

Developer point of VUE

An introduction to Vue.js

10/10/2019



Who am I?

- Lead Software Engineer @ Intrasoft International
- Research Fellow in University of Piraeus
- Core contributor in various open source projects (Vaadin Flow, Drupal)
- Ambassador of OracleJET and Prestashop

Email: panagiotis.adamopoulos@intrasoft-intl.com

Github: <https://github.com/panosadamop>

Linkedin: <https://www.linkedin.com/in/padamopoulos/>



An abstract graphic on a solid blue background. On the left side, there are several vertical, rounded rectangular bars of varying heights and shades of blue. A thin, dark blue horizontal line crosses the entire width of the image, positioned roughly in the middle. The text 'The web evolution' is written in white on the right side of the image.

The web evolution

The web evolution

- HTML specification was made public in late 1991 by Tim Berners-Lee.
- CSS was proposed by Håkon Wium Lie in October 1994



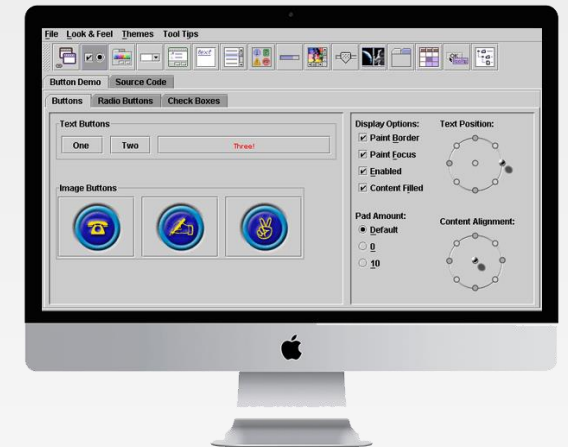
1997 - 200x

- Static Sites, Javascript, VBScript
- Flash Websites
- Dialup and slow ADSL connections



Around 2007 - 2008

- Web application made a trend
- .NET, php, jsp, database usage, CMS
- Fast internet connections



Today

- Various Frameworks
- MicroServices & MicroApps
- Enterprise Web Applications

The web evolution

Year (June)	Websites
2018	1,630,322,579
2017	1,766,926,408
2016	1,045,534,808
2011	346,004,403
2010	206,956,723
2009	238,027,855
2001	29,254,370
2000	17,087,182
1999	3,177,453
1995	23,500
1994	2,738



CSS Frameworks (first attempt for CSS with ACID tests)

- In mid-2000s, CSS libraries and frameworks began cropping up (Blueprint, 960gs)

Javascript Frameworks

- **Jquery released on August 2006** and standardized cross-browser compatibility and AJAX technology(existed since 1996)
- **AngularJS came into market in 2010.** After that we had Bower, Gulp, Webpack, Backbone, Knockout, React, Vue etc

Today we are talking about Micro-Frontends, MicroApps, FrontEnd Architectures etc



**Frameworks are designed
to help us deal with
complexity.**



**...but frameworks themselves
also introduce complexity of their own.**



Application Complexity vs. Framework Complexity





Pick the right tool
for the job



The background features a light gray gradient. A thin, dark gray horizontal line spans the width of the image. On the left side, there are several vertical bars of varying heights and widths. These bars are in two colors: a medium gray and a light blue. They are arranged in a way that some appear to be behind the horizontal line and others in front, creating a layered effect. The text 'VUE.JS' is positioned to the right of these bars, centered vertically relative to the horizontal line.

VUE.JS



- Creator @Vue.js
- Core Dev at Meteor
- Previously worked as a Creative Technologist at Google
- From 2016 working full-time on Vue.JS framework.

Twitter: <https://twitter.com/youyuxi>

Github: <https://github.com/yyx990803/>



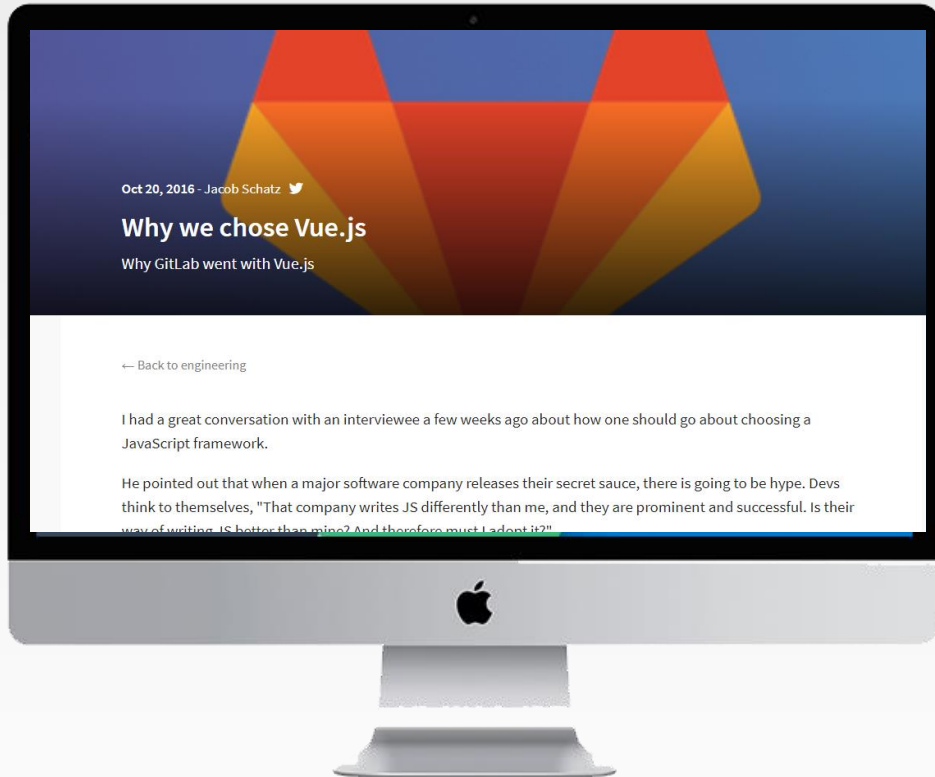
- Started in late 2013
- First release Feb. 2014 (v0.6)
- v1.0.0 Evangelion Oct. 2015
- Latest release v2.5.16





- Vue is a progressive framework for building user interfaces.
- Vue is designed from the ground up to be incrementally adoptable.
- The core library is focused on the view layer only.
- It is easy to pick up and integrate with other libraries or existing projects.

