FRONT END FRAMEWORKS

build web apps faster!

STORY SO FAR

CSS is a mess (but SCSS/Sass helps)

Javascript is a mess (but ES6 helps)

Web programming is way too forgiving for its own good

There are often multiple ways to do something and it’s not always clear which way is better

HOW NOT TO WRITE SPAGHETTI CODE

MODULARIZING CODE

ES5

```
// ------ myFunc.js ------
module.exports = function () { ... }

// ------- main.js -------
var myFunc = require('./myFunc')
myFunc();
```

ES6

```
// ------ myFunc.js ------
export default function () { ... }

// ------- main.js -------
import myFunc from './myFunc';
myFunc();
```

STRICT MODE

Eliminates some JavaScript silent errors by changing them to throw errors.

Fixes mistakes that make it difficult for JavaScript engines to perform optimizations

Prohibits some syntax likely to be defined in future versions of ECMAScript.

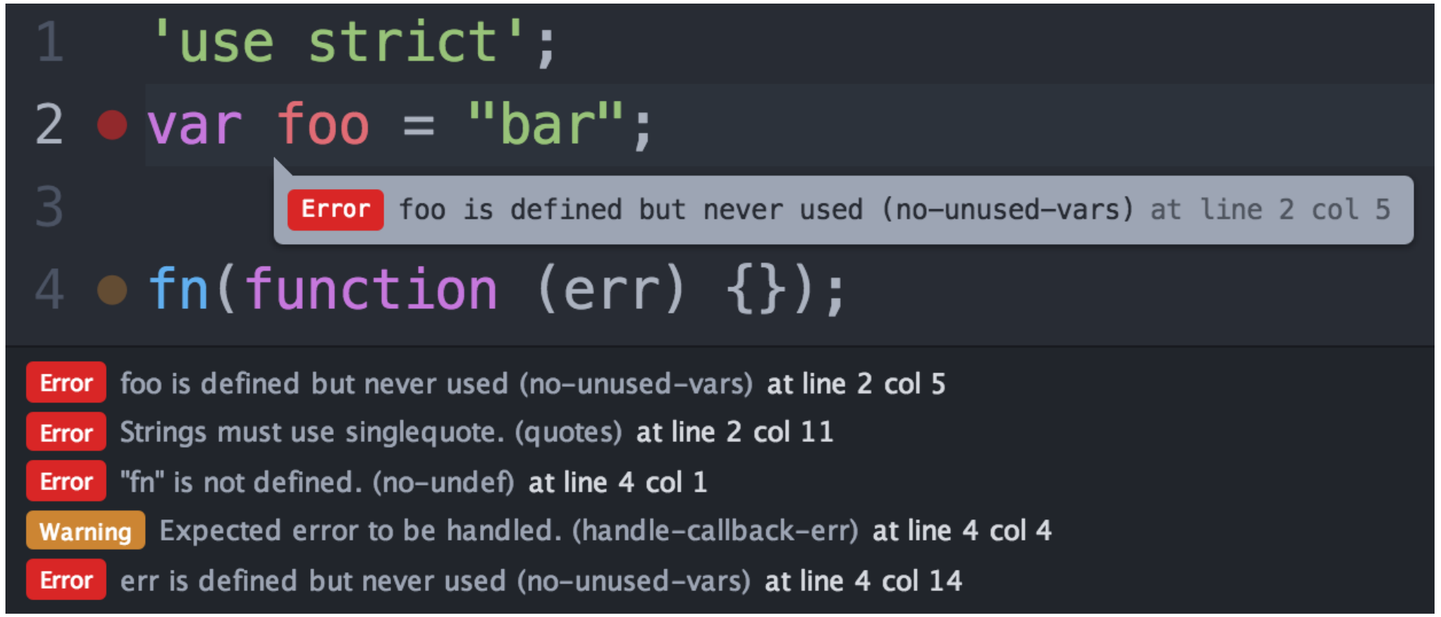
```
'use strict';

// Syntax error
var myFunc = function (a, b, b) {
  ...
}
```

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

LINTERS

built-in editor support



Used for automatically enforcing best practices, conventions

ESLint, JSLint, JSHint

PLAIN CSS AND JAVASCRIPT STILL AREN'T ENOUGH!

