Week 11

Conditionals, Loops, and Arrays, and More JS Libraries

Conditional Statements

A conditional statement allows you to do different things based on the truthiness of a condition.

"If x is the case, then do y!"

Quick review of booleans (`true` and `false`)

- Everything in JavaScript can be considered ("coerced") to be **`true`** or **`false`**.
- Things that are considered "true-like" are called **truthy**, and the others are called **falsy**.
  - Falsy: **`false, 0, '', null`**
  - Truthy: most other things that aren't empty
- Comparisons or checks also evaluate to **`true`** or **`false`.**

The `if` statement

```
let number = 5;
```

???

```
// print whether number is
equal to 5
```

The `if` statement

```
let number = 5;

if (number == 5) {
    console.log("Number is equal to 5!");
}
```

Number is equal to 5!

The `if` statement

```
let number = 5;

if (number == 5) {
    console.log("Number is equal to 5!");
}
if (number > 4) {
    console.log("Number is greater than 4!");
}
```

Number is equal to 5!

Number is greater than 4!

The `if … else` statement

```
let number = 5;
```

More than 3

```
if (number > 3) {
    console.log("More than 3");
} else {
    console.log("Less than or equal to 3");
}
```

The `if … else` statement

```
let number = 3;

if (number > 3) {
    console.log("More than 3");
} else {
    console.log("Less than or equal to 3");
}
```

```
Less than or equal to 3
```

The `if … else if … else` statement

```
let number = 3;


if (number == 3) {
    console.log("Number is 3");
} else if (number > 3) {
    console.log("Greater than 3");
} else {
    console.log("Less than 3");
}
```

`if` statements inside other `if` statements!

```
let a = 5, b = 3;


if (a < 10) {
    if (b < 10) {
        console.log(“Both a and b are less than 10”);
    }
}
```

Questions?

Demo

https://tinyurl.com/wdd-if-loop

Loopy Loops!

A for loop allows you to repeat some code until a certain condition is met, or for a certain number of times.

What is a for loop?

- Does a thing for you a certain number times without you having to write out every single iteration
- Cleaner, more maintainable code

```
for (let i = 0; i < 10; i++) {
    console.log("counting to 9: ", + i);
}
```

Why use a for loop?

```
console.log("counting to 9: ", + 1);
console.log("counting to 9: ", + 2);
console.log("counting to 9: ", + 3);
console.log("counting to 9: ", + 4);
console.log("counting to 9: ", + 5);
console.log("counting to 9: ", + 6);
console.log("counting to 9: ", + 7);
console.log("counting to 9: ", + 8);
console.log("counting to 9: ", + 9);
```

```
for (let i = 1; i < 10; i++) {
    console.log("counting to 9: ", + i);
}
```

for loop breakdown: start

- Create a variable named `i`
  - Stands for “index”
- Set the variable value to `0`

```
for (let i = 0; i < 10; i++) {
    console.log(“counting to 9: ”, + i);
}
```

for loop breakdown: end

- Keep doing the action(s) inside the for loop while `i` is less than `10`

```
for (let i = 0; i < 10; i++) {
    console.log("counting to 9: ", + i);
}
```

for loop breakdown: repeat

- Increment `i` by 1 every time the loop repeats execution
- Also written as `i += 1`

```
for (let i = 0; i < 10; i++) {
    console.log("counting to 9: ", + i);
}
```

What will this print?

```
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

What will this print?

```
for (let i = 1; i < 12; i++) {
    console.log(i);
}
```

What will this print?

```
for (let i = 0; i <= 10; i++) {
    console.log(i);
}
```

What will this print?

```
for (let i = 1; i < 4; i++) {
    for (let j = 1; j < 4; j++) {
        console.log(i, j);
    }
}
```

What will this print?

```
for (let i = 1; i < 4; i++) {
    for (let j = 1; j < 4; j++) {
        console.log(i, j);
    }
    console.log("potate");
}
```

Other Kinds of Loops

- While loops
- Do-while loops
- For-each loops
- Froot Loops!

Questions?

Demo

https://tinyurl.com/wdd-for-loop

Arrays

In JavaScript, an array is a list of values (perhaps of different types)

Array examples

```
[102, 82, 34]

["WDD", "is", "awesome!"]