Who is using VUE.JS

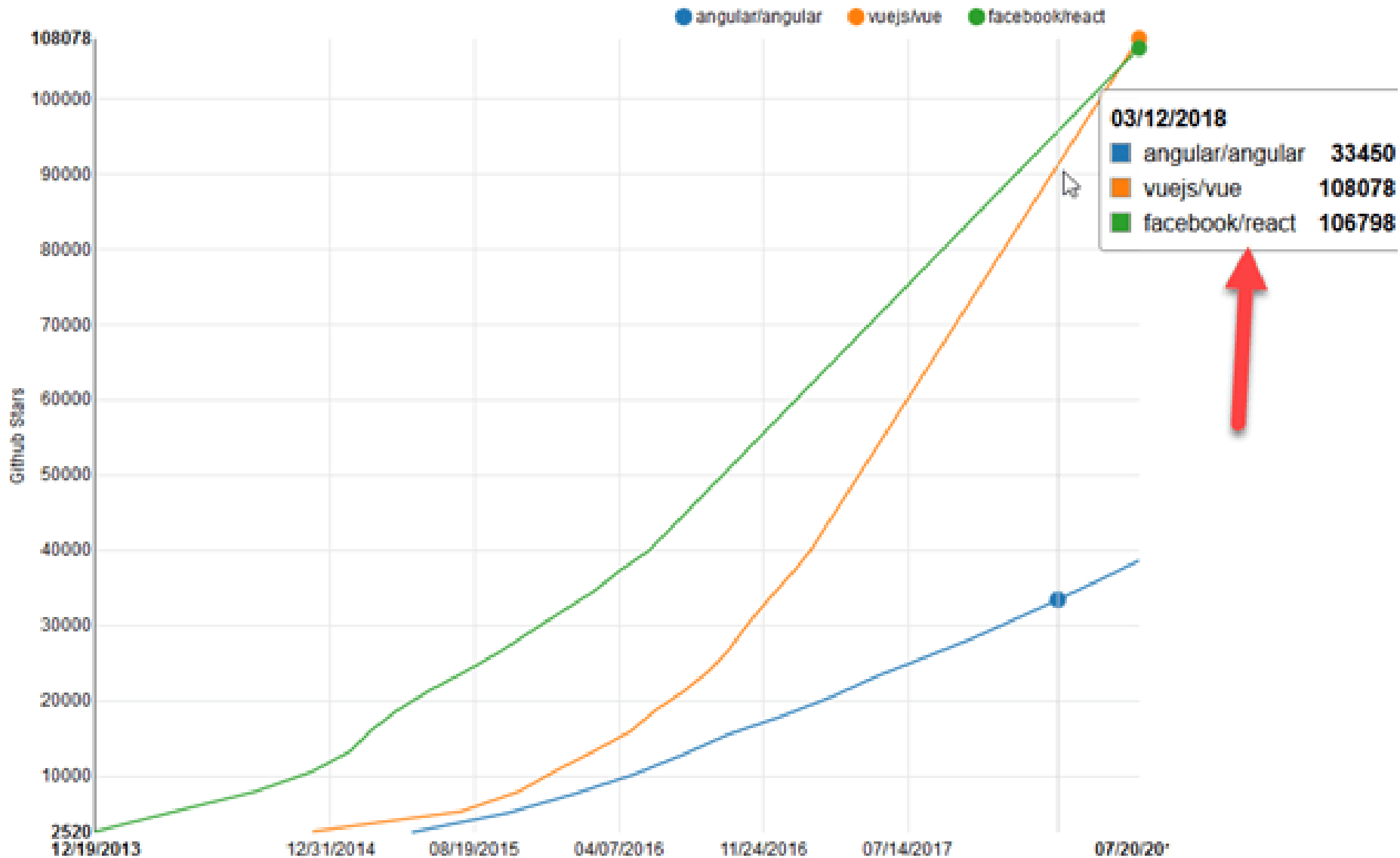


- GitLab
(<https://about.gitlab.com/2016/10/20/why-we-chose-vue/>)
- Weex (<https://weex.apache.org/>) Maintained by Alibaba Group)
- Baidu(Chinese search engine)
- Grammarly
- WizzAir
- Alibaba
- Xiaomi
- Facebook etc

An abstract graphic on the left side of the slide. It features a horizontal line that divides the slide in half. Below this line, there are several vertical bars of varying heights and widths. Some bars are dark gray, while others are light blue. The bars are positioned at different intervals along the horizontal axis, creating a sense of depth and movement. The background is a solid light gray.

Statistics & Benchmarks

VueJS Statistics

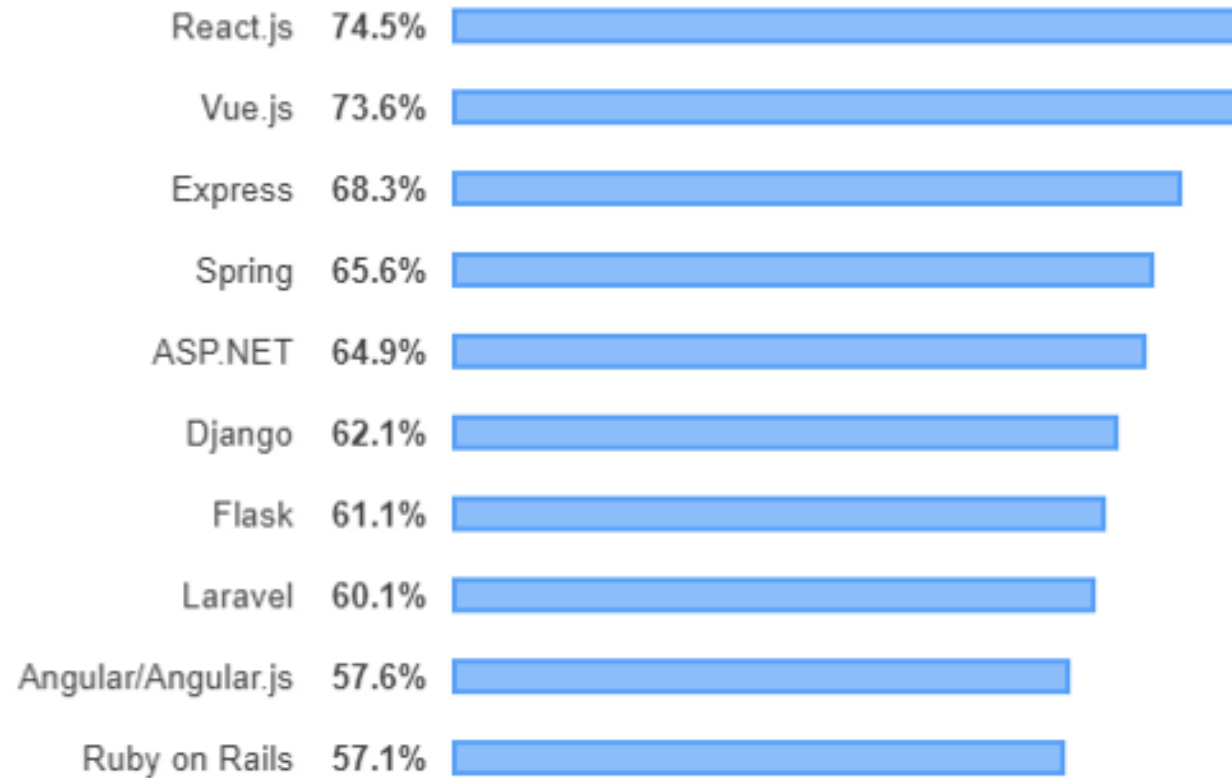


Most Loved, Dreaded, and Wanted Web Frameworks

Loved

Dreaded

Wanted



VueJS Benchmarks

DOM Manipulation (ms)