LIBRARIES AND PREPROCESSORS

JS Utility

Improved CSS Syntax

Bundling

https://javascript30.com/

JQUERY

Less overhead with simpler functions

Everything is used through the global variable "**$**"

Widely used for DOM manipulation

JQUERY - DOWNSIDES

Still possible to write "spaghetti code"

Painful versioning

Doesn’t offer a structure, just offers an API

Growing file size can be an overhead on load times

ES6 has faster native functions

Hides the ugly parts of JS, making it difficult to learn JS more difficult in the long run

MODERN FRONT END FRAMEWORKS

How It Feels to Learn Javascript in 2016

https://hackernoon.com/how-it-feels-to-learn-javascript-in-2016-d3a717dd577f

WHAT DO WE WANT?

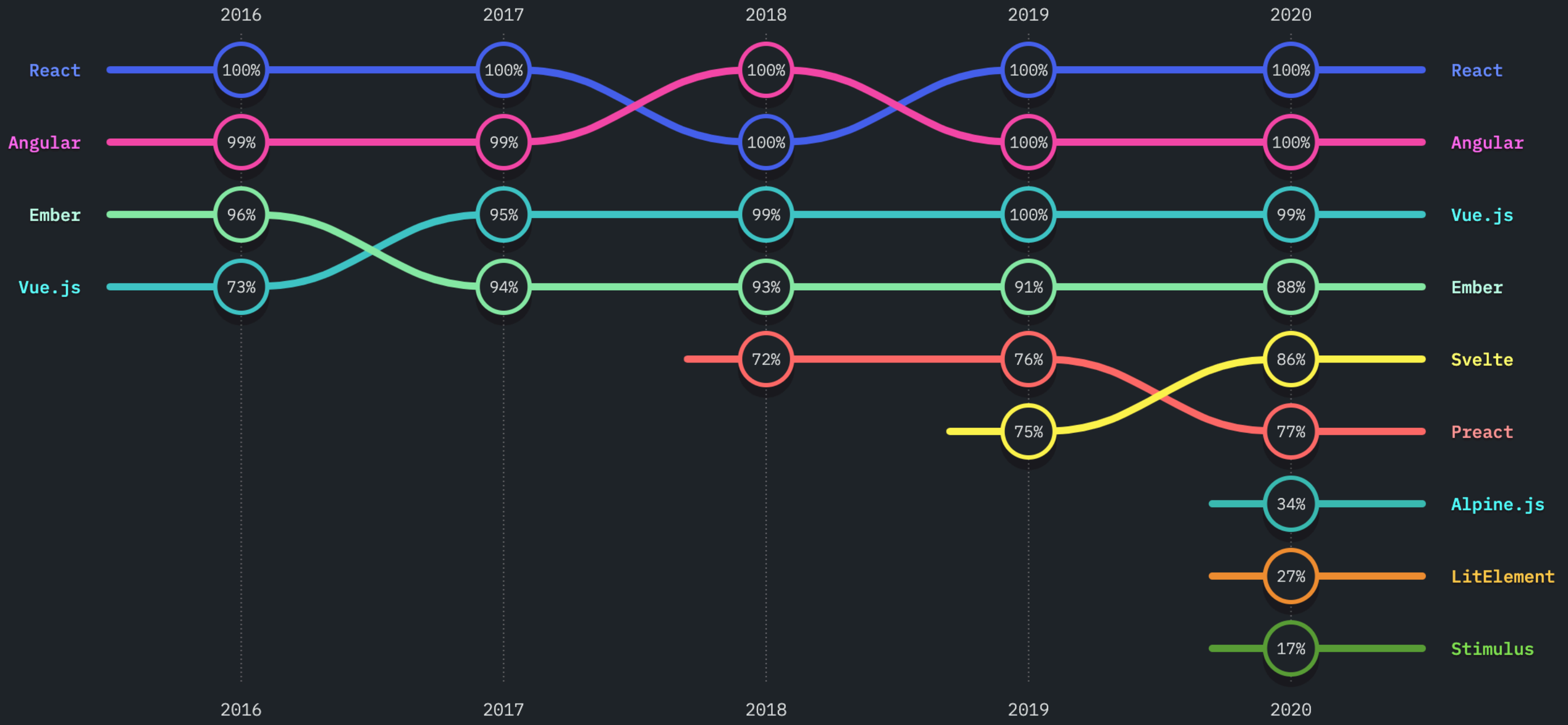
More boilerplate

Quick and Easy DOM Manipulation

Enforced guidelines & structure

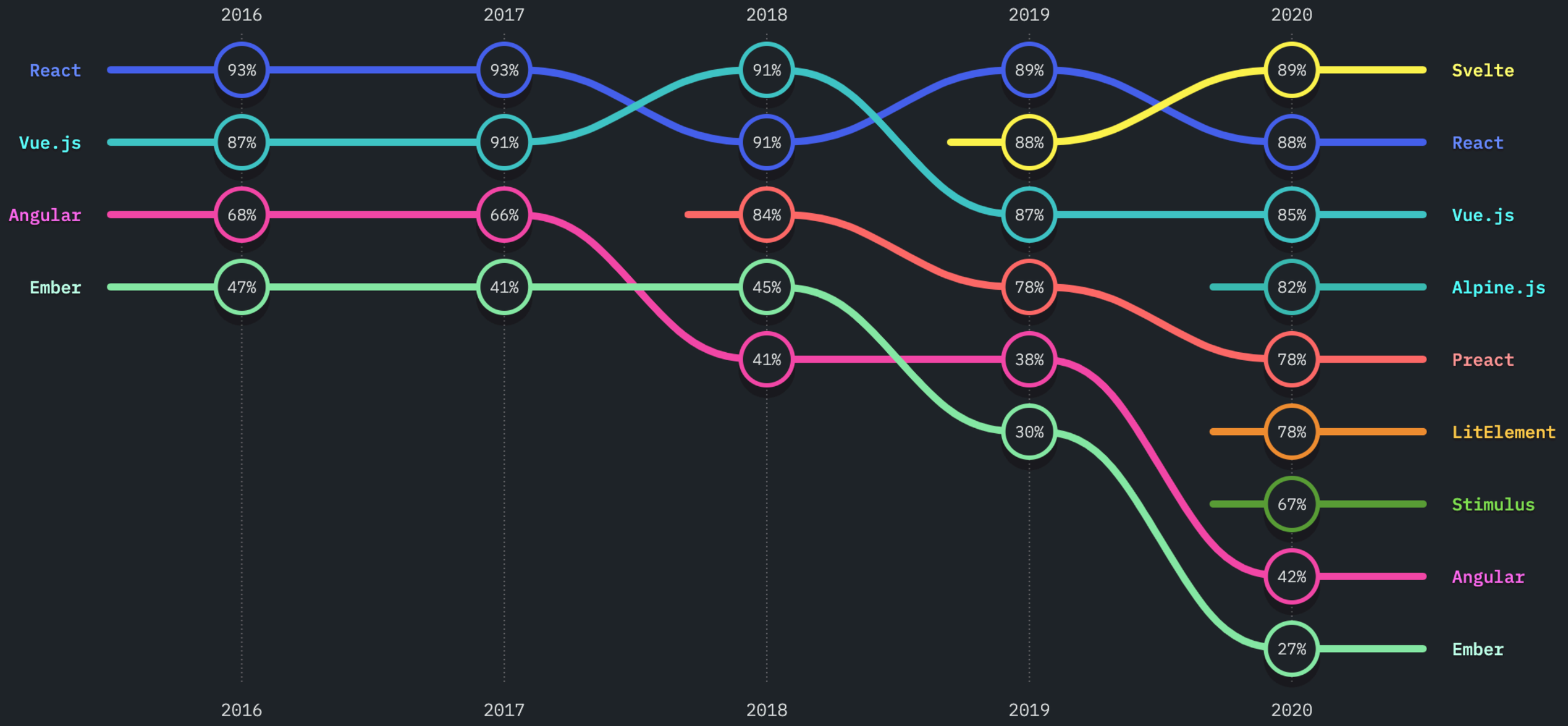
Easily build powerful web applications

AWARENESS



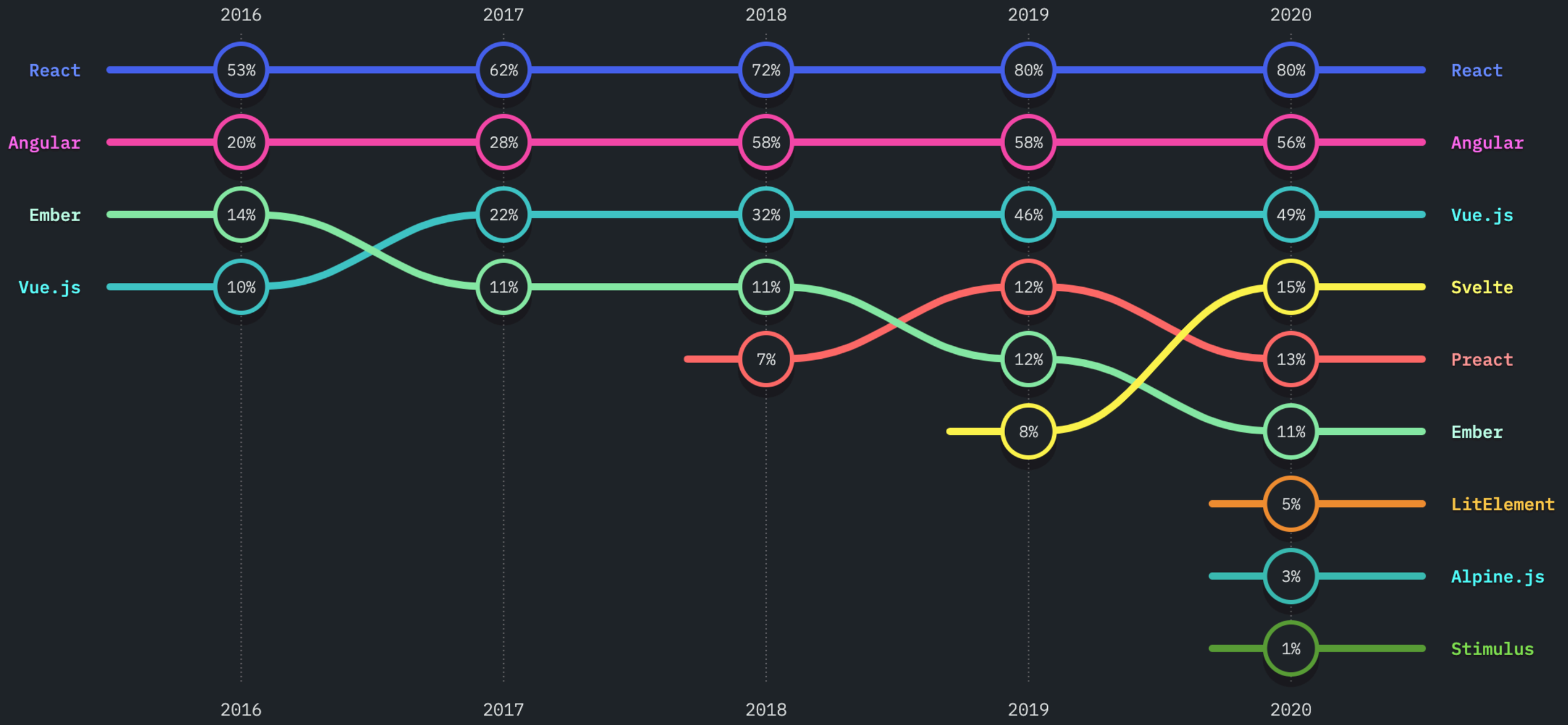
https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/

