Bootstrap content-->
</div>
```

- Just a class; anything can be a container

```
<main class="container">
  <!--Bootstrap content-->
</main>
```

Bootstrap

Grid System

- Grid system has 12 columns
  - 12 has a lot of factors (1, 2, 3, 4, 6)
- Content over 12 columns will wrap
  - (3+6+4=13, the 4 will wrap)
- 15px gutter for each
- Classes for `row` and `col-[size]-[number]`

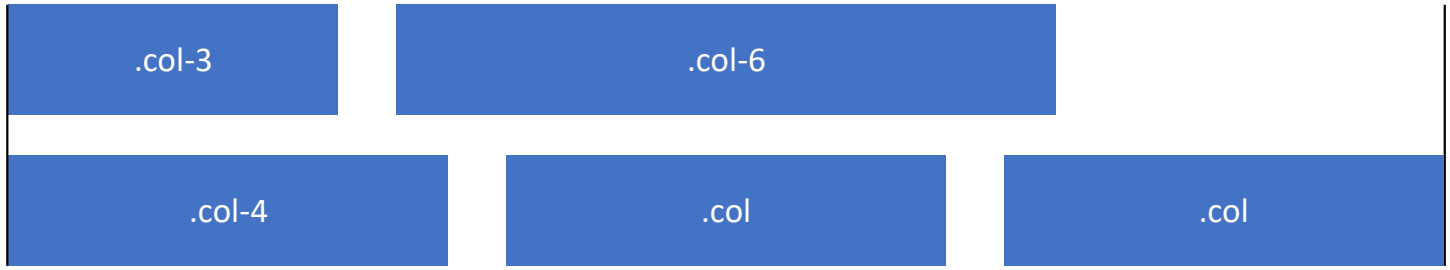
	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	`.col-xs-`	`.col-sm-`	`.col-md-`	`.col-lg-`
# of columns	12			
Column width	Auto	~62px	~81px	~97px
Gutter width	30px (15px on each side of a column)			
Nestable	Yes			
Offsets	Yes			
Column ordering	Yes			

Bootstrap

Grid System

- Within the same row, content will wrap once it goes over 12 columns
    - Size parameter is optional; will divide space proportionally

```
<main class="container">
  <div class="row">
    <div class="col-3">A</div>
    <div class="col-6">B</div>
    <div class="col-4">C</div>
    <div class="col">D</div>
    <div class="col">E</div>
  </div>
</main>
```

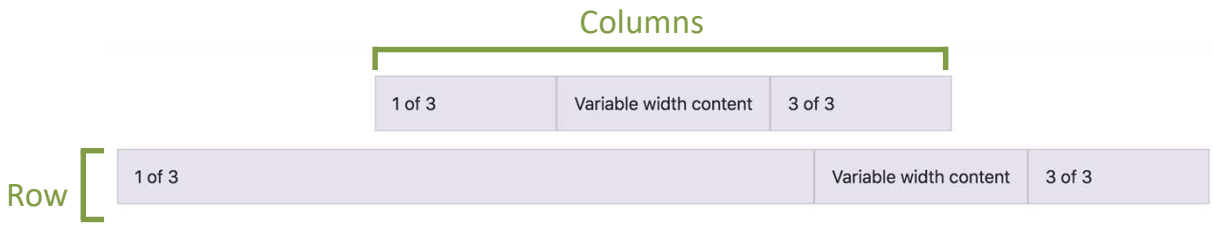


Bootstrap

Grid System

- Rows are block elements, while columns are inline

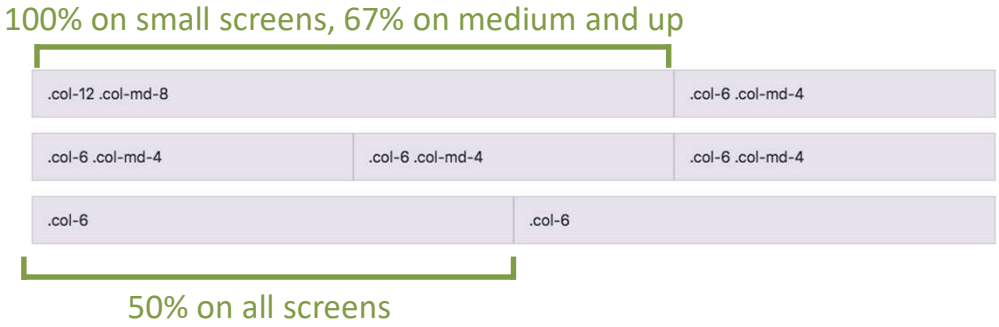
https://getbootstrap.com/docs/4.1/layout/grid/



Bootstrap

Grid System

- `.col` with no size defaults to the smallest (`xs`)
- The largest size listed will cover any larger sizes which are not-listed
- Will default to width 12 when no size is specified



Breakpoints