["It is", 2018]

["Is it", true, 432]
```

Array indexing

A lot of programming languages have zero-indexed arrays, so to get the first item from an array, we use `[0]`

```
const numbers = [20, "potato", 30];

console.log(numbers[0]); // Output: 20
console.log(numbers[1]); // Output: potato
console.log(numbers[2]); // Output: 30

// Similar to how strings output characters

const text = "WDD";

console.log(text[0]); // Output: W
console.log(text[1]); // Output: D
```

Array properties & methods

`array.length`

Returns the length of the array

`array.push(item1, item2, item3, …)`

Adds a new item to the end of the array

`array.pop()`

Removes the last item in the list and returns it

Notice that the array mutates "internally"

```
const numbers = [];
console.log(numbers.length); // Output: 0

numbers.push(1, 2, 3);
console.log(numbers); // Output: [1, 2, 3]
console.log(numbers.length); // Output: 3

numbers.pop();
console.log(numbers); // Output: [1, 2]
console.log(numbers.length); // Output: 2

numbers.push(4, 5, 6);
console.log(numbers); // Output: [1, 2, 4, 5, 6]
console.log(numbers.length); // Output: 5
```

Iterating over an array

Approaching from a traditional for-loop

Since there are as many as `farms.length` strings in the array, we iterate `i` over the range of indices and use `farms[i]` to get the name of each farm in the array.

```
const farms = [
   "Homegrown Organic Farms",
   "Tomatero Organic Farm",
   "Tutti Frutti Farms"
];