SATISFACTION



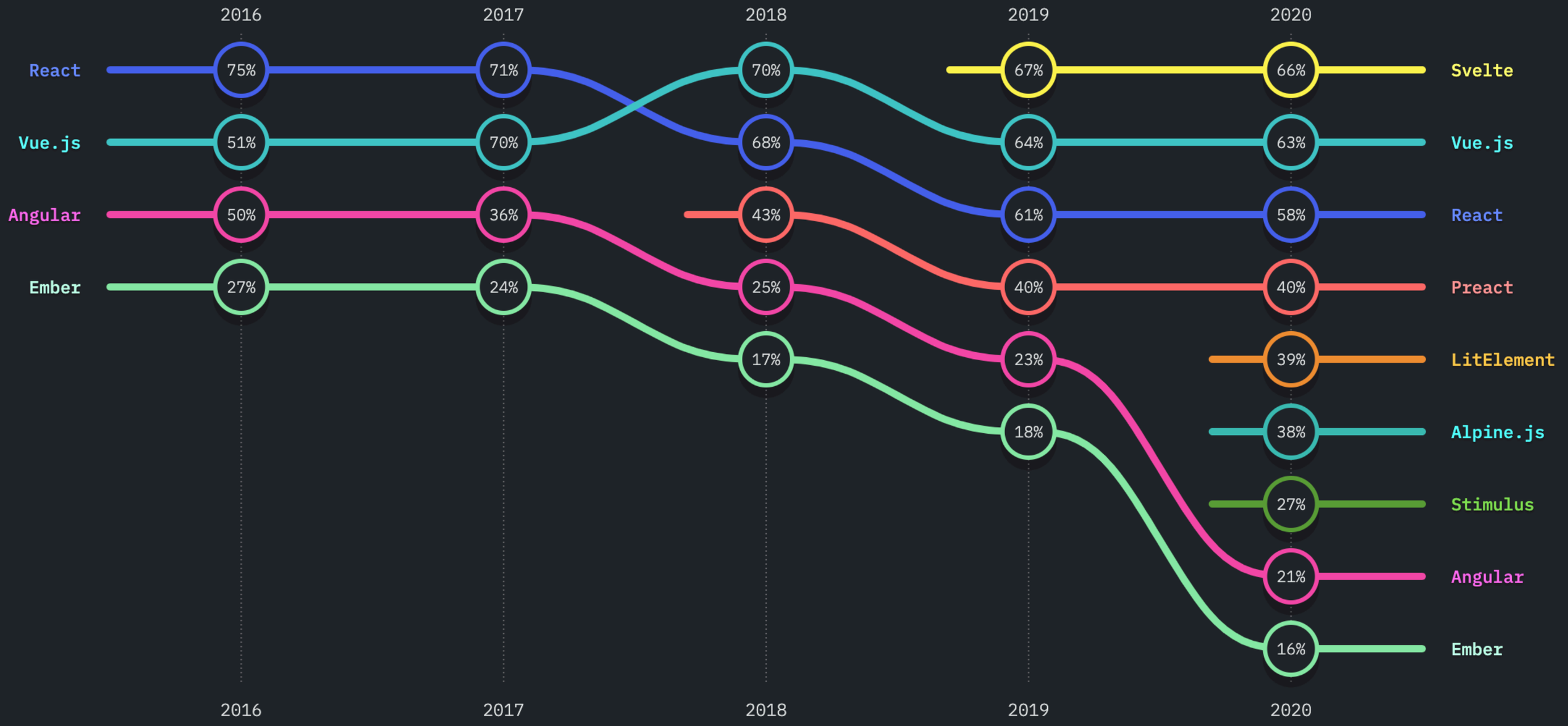
https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/

USAGE



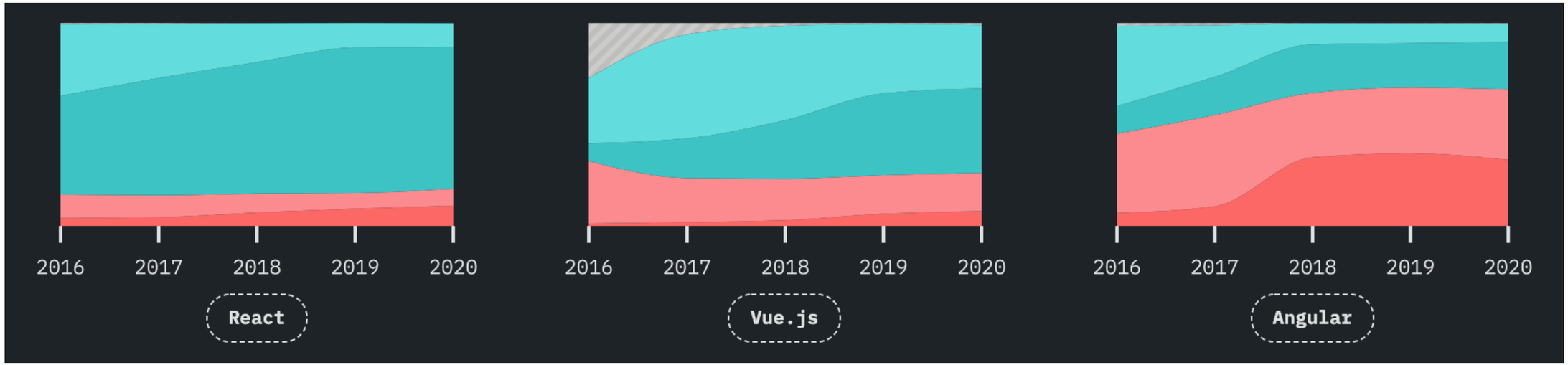
https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/