```
@media screen and (max-width: 640px) {
  /* small screens */
}

@media screen and (min-width: 640px and max-width:
1024px) {
  /* medium screens */
}

@media screen and (min-width: 1024px) {
  /* large screens */
}
```

Bootstrap

Hiding and showing

- There are some helpful classes for showing and hiding content across breakpoints

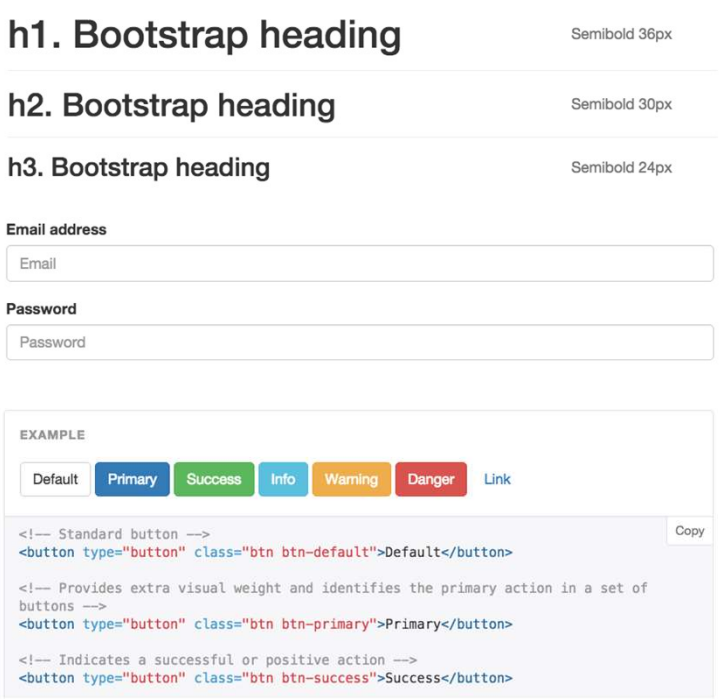
Screen size	Class
Hidden on all	`.d-none`
Hidden only on xs	`.d-none .d-sm-block`
Hidden only on sm	`.d-sm-none .d-md-block`
Hidden only on md	`.d-md-none .d-lg-block`
Hidden only on lg	`.d-lg-none .d-xl-block`
Hidden only on xl	`.d-xl-none .d-xxl-block`
Hidden only on xxl	`.d-xxl-none`
Visible on all	`.d-block`
Visible only on xs	`.d-block .d-sm-none`
Visible only on sm	`.d-none .d-sm-block .d-md-none`
Visible only on md	`.d-none .d-md-block .d-lg-none`
Visible only on lg	`.d-none .d-lg-block .d-xl-none`
Visible only on xl	`.d-none .d-xl-block .d-xxl-none`
Visible only on xxl	`.d-none .d-xxl-block`

http://getbootstrap.com/css

Bootstrap

Default styling

- Bootstrap will change a lot of styles for you
- There are other custom styles involving various suffixes

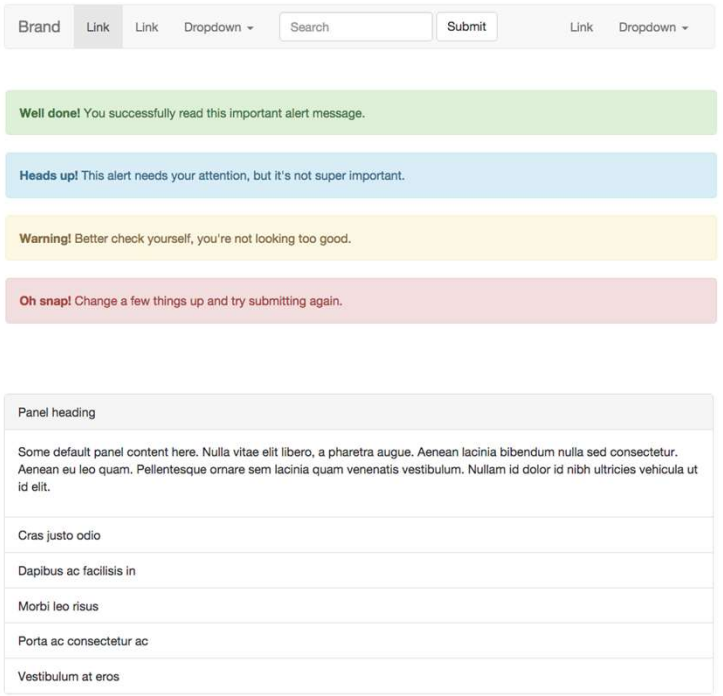


http://getbootstrap.com/css

Bootstrap

Components

- Components are elements pre-arranged into common patterns
- Makes making navigation bars, dropdowns, alerts, etc. simpler
- Some require JavaScript



http://getbootstrap.com/css

Introducing Interactivity to the Web

Language Roles

Markup
language

Styling
language

Programming
language

Language Roles

Specify how content is rendered

Visually style content

Dynamically manipulate content

Why JavaScript?

- Make pages dynamic
- Make pages personalized
- Make pages interact with other sources, like databases and APIs