for (let i = 0; i < farms.length; i++) {
   console.log("Does", farms[i], "have
potatoes?");
}
```

Questions?

Bring On the Pasta

Languages and Frameworks and Libraries, Oh My!

- Where JavaScript is a **language**, jQuery is a **library**:
  - JavaScript, like HTML or CSS, has a set of syntax rules that define a logical, executable environment.
  - Frameworks, libraries, and plugins are built on top of a language, and act as tools that we can use right out of the box.

Handy article: is jQuery a framework or a library

Languages and Frameworks and Libraries, Oh My!

- What about **frameworks**, **libraries** and **plugins**?
  - Frameworks tell us how to **structure** our code to make things easier for ourselves in the long run.
  - Libraries are **building blocks** that offer us generally helpful functionality for broad applications.
  - Plugins are **extensions** of an application, framework, or library.

Handy articles: frameworks vs. libraries (in JS), libraries vs. plugins (in Java)

Questions?

Where do I find plugins?

tldr: Online!

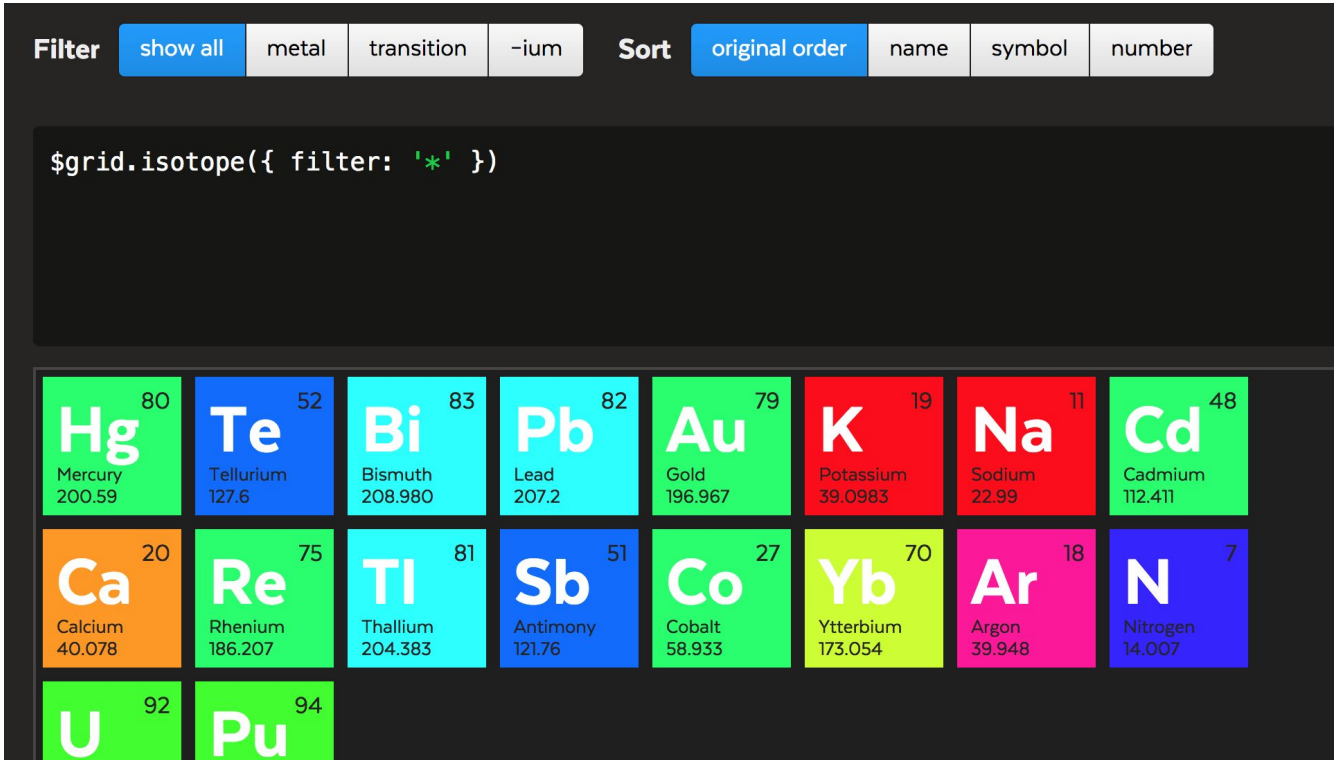
Some nice starting points:

- https://www.javascripting.com/
- https://plugins.jquery.com/
- https://www.creativebloq.com/jquery/top-jquery-plugins-6133175
- https://tutorialzine.com/2013/04/50-amazing-jquery-plugins
- http://www.unheap.com/***
- https://getflywheel.com/layout/best-javascript-libraries-frameworks-2019/

particles.js

Drag me around

jQuery UI

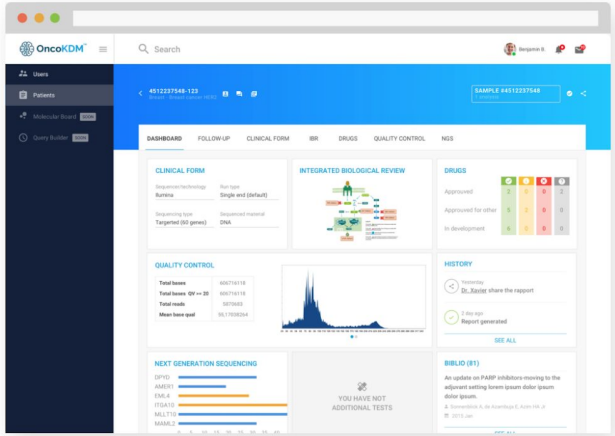

Filter
show all
metal
transition
-ium
Sort
original order
name
symbol
number
$grid.isotope({ filter: '*' })
Hg 80 Mercury 200.59
Te 52 Tellurium 127.6
Bi 83 Bismuth 208.980
Pb 82 Lead 207.2
Au 79 Gold 196.967