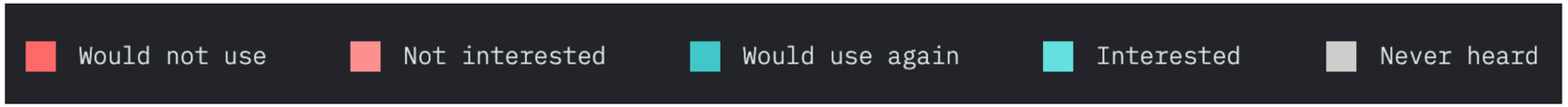
INTEREST



https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/

EXPERIENCE OVER TIME





https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/

ANGULAR

ANGULAR

Backed by Google

Popularized Single-Page Applications (**SPA**s)

Key concepts: Make it modular, testable, maintainable

Services, Factories, Controllers allowed for modularity

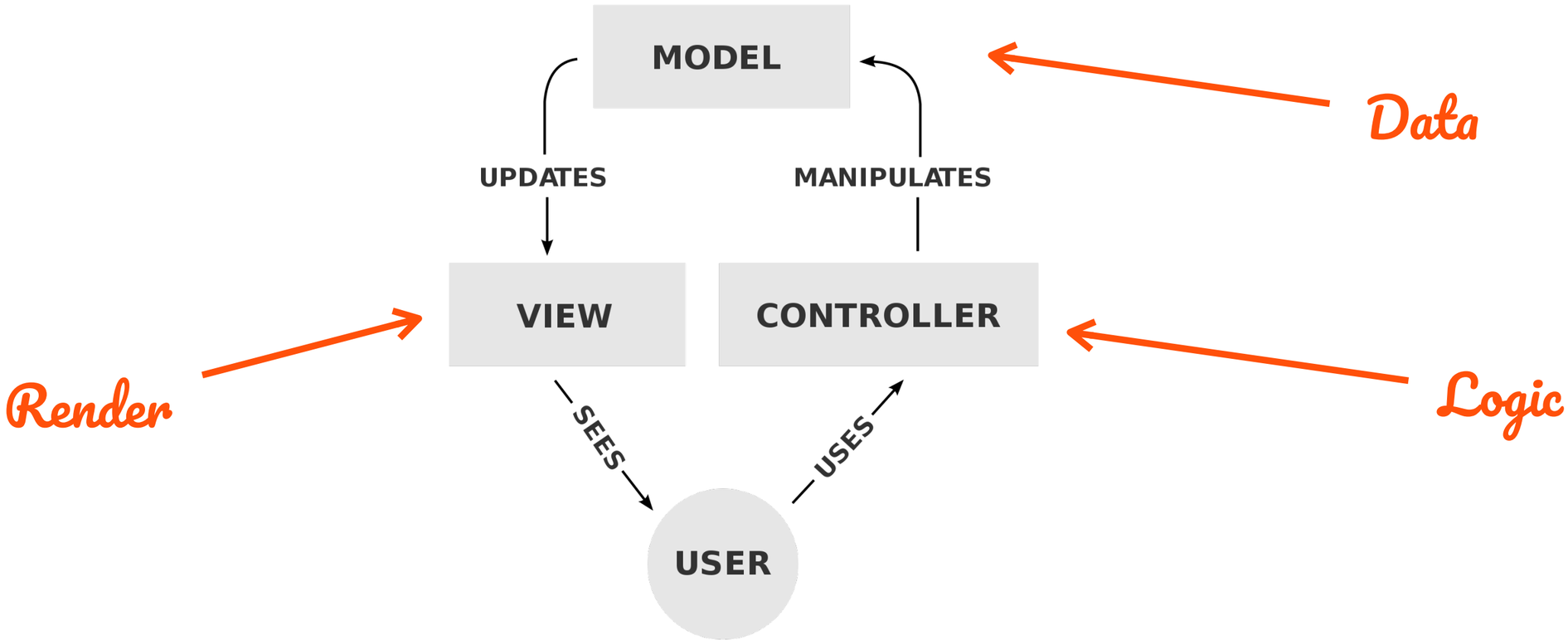
Big shifts from Angular v1 to v2 (Angular.js to Angular)

Steeper learning curve (TypeScript, Dependency Injection, etc.)

https://angular.io/

ANGULAR CONCEPTS

Model-View-Controller Paradigm



ANGULAR CONCEPTS

Two-Way Data Binding

Automatic synchronization between the model and the view

Checks for changes in the model or view and does “dirty-checking”