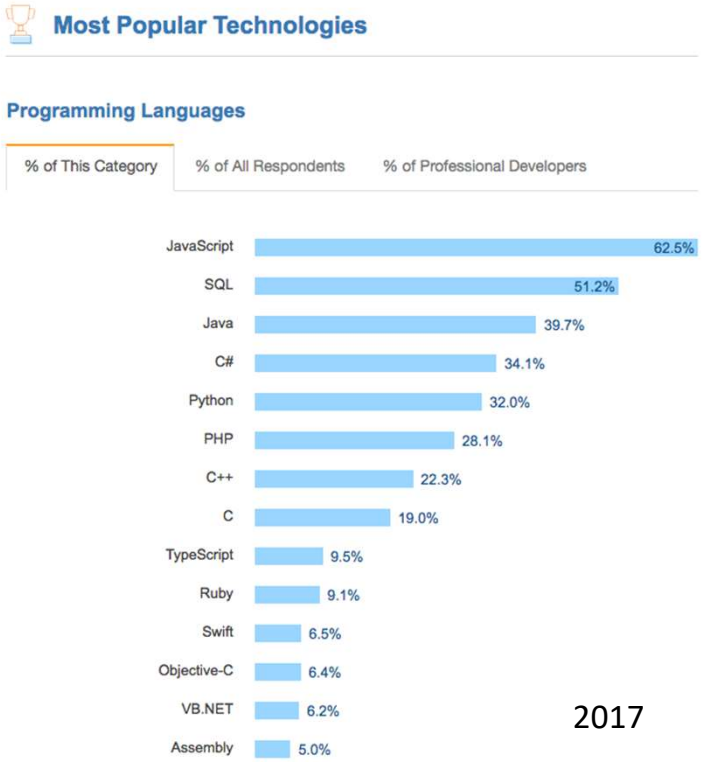
Other web programming languages

- Ruby, via Ruby on Rails
- Python, via Django or web2py
- These days, you can create a dynamic website in almost any language

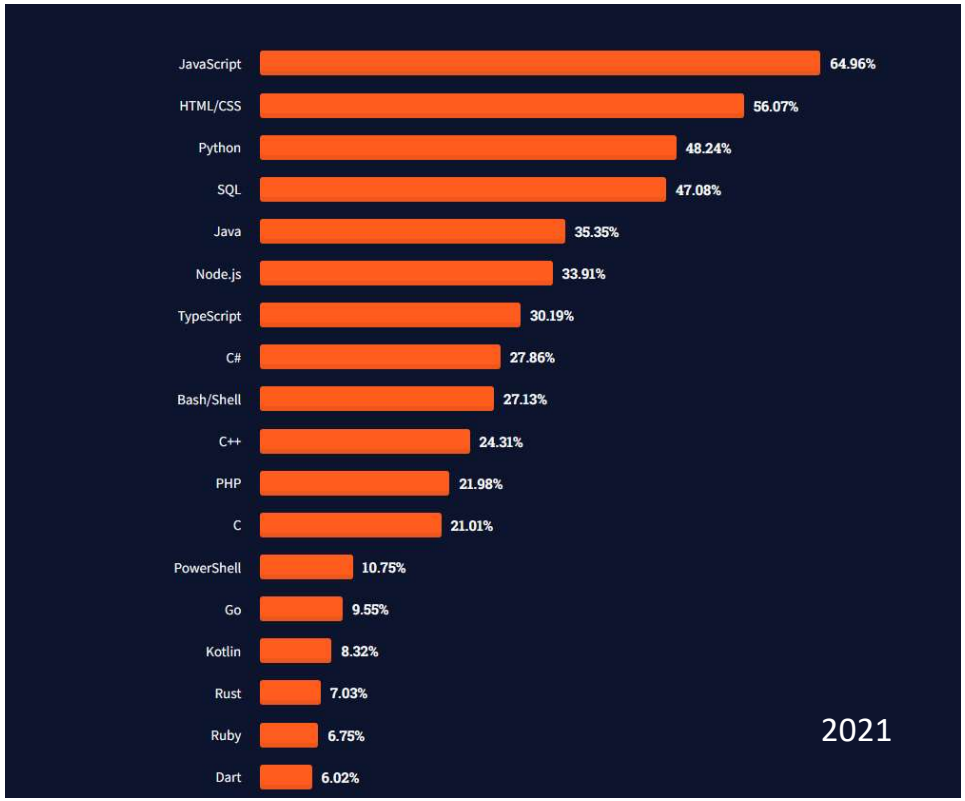
Other web programming languages

- Some languages transpile to JavaScript
- TypeScript, by Microsoft, introduces types
  - More on TypeScript later
- Kotlin, by Google, runs on the Java virtual machine and compiles to JavaScript
  - Links all of Google's platforms

JavaScript's popularity



2017



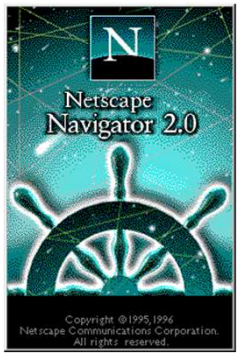
2021

https://insights.stackoverflow.com/survey/2021#technology-programming-languages

How did JavaScript become the most popular language for web development?

History of JavaScript

- *"Developed under the name Mocha, the language was officially called **LiveScript** when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it later was renamed JavaScript"*



- Java's popularity was on the rise
  - Marketing ploy
  - Intended to be the "web" language to Java's "desktop"

https://medium.com/@benastontweet/lesson-1a-the-history-of-javascript-8c1ce3bffb17

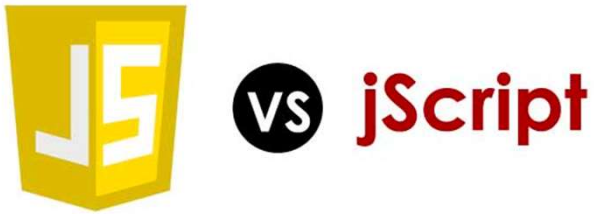
History of JavaScript

- Netscape submitted JavaScript to ECMA International for consideration as an industry standard
- Subsequent versions were standardized as “ECMAScript”
- Today, ECMAScript and JavaScript are more or less two different names for the same language

European Computer Manufacturers Association

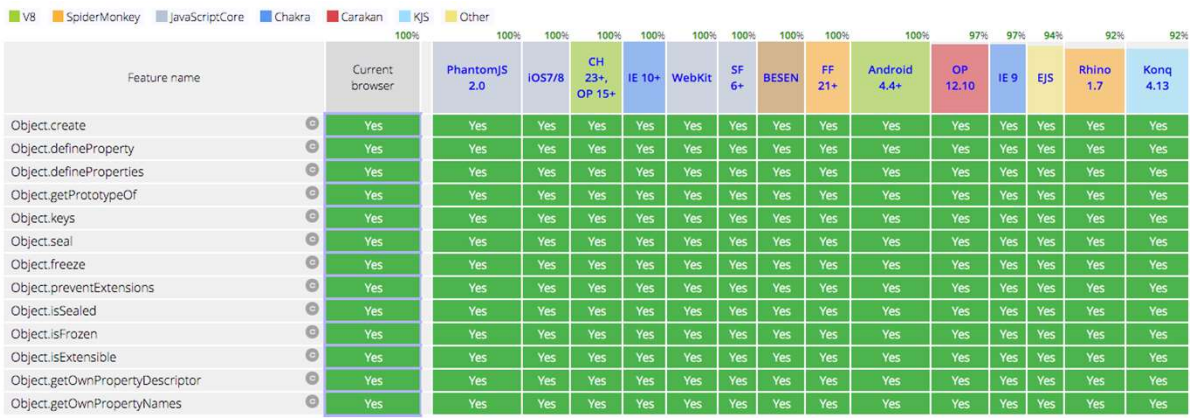
History of JavaScript

- Alternatives started springing up in the late 1990s and early 2000's
  - Microsoft introduced JScript engine
  - Macromedia Flash's ActionScript
- Both were vaguely JavaScript-like, but standards differed



History of JavaScript