K 19 Potassium 39.0983
Na 11 Sodium 22.99
Cd 48 Cadmium 112.411
Ca 20 Calcium 40.078
Re 75 Rhenium 186.207
Tl 81 Thallium 204.383
Sb 51 Antimony 121.76
Co 27 Cobalt 58.933
Yb 70 Ytterbium 173.054
Ar 18 Argon 39.948
N 7 Nitrogen 14.007
U 92
Pu 94

Isotope

OncoKDM™

REQUEST A DEMO

Login

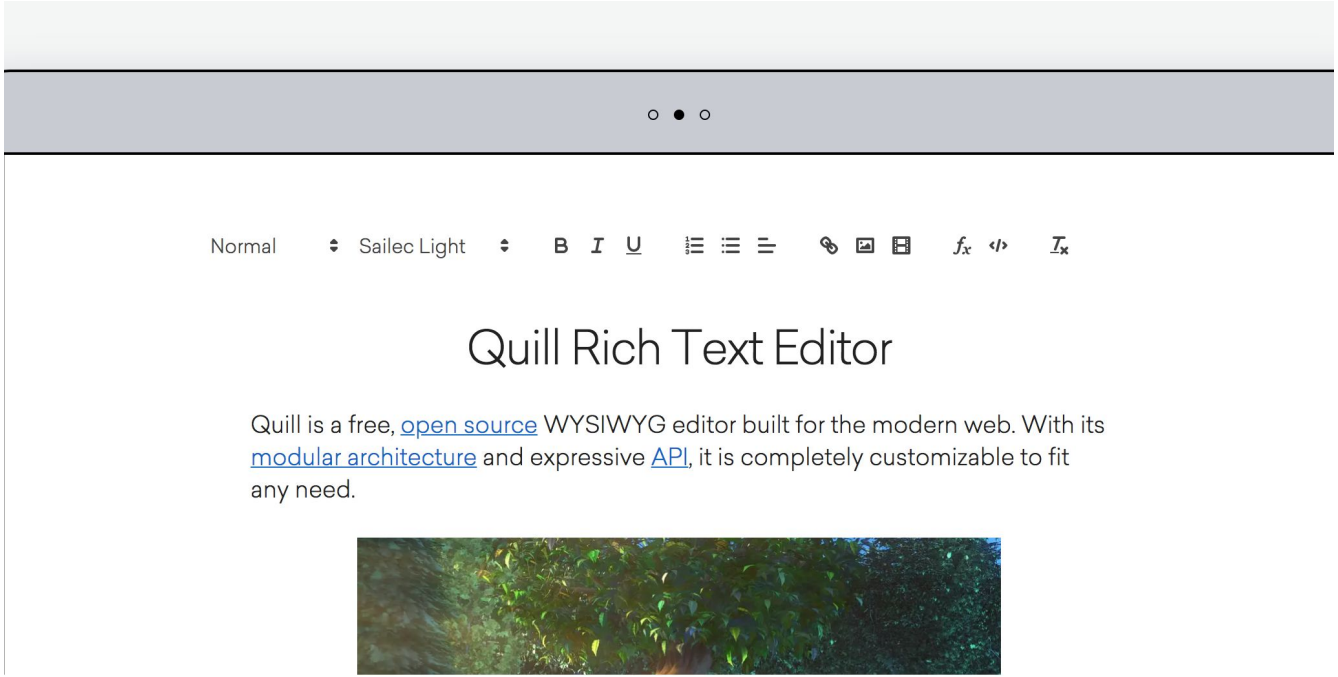


Comprehensive report

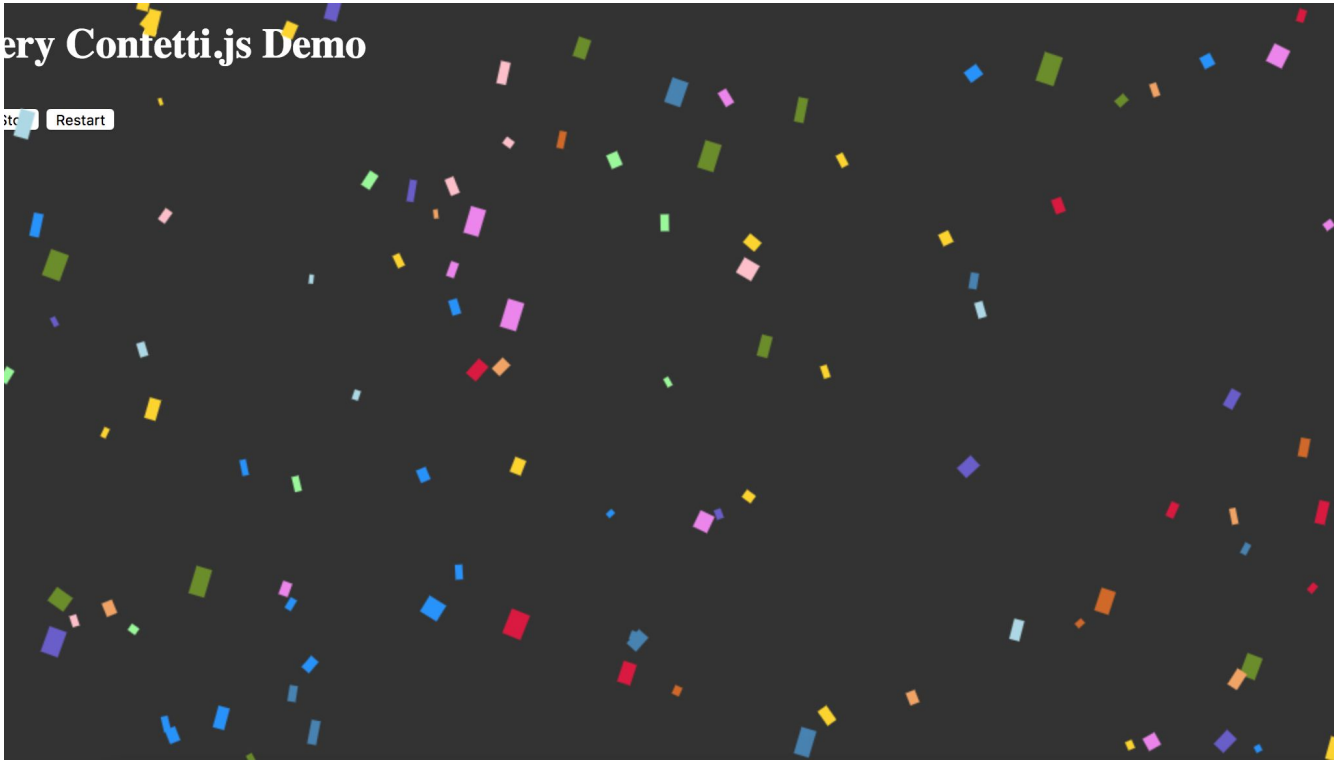
Import your raw data from the lab directly into our platform. In a very short time, OncoKDM will get you back a useful and comprehensive report that you can review, validate and then transfer to the oncologist.

Waypoints

example

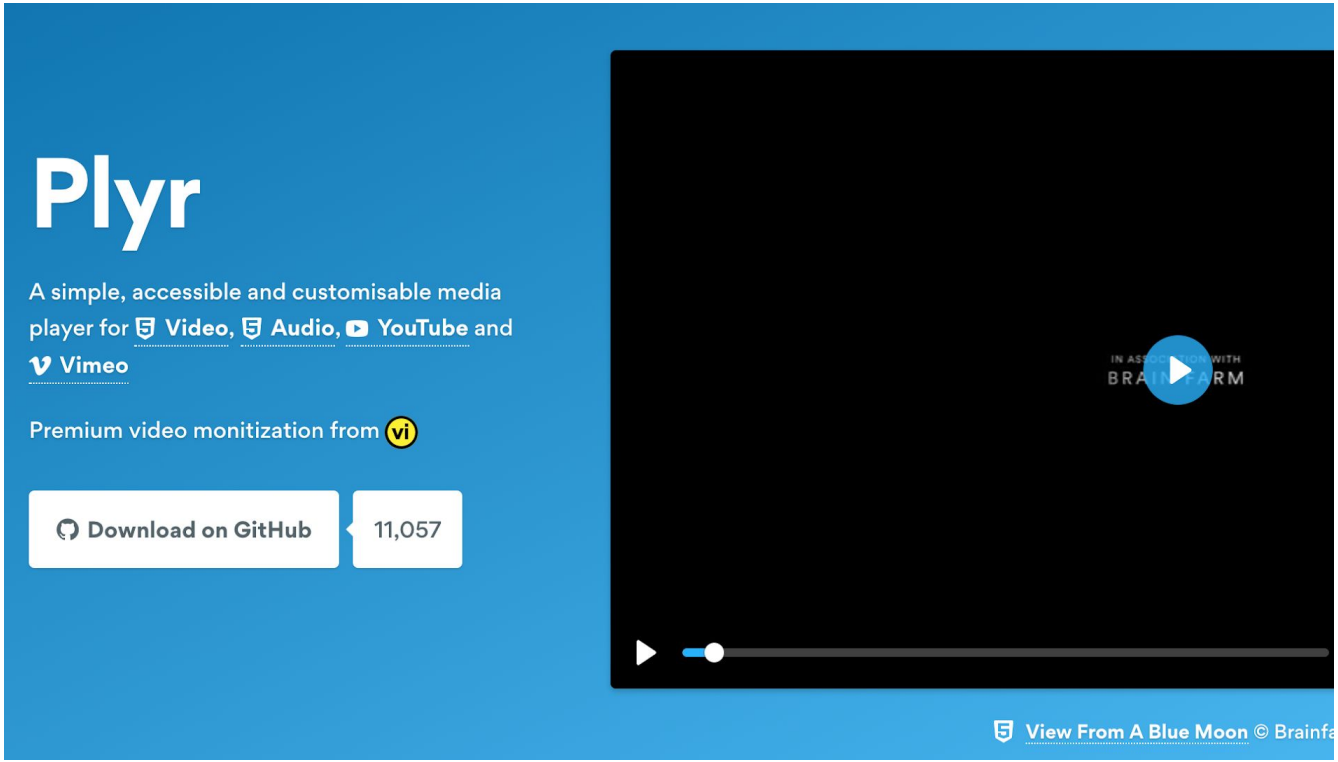

Normal
Sailec Light
Quill Rich Text Editor
Quill is a free, open source WYSIWYG editor built for the modern web. With its modular architecture and expressive API, it is completely customizable to fit any need.