Name	preact-v8.3.1-keyed	inferno-v5.6.1-keyed	vue-v2.5.17-keyed	react-v16.5.2-keyed	react-redux-v16.1.0 + 3.7.2-keyed	angular-optimized-v6.1.0-keyed
create rows Duration for creating 1000 rows after the page loaded.	217.1 ± 1.0 (1.3)	161.1 ± 9.1 (1.0)	251.1 ± 29.8 (1.6)	235.4 ± 23.4 (1.5)	271.8 ± 14.6 (1.7)	198.7 ± 12.2 (1.2)
replace all rows Duration for updating all 1000 rows of the table (with 5 warmup iterations).	183.9 ± 3.6 (1.2)	158.0 ± 2.4 (1.0)	186.3 ± 17.6 (1.2)	196.1 ± 21.3 (1.2)	216.0 ± 16.0 (1.4)	205.7 ± 9.4 (1.3)
partial update Time to update the text of every 10th row (with 5 warmup iterations) for a table with 10k rows.	110.1 ± 2.7 (1.5)	76.2 ± 6.7 (1.0)	201.7 ± 39.6 (2.8)	88.8 ± 2.8 (1.2)	114.7 ± 4.1 (1.6)	72.6 ± 3.6 (1.0)
select row Duration to highlight a row in response to a click on the row. (with 5 warmup iterations).	12.5 ± 6.4 (1.0)	5.4 ± 4.6 (1.0)	10.9 ± 2.3 (1.0)	8.0 ± 7.1 (1.0)	12.3 ± 2.1 (1.0)	5.5 ± 3.7 (1.0)
swap rows Time to swap 2 rows on a 1K table. (with 5 warmup iterations).	21.5 ± 6.2 (1.3)	15.8 ± 4.6 (1.0)	23.4 ± 4.5 (1.5)	119.3 ± 3.0 (7.5)	153.4 ± 48.7 (9.6)	19.6 ± 3.2 (1.2)
remove row Duration to remove a row. (with 5 warmup iterations).	53.3 ± 1.7 (1.1)	49.1 ± 1.1 (1.0)	57.0 ± 1.6 (1.2)	61.9 ± 7.2 (1.3)	57.2 ± 3.1 (1.2)	49.5 ± 0.9 (1.0)
create many rows Duration to create 10,000 rows	2,112.7 ± 101.8 (1.4)	1,563.7 ± 37.7 (1.0)	1,829.1 ± 91.4 (1.2)	2,727.3 ± 1095.2 (1.7)	2,617.3 ± 52.3 (1.7)	1,690.5 ± 111.4 (1.1)
append rows to large table Duration for adding 1000 rows on a table of 10,000 rows.	313.3 ± 6.5 (1.3)	241.1 ± 7.2 (1.0)	408.0 ± 10.9 (1.7)	302.3 ± 11.2 (1.3)	355.7 ± 38.9 (1.5)	265.7 ± 14.9 (1.1)
slowdown geometric mean	1.25	1.01	1.43	1.61	1.78	1.11

Startup metrics

Name	preact-v8.3.1-keyed	inferno-v5.6.1-keyed	vue-v2.5.17-keyed	react-v16.5.2-keyed	react-redux-v16.1.0 + 3.7.2-keyed	angular-optimized-v6.1.0-keyed
consistently interactive a pessimistic TTI - when the CPU and network are both definitely very idle. (no more CPU tasks over 50ms)	1,954.4 ± 0.5 (1.0)	2,029.2 ± 0.1 (1.0)	2,253.5 ± 0.1 (1.2)	2,479.1 ± 0.7 (1.3)	2,893.3 ± 75.4 (1.5)	2,855.5 ± 0.8 (1.5)
script bootstrap time the total ms required to parse/compile/evaluate all the page's scripts	16.0 ± 0.0 (1.0)	16.0 ± 0.0 (1.0)	62.6 ± 12.2 (3.9)	81.5 ± 26.7 (5.1)	137.7 ± 34.8 (8.6)	96.0 ± 26.8 (6.0)
main thread work cost total amount of time spent doing work on the main thread. Includes style/layout/etc.	457.0 ± 107.6 (1.0)	449.5 ± 3.9 (1.0)	512.9 ± 6.8 (1.1)	546.6 ± 39.9 (1.2)	638.3 ± 95.2 (1.4)	561.9 ± 79.5 (1.3)
total byte weight network transfer cost (post-compression) of all the resources loaded into the page.	155,751.0 ± 0.0 (1.0)	165,849.0 ± 0.0 (1.1)	215,463.0 ± 0.0 (1.4)	253,853.0 ± 0.0 (1.6)	328,959.0 ± 0.0 (2.1)	331,461.0 ± 0.0 (2.1)

Memory Allocation (MB)

Name	preact-v8.3.1-keyed	inferno-v5.6.1-keyed	vue-v2.5.17-keyed	react-v16.5.2-keyed	react-redux-v16.1.0 + 3.7.2-keyed	angular-optimized-v6.1.0-keyed
ready memory Memory usage after page load.	2.3 ± 0.3 (1.0)	2.3 ± 0.3 (1.0)	2.6 ± 0.3 (1.1)	2.8 ± 0.2 (1.2)	3.2 ± 0.4 (1.4)	3.3 ± 0.2 (1.4)
run memory Memory usage after adding 1000 rows.	5.3 ± 0.0 (1.1)	4.7 ± 0.0 (1.0)	7.1 ± 0.0 (1.5)	6.8 ± 0.0 (1.4)	7.7 ± 0.0 (1.6)	6.0 ± 0.0 (1.3)
update each 10th row for 1k rows (5 cycles) Memory usage after clicking update every 10th row 5 times	5.3 ± 0.0 (1.1)	4.8 ± 0.0 (1.0)	7.2 ± 0.0 (1.5)	7.6 ± 0.0 (1.6)	8.7 ± 0.1 (1.8)	6.1 ± 0.0 (1.3)
replace 1k rows (5 cycles) Memory usage after clicking create 1000 rows 5 times	8.1 ± 0.0 (1.7)	4.8 ± 0.0 (1.0)	7.2 ± 0.0 (1.5)	8.1 ± 0.0 (1.7)	9.9 ± 0.1 (2.1)	6.4 ± 0.0 (1.3)
creating/clearing 1k rows (5 cycles) Memory usage after creating and clearing 1000 rows 5 times	5.4 ± 0.0 (2.1)	2.6 ± 0.0 (1.0)	3.0 ± 0.0 (1.1)	3.8 ± 0.0 (1.4)	4.9 ± 0.0 (1.9)	3.9 ± 0.0 (1.5)

An abstract graphic on a solid blue background. A thin horizontal line crosses the middle of the frame. To the left of this line, several vertical, rounded rectangular bars of varying heights and shades of blue (dark blue and light blue) are positioned. Some bars extend above the line, while others extend below it. The word "Advantages" is written in white text to the right of the horizontal line.