- Standards later converged
  - Firefox came out in 2005
  - Adobe bought Flash
  - JScript followed the standards
- But browser's implementations of the language still vary

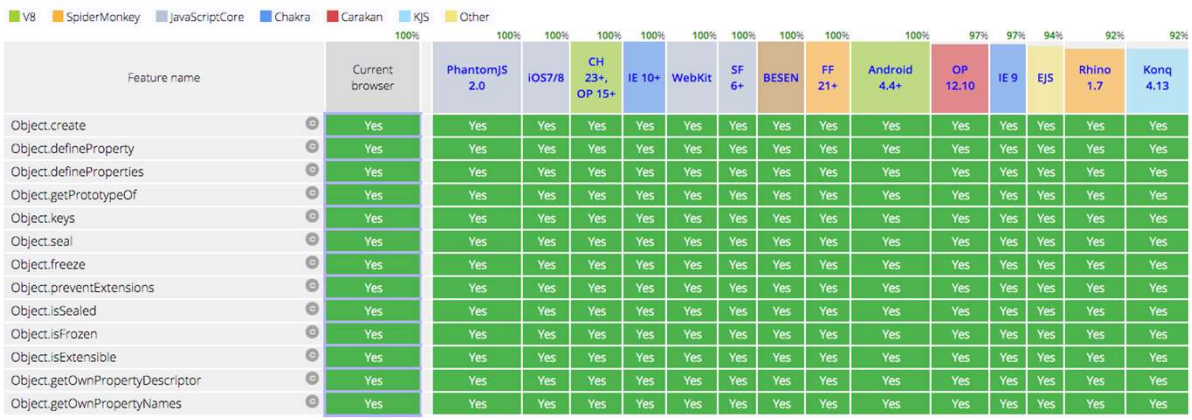


V8 SpiderMonkey JavaScriptCore Chakra Carakan KJS Other

Feature name	Current browser	PhantomJS 2.0	iOS7/8	CH 23+, OP 15+	IE 10+	WebKit	SF 6+	BESEN	FF 21+	Android 4.4+	OP 12.10	IE 9	EJS	Rhino 1.7	Konq 4.13
	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	97%	97%	94%	92%	92%
Object.create	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperty	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperties	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getPrototypeOf	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.keys	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.seal	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.freeze	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.preventExtensions	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isSealed	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isFrozen	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isExtensible	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyDescriptor	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyNames	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

History of JavaScript

- JavaScript Engines
  - SpiderMonkey (Firefox)
  - V8 (Chrome)
  - JavaScriptCore (Safari)
  - Carakan (Opera)
  - Chakra (IE & Edge)



V8 SpiderMonkey JavaScriptCore Chakra Carakan KJS Other

Feature name	Current browser	PhantomJS 2.0	iOS7/8	CH 23+, OP 15+	IE 10+	WebKit	SF 6+	BESEN	FF 21+	Android 4.4+	OP 12.10	IE 9	EJS	Rhino 1.7	Konq 4.13
	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	97%	97%	94%	92%	92%