MVC IN ANGULAR

```
// create the module and name it
var myApp = angular.module('myApp', []);

// create the controller and inject Angular's $scope
myApp.controller('MainController', function($scope) {

    // create a message to display in our view
    $scope.message = 'Everyone come and see how good I look!';
});
```

```
<div id="main">
    {{ message }}
</div>
```

VUE

VUE

A progressive framework for building user interfaces

The core library is focused on view layer only

Easy to integrate with other libraries

Capable of supporting SPAs

Easier learning curve

Much less opinionated than Angular

https://vuejs.org/

VUE

HTML

```
<div id="app">
  {{ message }}
</div>
```

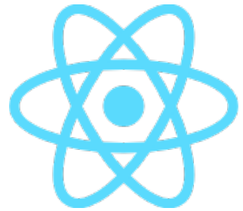
JS

```
var app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

Hello Vue!

REACT

REACT



Backed by Facebook

Technically a library, not a Framework

Unidirectional data flow

Declarative: Every component has a state with data injected into it

Component Based: Each module manages its data and views

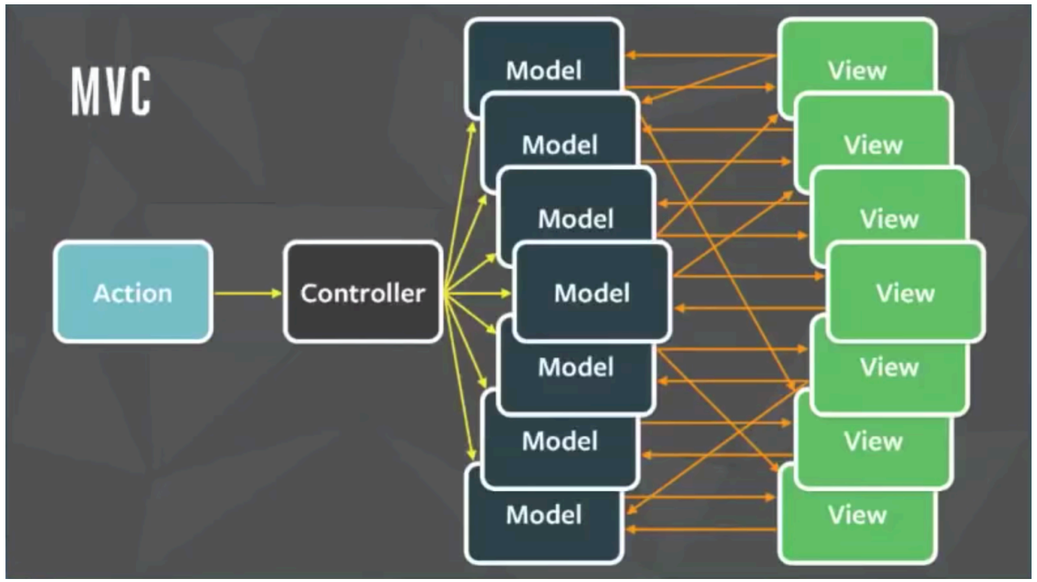
https://reactjs.org/

REACT

"React is a library for building composable user interfaces. It encourages the creation of reusable UI components which present data that changes over time."

https://facebook.github.io/react/blog/2013/06/05/why-react.html

UNIDIRECTIONAL DATA FLOW



A BACKGROUND ON PAINTING

The DOM tree is converted into pixels that are laid out onto the page to create the render tree

Reflows cause the DOM Tree to be repainted into a newly updated render tree

Behind-the-scenes computation creates new visual representations