Advantages

- Very Small Size
- Easy to Understand and Develop
- Simple Integration
- Detailed Documentation
- Flexibility
- Component oriented
- Mix the best of React and Angular



Very Small Size

Name	Size(gzipped)
React + react-dom	34.4 kB
Angular	62.2 kB
jQuery	31.1 kB
Vue.js	23.8 kB



Ease of development

```
1 <template>
2   <div>
3     <h1>Todos</h1>
4     <div v-for="todo in todos">
5       <ToDoItem v-bind:todo="todo" />
6     </div>
7   </div>
8
9 </template>
10
11 <script>
12   import ToDoItem from "../ToDoItem";
13   export default {
14     name: "Todos",
15     components: {
16       ToDoItem
17     },
18     props: ["todos"]
19   }
20 </script>
21
22 <style scoped>
23
24 </style>
25 |
```

- tag script (Vue.js runtime)
- webpack
- web components
- browserify
- requirejs





Explanation of Different Builds

In the `dist/` directory of the NPM package you will find many different builds of Vue.js. Here's an overview of the difference between them:

	UMD	CommonJS	ES Module
Full	vue.js	vue.common.js	vue.esm.js
Runtime-only	vue.runtime.js	vue.runtime.common.js	vue.runtime.esm.js
Full (production)	vue.min.js	-	-
Runtime-only (production)	vue.runtime.min.js	-	-

Terms

- **Full:** builds that contains both the compiler and the runtime.

An abstract graphic on a solid blue background. A thin horizontal line crosses the middle of the frame. To the left of this line, several vertical, rounded rectangular bars of varying heights and shades of blue (dark blue and light blue) are positioned. Some bars are solid, while others are semi-transparent, creating a layered effect. The word "Ecosystem" is written in white, sans-serif font to the right of the horizontal line.

Ecosystem

Project	Description
vue-router	Single-page application routing
vuex	Large-scale state management
vue-cli	Project scaffolding
vue-loader	Single File Component (*.vue file) loader for webpack
vue-server-renderer	Server-side rendering support
vue-class-component	TypeScript decorator for a class-based API
vue-rx	RxJS integration
vue-devtools	Browser DevTools extension

- NativeScript with Vue.js
- Apache Weex - Mobile cross-platform UIs
- Nuxt - Server Side Rendering (SSR)