Object.create	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperty	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.defineProperties	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getPrototypeOf	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.keys	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.seal	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.freeze	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.preventExtensions	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isSealed	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isFrozen	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.isExtensible	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyDescriptor	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Object.getOwnPropertyNames	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Versions of JavaScript

- You may see references to ECMAScript
- ECMAScript is just the standard for JavaScript
  - The last “major” release was ECMAScript 6, or ES6, or ECMAScript 2015, or ES2015
  - The latest is ECMAScript 2020, or ES11, or ES2020 (released in June 2020)

Versions of JavaScript

- Engines/Browsers continually play catch-up,
  so many tools support slightly older versions of the standard

		Compilers/polyfills					Desktop browsers												
	8%	2%	28%	39%	0%	28%	0%	1%	1%	1%	2%	2%	5%	8%	8%	5%	8%	9%	9%
Feature name ▸	Current browser	Traceur	Babel 6 + core-js	Babel 7 + core-js	Closure 2018.09	TypeScript + core-js	IE 11	Edge 16	Edge 17	Edge 18 Preview	FF 60 ESR	FF 61	FF 62	FF 63 Beta	FF 64 Nightly	CH 68, OP 55	CH 69, OP 56	CH 70, OP 57	CH 71, OP 58
Candidate (stage 3)																			
string.trimming ▸	4/4	0/4	4/4	4/4	0/4	4/4	0/4	2/4	2/4	2/4	2/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4