Quill

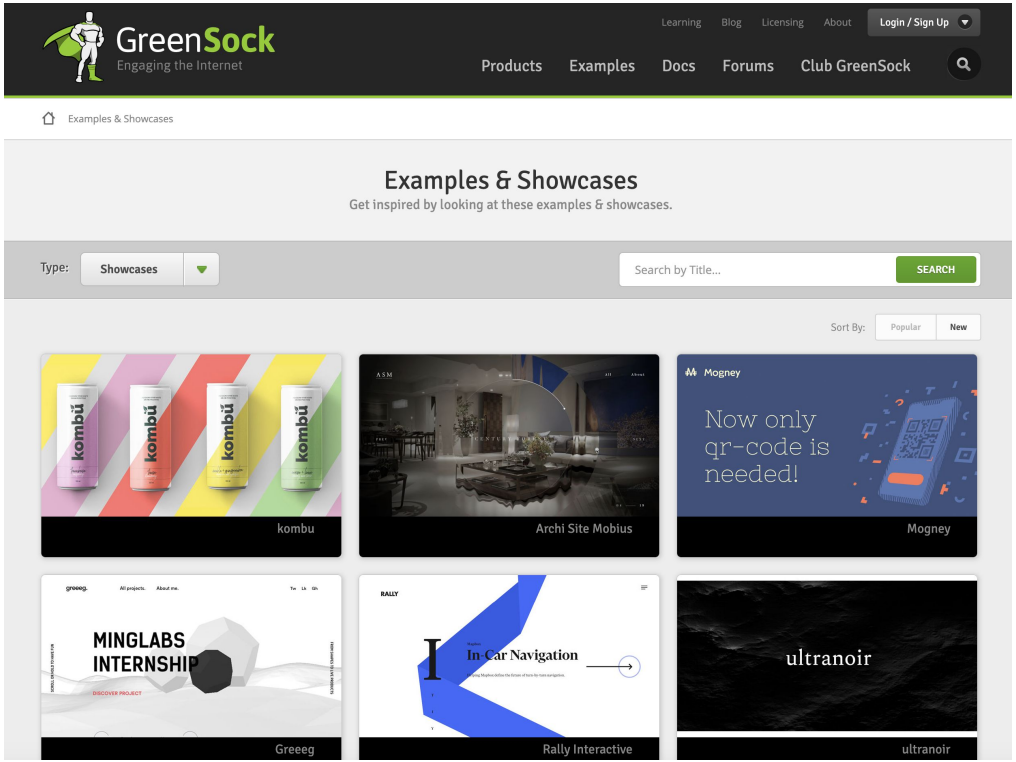


Confetti

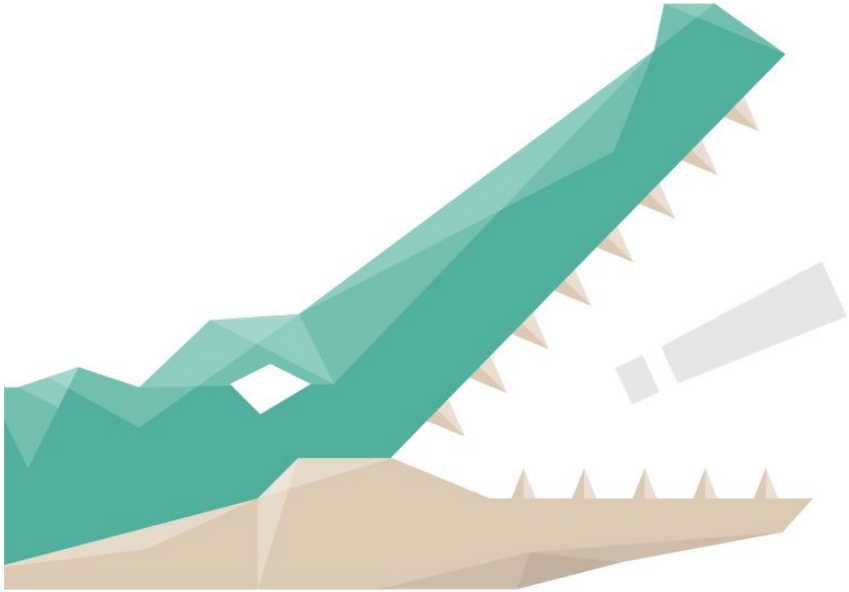
for when you need to recreate your berkeley letter of acceptance to feel a sense of gratification again


Plyr
A simple, accessible and customisable media player for Video, Audio, YouTube and Vimeo
Premium video monitization from
Download on GitHub
11,057
View From A Blue Moon © Brainfa

Plyr


GreenSock
Engaging the Internet
Learning
Blog
Licensing
About
Login / Sign Up
Products
Examples
Docs
Forums
Club GreenSock
Examples & Showcases
Examples & Showcases
Get inspired by looking at these examples & showcases.
Type:
Showcases
Search by Title...
SEARCH
Sort By:
Popular
New
kombu
Archi Site Mobius
Mogney
Now only qr-code is needed!
Mogney
MINGLABS INTERNSHIP
Greeeg
In-Car Navigation
Rally Interactive
ultranoir
ultranoir

Greensock