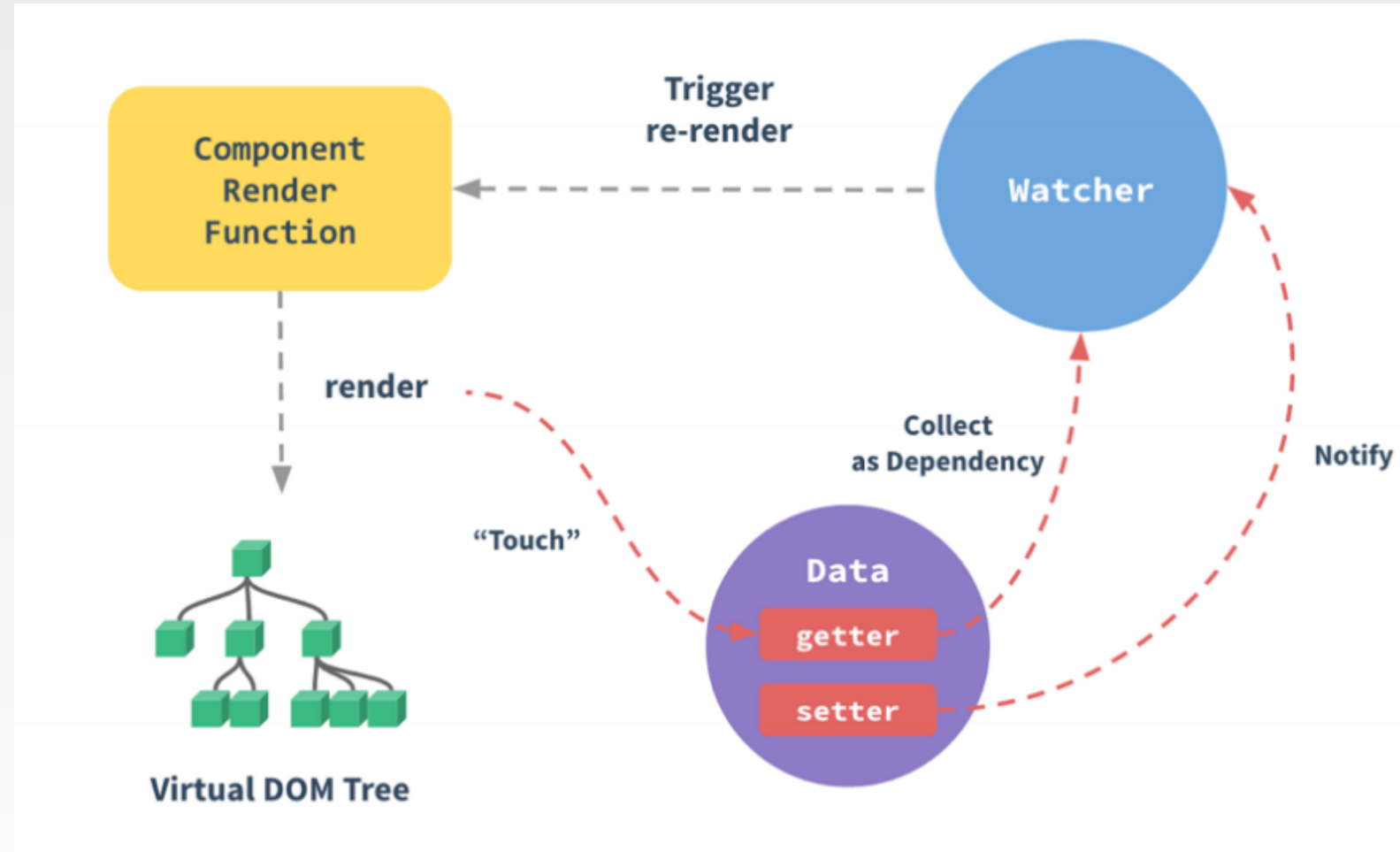


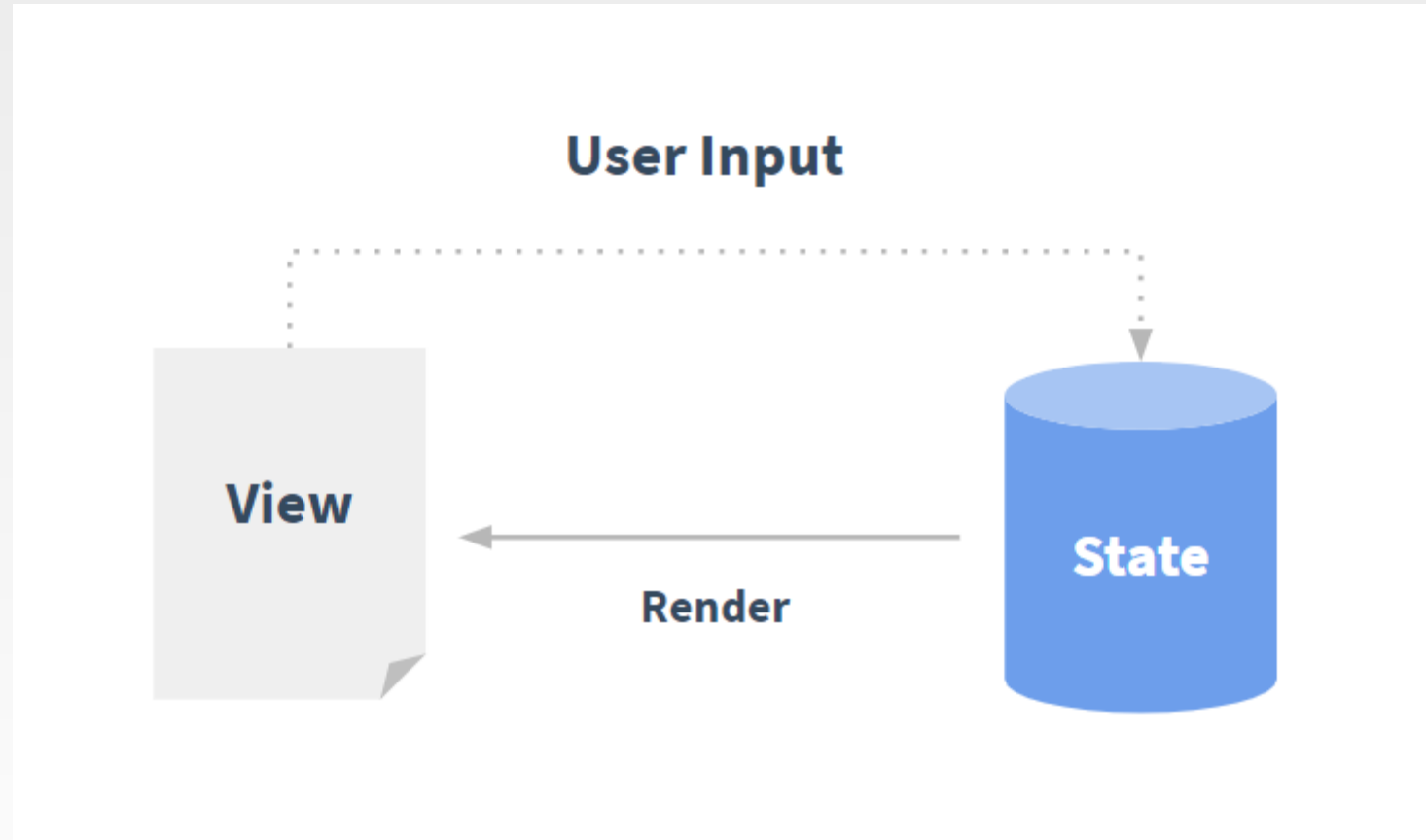
The background is a solid dark blue. On the left side, there are several vertical, rounded rectangular bars of varying heights and widths, in different shades of blue. A thin, light blue horizontal line crosses the entire width of the image, positioned slightly below the vertical center.

How it works

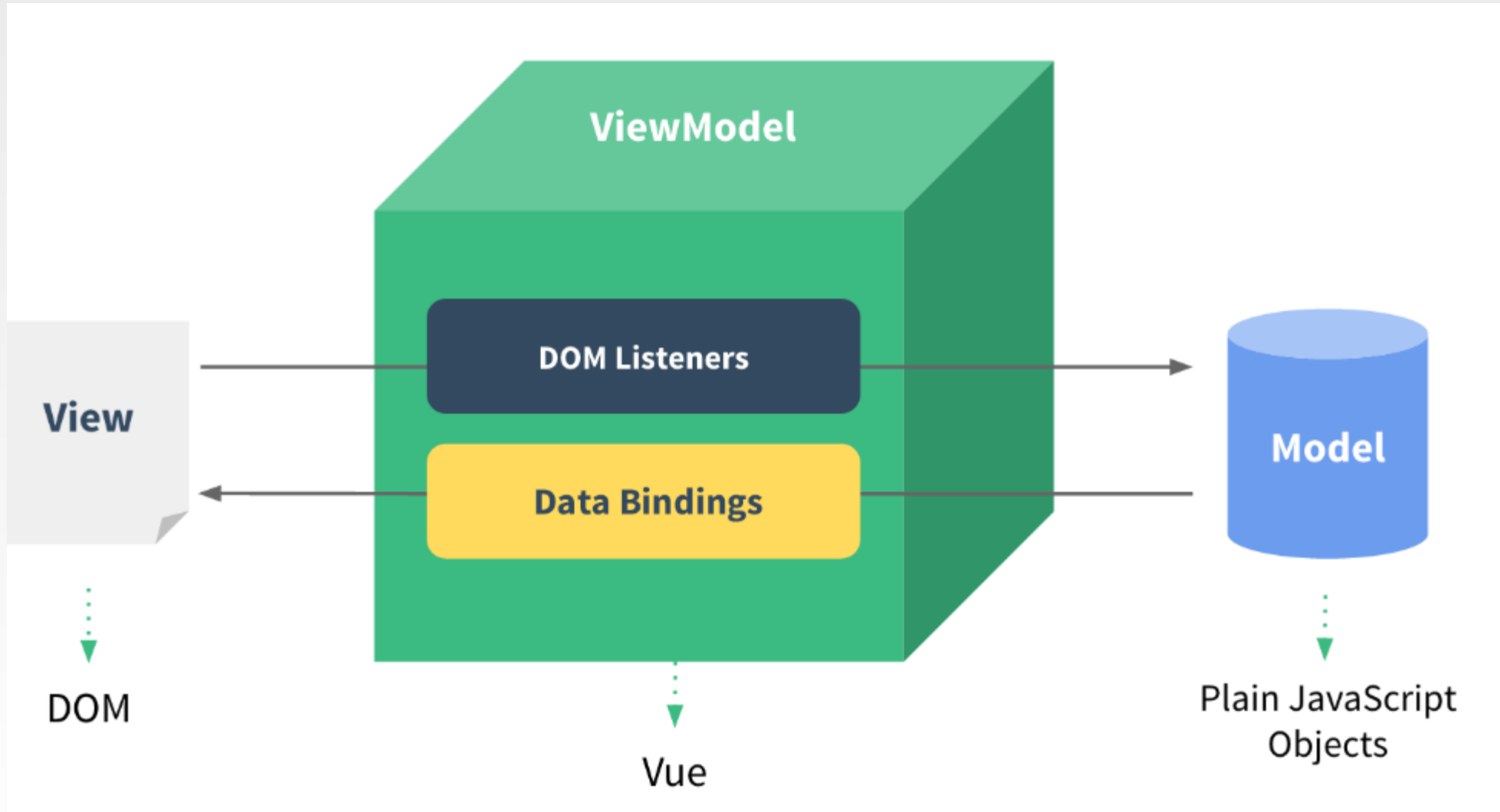
How it works

- Inspired by Model-view-viewModel (MVVM) architectural pattern
- Uses Declarative Rendering
- Dependency tracking system with getter/setters