global ▸	0/2	0/2	2/2	2/2	0/2	2/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2
String.prototype.matchAll	No	No	Yes[4]	Yes[4]	No	Yes[5]	No	No	No	No	No	No	No	No	No	Flag[9]	Flag[9]	Flag[9]	Flag[9]
instance class fields ▸	0/3	1/3	1/3	1/3	0/3	1/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3	0/3
static class fields ▸	0/2	1/2	1/2	1/2	0/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2
Function.prototype.toString revision ▸	7/7	0/7	0/7	0/7	0/7	0/7	1/7	4/7	4/7	4/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7
Array.prototype.{flat, flatMap}[10] ▸	2/2	0/2	1/2	1/2	0/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	2/2	2/2	2/2	0/2	2/2	2/2	2/2
Symbol.prototype.description	No	No	No	No	No	No	No	No	No	No	No	No	No	Yes	Yes	No	Flag[9]	Yes	Yes
BigInt ▸	8/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	8/8	8/8	8/8	8/8
Object.fromEntries	No	No	No	No	No	No	No	No	No	No	No	No	No	Yes	Yes	No	No	No	No

Versions of JavaScript

- Polyfills ensure a user's browser has the latest libraries
  - Downloads "fill" versions of added functions, re-written using existing functions
  - Recreates missing features for older browsers
- Sometimes called a "shim" or a "fallback"

JavaScript

- Interpreted language
- Executed by a JavaScript engine
- Engine runs the same code that a programmer writes

Java

- Compiled language (into bytecode)
- Run in a Java Virtual Machine (JVM)
- Bytecode is unreadable by people

JavaScript

- Standardized through ECMAScript, but discrepancies exist
- Debugging dependent on execution environment
- Prototype-based?
- Used in every browser without a plugin

Java

- “Write once, deploy anywhere”
- Bugs found at compile time
- Class-based
- Requires a plugin to be run in most browsers

JavaScript is just a programming language

Printing in JavaScript