This can be expensive and slow for our webpage! - If only there was a better way!

VIRTUAL DOM

In-memory, lightweight clone of the DOM, represented as a single JS Object

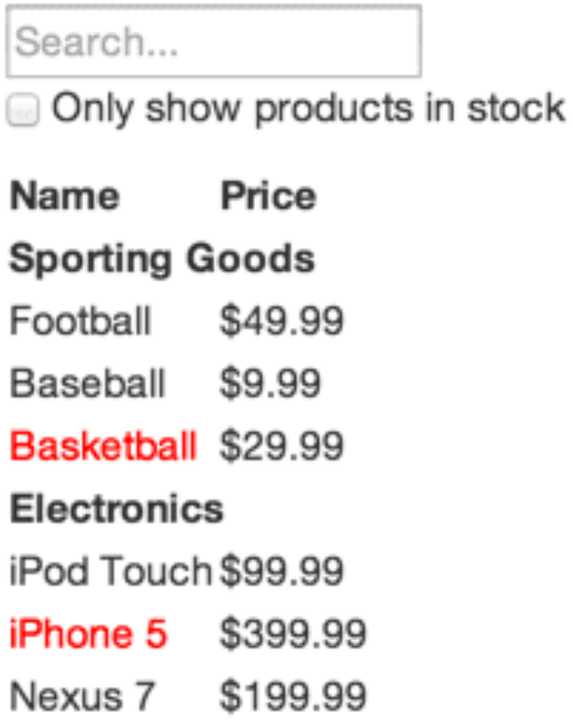
Repaint the DOM with the smallest amount of changes possible

- First, React notices that the data has changed
- React will execute the change within the lightweight Virtual DOM
- React compares the Virtual DOM with the real DOM by using “diff”
- React immediately patches changes from the Virtual DOM to the real DOM
- Avoids expensive traversing of the DOM tree

React Example 1

CodePen

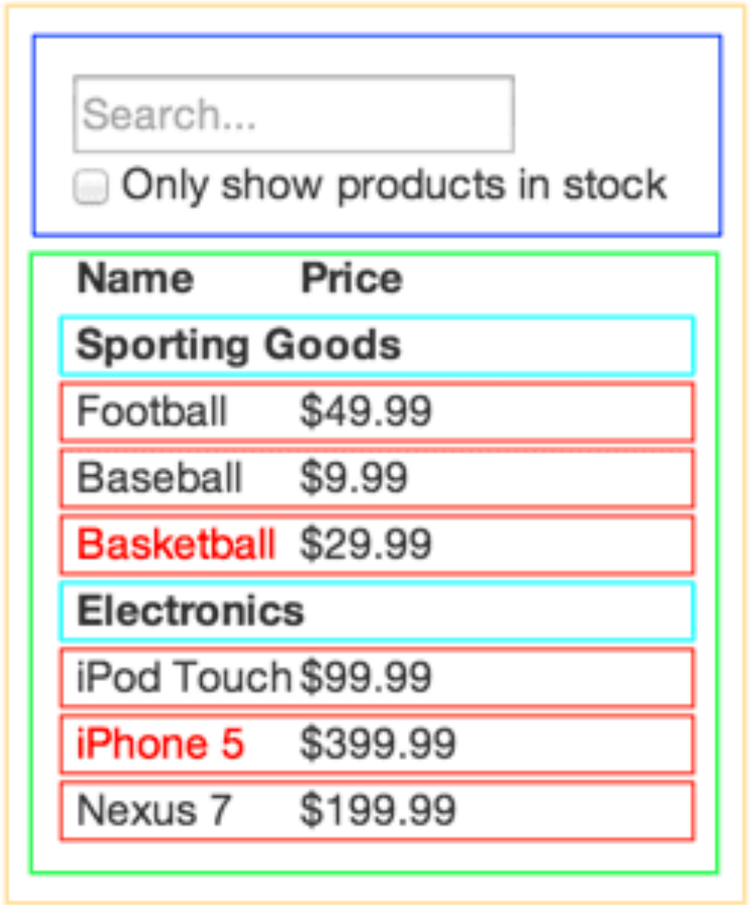
THINKING IN REACT




```
[
  {category: "Sporting Goods", price: "$49.99", stocked: true, name: "Football"},
  {category: "Sporting Goods", price: "$9.99", stocked: true, name: "Baseball"},
  {category: "Sporting Goods", price: "$29.99", stocked: false, name: "Basketball"},
  {category: "Electronics", price: "$99.99", stocked: true, name: "iPod Touch"},
  {category: "Electronics", price: "$399.99", stocked: false, name: "iPhone 5"},
  {category: "Electronics", price: "$199.99", stocked: true, name: "Nexus 7"}
];
```

https://reactjs.org/docs/thinking-in-react.html

THINKING IN REACT



FilterableProductTable (orange): contains the entirety of the example

SearchBar (blue): receives all user input

ProductTable (green): displays and filters the data collection based on user input

ProductCategoryRow (turquoise): displays a heading for each category

ProductRow (red): displays a row for each product

https://reactjs.org/docs/thinking-in-react.html

React Example 2

CodePen

NEXT CLASS: REACT

https://uiuc-web-programming.gitlab.io/fa21/