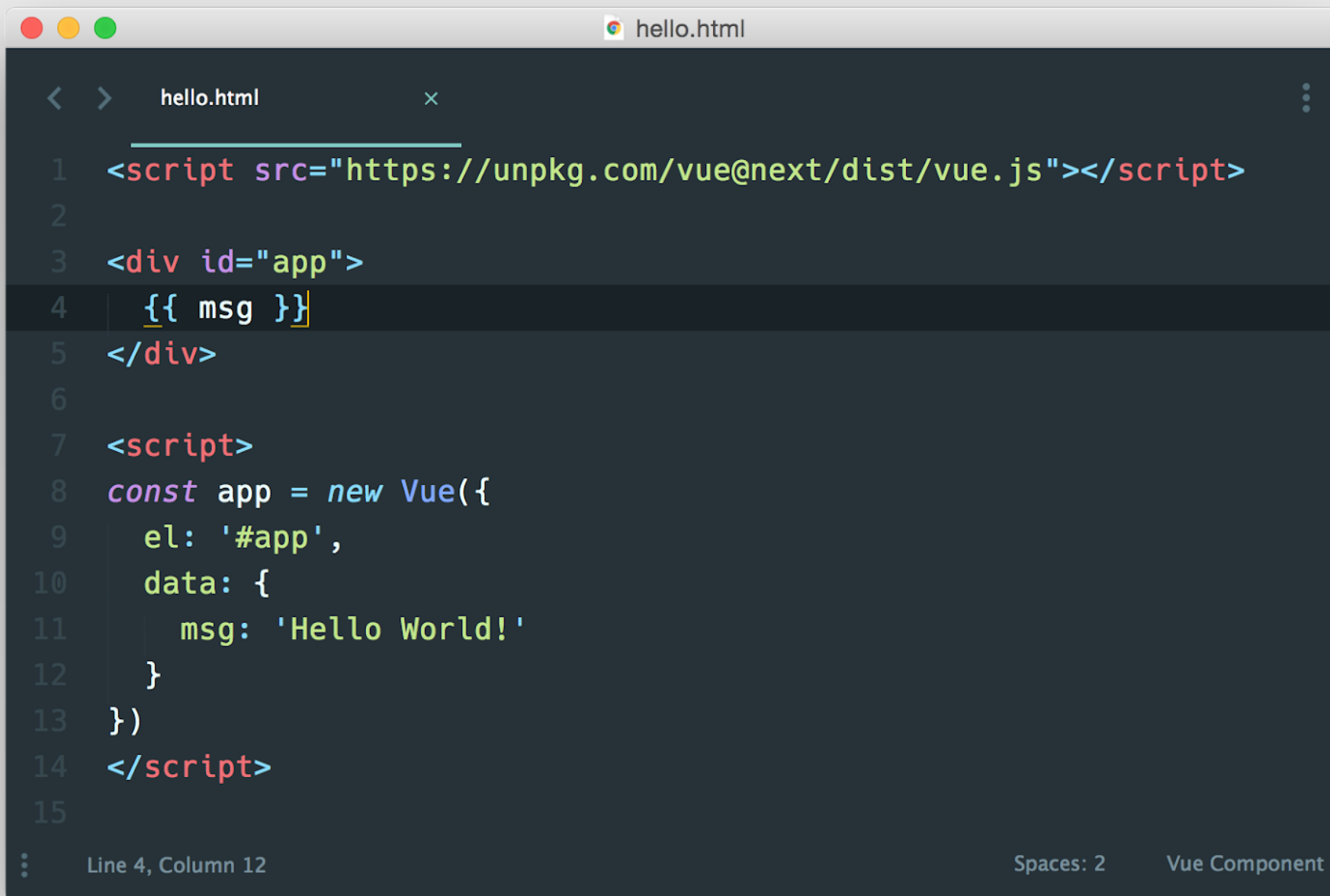




Reactive two way Data Binding



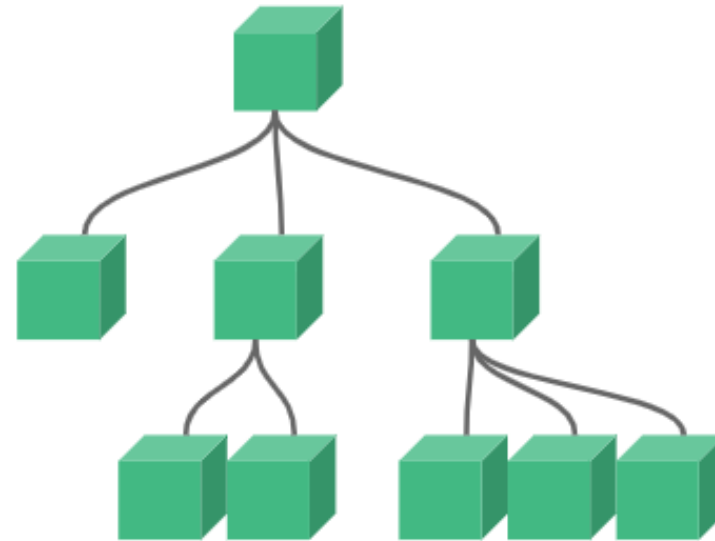
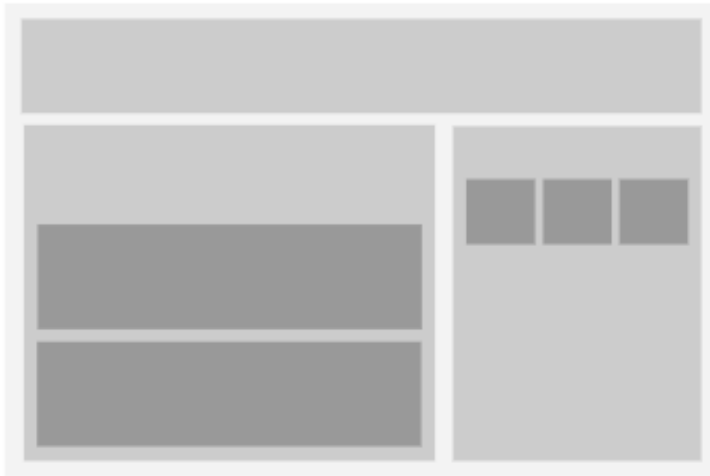
Declarative Rendering



```
1 <script src="https://unpkg.com/vue@next/dist/vue.js"></script>
2
3 <div id="app">
4   {{ msg }}
5 </div>
6
7 <script>
8   const app = new Vue({
9     el: '#app',
10    data: {
11      msg: 'Hello World!'
12    }
13  })
14 </script>
15
```