```
console.log("Hello, world!");
```

- Won't be visible in the browser
- Shows in the JavaScript Console

https://repl.it/@m5b/inf133-javascript-demo#index.html

JavaScript Syntax

- Has functions and objects
  - foo() bar.baz
  - They look like Java, but act differently

JavaScript Variables

- Variables are dynamically typed

```
var x = 'hello'; //value is a string
console.log(typeof x); //string

x = 42; //value is now a Number
console.log(typeof x); //number
```

- Unassigned variables have a value of `undefined`

```
var hoursSlept;
console.log(hoursSlept);
```

JavaScript types

```
console.log('40' + 2); //'402'
console.log('40' - 4); //36

var num = 10;
var str = '10';

//comparisons: these will all be booleans (true/false)
console.log(num == str); //true
console.log(num === str); //false
console.log('' == 0); //true
```

Minus isn't defined for strings, so JavaScript knows to convert this

=== "strict equality" same as "==" but without type conversion

JavaScript loops and conditionals

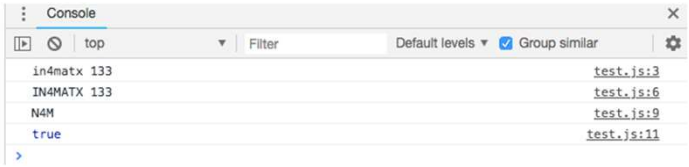
```
var i = 4.4;

if(i > 5) {
  console.log('i is bigger than 5');
} else if(i >= 3) {
  console.log('i is between 3 and 5');
} else {
  console.log('i is less than 3');
}

for(var x = 0; x < 5; x++) {
  console.log(x);
}
```

JavaScript methods

- Called with dot notation




```
var className = 'in4matx 133';
console.log(className);

className = className.toUpperCase();
console.log(className);

var part = className.substring(1, 4);
console.log(part);

console.log(className.indexOf('MATX') >= 0); //whether
the substring appears
```

JavaScript arrays

- Similar to Java, but can be a mix of different types

```
var letters = ['a', 'b', 'c'];
var numbers = [1, 2, 3];
var things = ['raindrops', 2.5, true, [5, 9, 8]]; //arrays can be nested
var empty = [];
var blank5 = new Array(5); //empty array with 5 items

//access using [] notation like Java
console.log( letters[1] ); //=> "b"
console.log( things[3][2] ); //=> 8

//assign using [] notation like Java
letters[0] = 'z';
console.log( letters ); //=> ['z', 'b', 'c']

//assigning out of bounds automatically grows the array
letters[10] = 'g';
console.log( letters);
    //=> [ 'z', 'b', 'c', , , , , , , , 'g' ]
console.log( letters.length ); //=> 11
```

JavaScript arrays

- Arrays have their own methods

```
//Make a new array
var array = ['i','n','f','x'];

//add item to end of the array
array.push('133');
console.log(array); //=> ['i','n','f','x','133']

//combine elements into a string
var str = array.join('-');
console.log(str); //=> "i-n-f-x-133"

//get index of an element (first occurrence)
var oIndex = array.indexOf('x'); //=> 3

//remove 1 element starting at oIndex
array.splice(oIndex, 1);
console.log(array); //=> ['i','n','f','133']
```

JavaScript objects

- An unordered set of key and value pairs
  - Like a HashMap in Java or a dictionary in Python
  - Sometimes called *associative arrays*

Quotes around keys are optional

```
ages = {alice:40, bob:35, charles:13}
extensions = {'mark':1622, 'in4matx':9937}
num_words = {1:'one', 2:'two', 3:'three'}
things = {num:12, dog:'woof', list:[1,2,3]}
empty = {}
empty = new Object(); //empty object
```

Goals for this Lecture

By the end of this lecture, you should be able to…

- Explain the basic functionality of CSS frameworks like Bootstrap
- Explain the different roles HTML, CSS, and JavaScript play
- Describe how JavaScript standards evolved
- Follow JavaScript syntax for traditional programming concepts like typing, variable assignment, loops, and conditionals