SnapSVG

Scrollmagic

three.js

(example: Codrops' "The Aviator")

matter.js

Download

Latest Build

Source Code

npm package

Docs

Demos

Documentation

Wiki

License

Changelog

Info

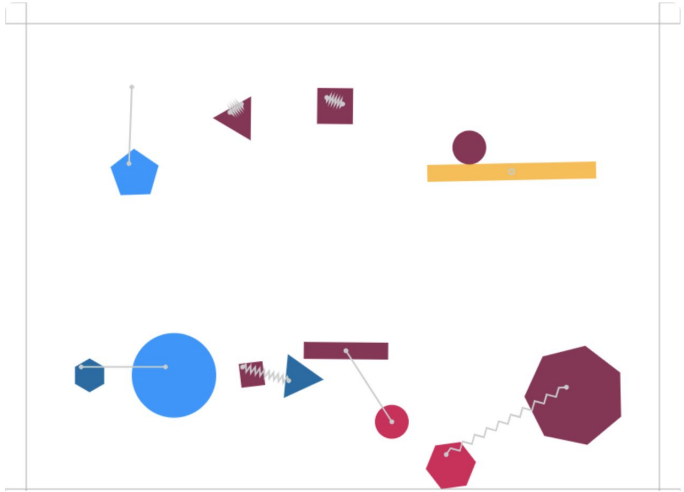
Features

Install

Usage

Matter.js is a 2D physics engine for the web

— see all demos →



matter.js


Download
Examples
Gallery
Resources
Careers
CONSULTANCY
SUPPORT US
PixiJS v4
The HTML5 Creation Engine
Create beautiful digital content with the fastest, most flexible 2D WebGL renderer.
DOWNLOAD
GET STARTED
Created by goodboy

PixiJS

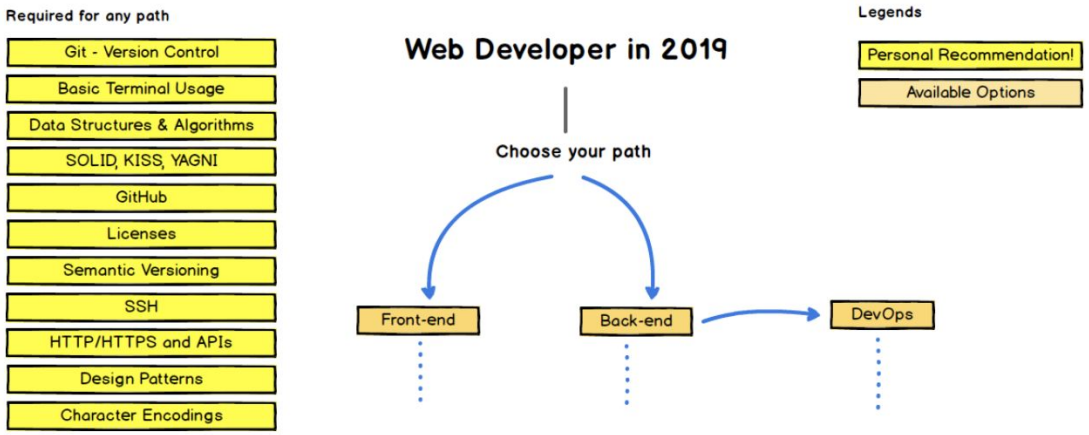
To plugin or not to plugin?

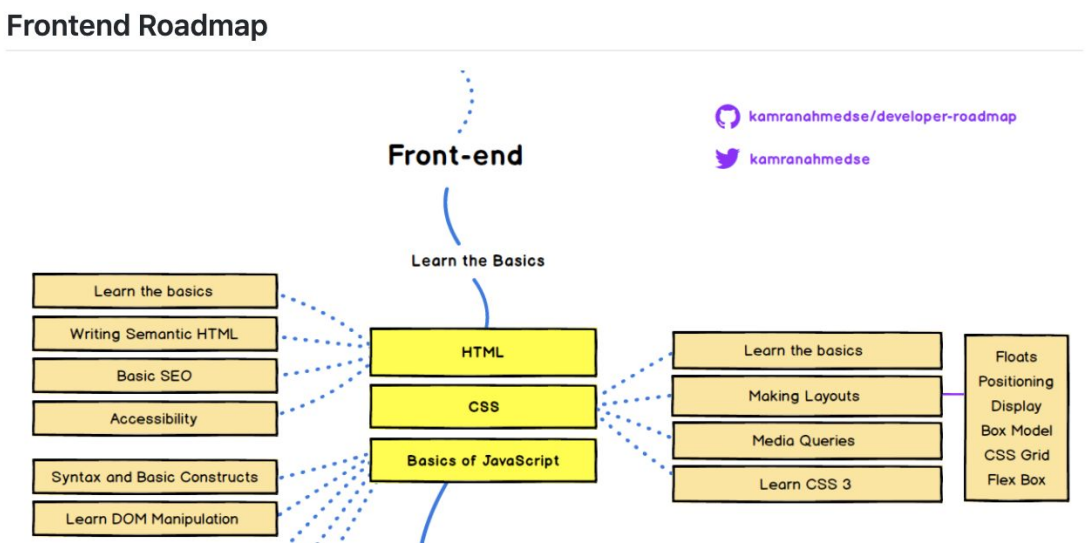
A very heated debate! A few online opinions for your perusal:

- All JavaScript frameworks are terrible
  - By the same author: JavaScript Frameworks Are Great
- Why You Shouldn't Use A Web Framework
- When not to use a JavaScript framework
- When to avoid using Javascript Libraries (Jquery)
- The deepest reason why modern JavaScript frameworks exist
- Why you shouldn’t spend one more second choosing a JS framework

To plugin or not to plugin?

- Some helpful takeaways and common arguments:
  - Choose tools that solve problems you have had and understand, not problems that you can foresee yourself having.
  - Minimize complexity--use only those tools which you truly need.
  - Reinvent the wheel sparingly. If a tool already does something well, use it! But if it does it in an unnecessarily complex way for your use case, then build your own (e.g. grid and animation systems).
  - If JS already natively has it, you might not need a plugin for it.





Onward and Upward: Your Web Dev Path

roadmap

Attendance

Questions?