Line 4, Column 12

Spaces: 2 Vue Component



Component System

```
1 <template lang="jade">
2   div.my-component
3     h1 {{ msg }}
4     other-component
5 </template>
6
7 <script>
8   import OtherComponent from './other-component.vue'
9
10  export default {
11    components: { OtherComponent },
12    data() {
13      return {
14        msg: 'Hello Vue.js!'
15      }
16    }
17  }
18 </script>
19
20 <style lang="sass" scoped>
21   $font-stack: Helvetica, sans-serif;
22   $primary-color: #333;
23
24   .my-component {
25     font: 100% $font-stack;
26     color: $primary-color;
27   }
28 </style>
```

Single File Vue Components

```
1 <template lang="jade">
2   div.my-component
3     h1 {{ msg }}
4     other-component
5 </template>
6
7 <script>
8   import OtherComponent from './other-component.vue'
9
10  export default {
11    components: { OtherComponent },
12    data() {
13      return {
14        msg: 'Hello Vue.js!'
15      }
16    }
17  }
18 </script>
19
20 <style lang="sass" scoped>
21   $font-stack: Helvetica, sans-serif;
22   $primary-color: #333;
23
24   .my-component {
25     font: 100% $font-stack;
26     color: $primary-color;
27   }
28 </style>
```

- Collocation of Template, Logic & Style
- Just use HTML, CSS & JavaScript / TypeScript
- Embedded pre-processor support: seamlessly use Babel, SASS etc in the same file
- Hot-reload out of the box
- Scoped CSS / CSS Modules

Some code examples

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Vue.js</title>
  <script src="vue.js"></script>
</head>
<body>
  <div id="selector">
    <h1>{{ message }}</h1>
  </div>
</body>
</html>
```

```
// Define a plain JSON
// object to hold the data
var data = {
  message: "Hello"
};

// Instantiate Vue on an element
new Vue ({
  el: "#selector",
  data: data
})

// Works since getters/setters
// are now proxied
data.message = "Hi";
```

Directives

v-bind

```
<!-- full syntax -->  
<a v-bind:href="url"></a>
```

```
<!-- shorthand -->  
<a :href="url"></a>
```

v-on

```
<!-- full syntax -->  
<a v-on:click="doSomething"></a>
```

```
<!-- shorthand -->  
<a @click="doSomething"></a>
```

Binding HTML Classes

```
<div :class="{  
  'active': isActive,  
  'text-danger': HasError }  
">  
</div>
```

```
data: {  
  isActive: true,  
  hasError: false  
}
```

Conditional Rendering

```
<div v-if="type === 'A'"> A </div>  
<div v-else-if="type === 'B'"> B </div>  
<div v-else-if="type === 'C'"> C </div>  
<div v-else> Not A/B/C </div>
```

Loops

```
<ul id="list">
  <li v-for="(item, index) in items">
    {{ parentMessage }} -
    {{ index }} -
    {{ item.message }}
  </li>
</ul>
```

```
new Vue({
  el: '#list',
  data: {
    parentMessage: 'Parent',
    items: [
      { message: 'Foo' },
      { message: 'Bar' }
    ]
  }
})
```


Event & Key Modifiers

```
<!-- the click event's propagation will be stopped -->  
<a @click.stop="oThis"></a>  
<!-- the submit event will no longer reload the page -->  
<form @submit.prevent="onSubmit"></form>  
<!-- modifiers can be chained -->  
<a @click.stop.prevent="doThat"></a>  
  
<!-- also available .tab, .delete, .esc, .space, ...-->  
<input @keyup.enter="submit">
```

An abstract graphic on a solid blue background. On the left side, there are several vertical, rounded rectangular bars of varying heights and shades of blue. A thin, dark blue horizontal line spans the width of the image, intersecting these bars. To the right of this line, the text "Dev & Tools" is written in a white, sans-serif font.

Dev & Tools

Setup a new Vue Application with CDN

We can just create a basic HTML file and add a link to a Vue CDN in the head, and create a `<div>` with an id of `app`.

index.html

```
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width,initial-scale=1.0" />
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

    <title>Vue App</title>
  </head>

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

    <script>
      const App = new Vue({
        el: '#app',
        data: {
          message: 'Hello Vue!',
        },
      })
    </script>
  </body>
</html>
```

Setup a new Vue Application with Vue CLI

Install the Vue CLI using

install with npm

```
npm i -g @vue/cli @vue/cli-service-global
```

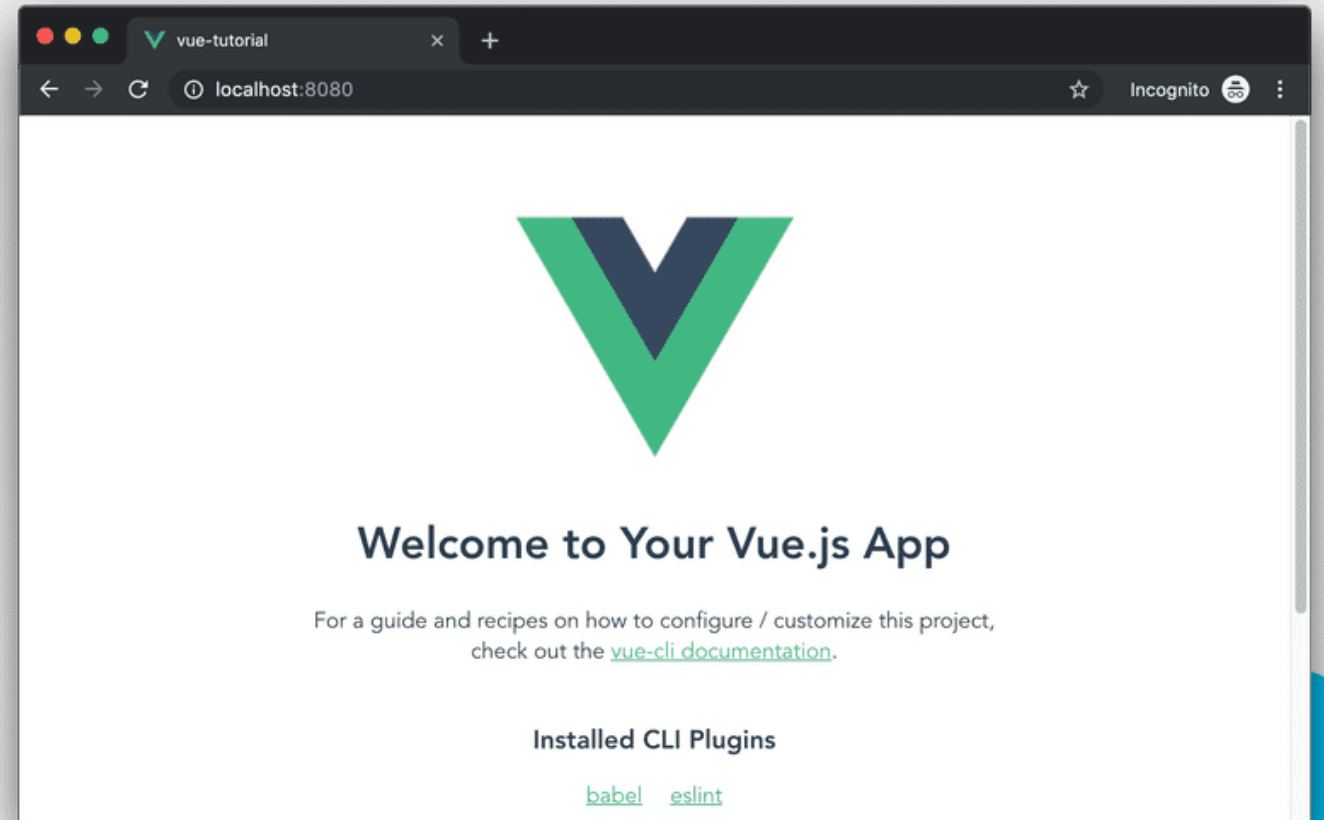
or

install with yarn

```
yarn global add @vue/cli @vue/cli-service-global
```

Run the following:

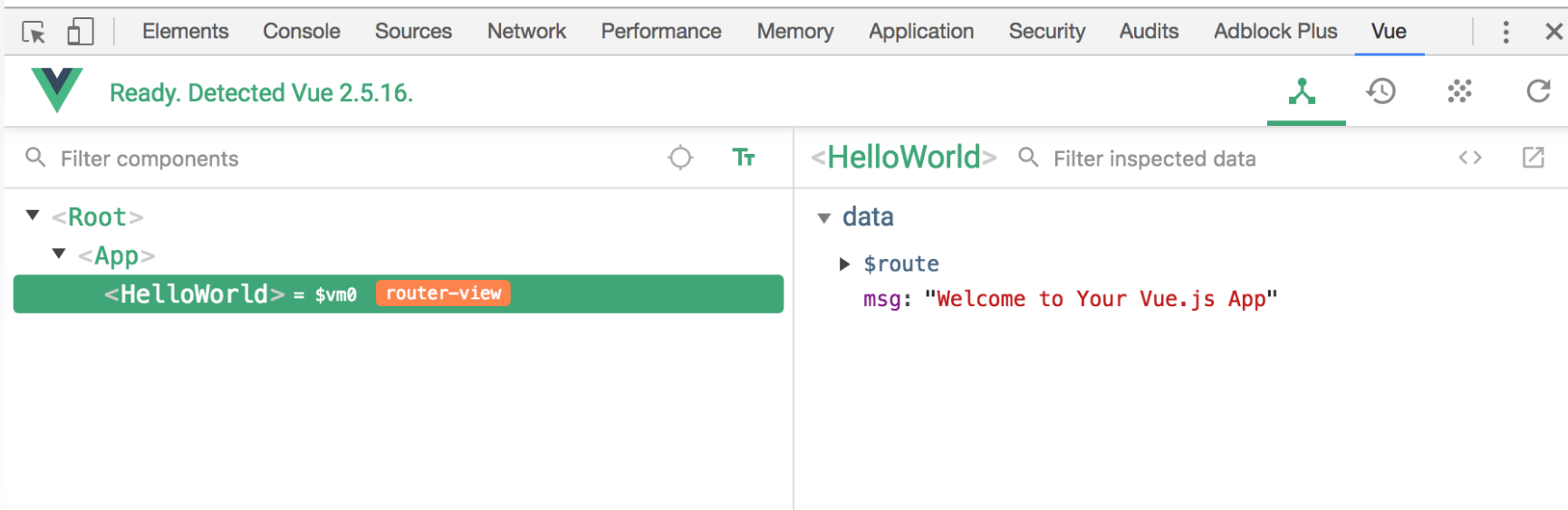
- `vue create vue-app`
- `cd vue-app`
- `npm run serve` or `yarn serve`




Vue DevTools on Chrome
Vue DevTools on FireFox





Welcome to Your Vue.js App





Elements Console Sources Network Performance Memory Application Security Audits Adblock Plus **Vue**

 Ready. Detected Vue 2.5.16.

Filter components  

- <Root>
 - <App>
 - <HelloWorld> = \$vm0 router-view**

<HelloWorld> Filter inspected data  

- data
 - \$route
 - msg: "Welcome to Your Vue.js App"

The background is a solid dark blue. On the left side, there are several vertical, rounded rectangular bars of varying heights and widths, in different shades of blue. A thin, light blue horizontal line crosses the entire width of the image, positioned roughly in the middle vertically.

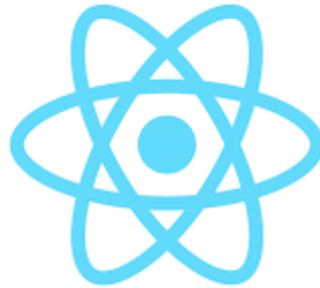
A short comparison

A short comparison



Angular

Developed by Google, was first released in 2010. It is a TypeScript based JavaScript framework. Although AngularJS (version 1) still gets updates, we will move on to Angular. The latest stable version is Angular 8, released in 05 2019.



React

Developed by Facebook, was initially released in 2013. Facebook uses React extensively in their products. The current stable version is 16.X, released in November 2018.



VueJS

Developed by ex-Google employee Evan You in 2014. Vue has seen a substantial shift in popularity. The current stable version is 2.6.x. Vue 3, currently in the prototyping phase, is planning to move to TypeScript.

Here are some questions that can help get a better comparative understanding of these frameworks:

- Are they mature enough to build scalable applications?
- Is it easy to find developers for each of these frameworks?
- Do you know core concepts behind each framework listed here?
- Performance, speed, and the learning curve of each framework?

Let's find answers to each question, one by one.



Here are some questions that can help get a better comparative understanding of these frameworks:

- **Are they mature enough to build scalable applications?**
- **Is it easy to find developers for each of these frameworks?**
- **Do you know core concepts behind each framework listed here?**
- **Performance, speed, and the learning curve of each framework?**

Let's find answers to each question, one by one.



Here are some questions that can help get a better comparative understanding of these frameworks:

- Are they mature enough to build scalable applications?
- **Is it easy to find developers for each of these frameworks?**
- Do you know core concepts behind each framework listed here?
- Performance, speed, and the learning curve of each framework?

Let's find answers to each question, one by one.



Here are some questions that can help get a better comparative understanding of these frameworks:

- Are they mature enough to build scalable applications?
- Is it easy to find developers for each of these frameworks?
- **Do you know core concepts behind each framework listed here?**
- Performance, speed, and the learning curve of each framework?

Let's find answers to each question, one by one.



Here are some questions that can help get a better comparative understanding of these frameworks:

- Are they mature enough to build scalable applications?
- Is it easy to find developers for each of these frameworks?
- Do you know core concepts behind each framework listed here?
- **Performance, speed, and the learning curve of each framework?**

Let's find answers to each question, one by one.





Thank you for
your attention

Follow us

