

Doit! Vue.js

2019.03.19 이경임 금융서비스부 메체서비스팀

2019.03.18

Doit! Vue.js

INDEX

주제	세부토픽
Vue.js 알아보기	<ol style="list-style-type: none">웹의 발전과 VueVue.js 알아보기Angular vs React vs Vue
Vue.js 기술적 특징	<ol style="list-style-type: none">뷰 인스턴스 & 컴포넌트뷰 라우터뷰 HTTP 통신뷰 템플릿
Vue.js로 화면개발하기	<ol style="list-style-type: none">Vue 프로젝트 구성방법Vue.js로 구현한 할일관리앱

2019.03.18

Doit! Vue.js

01

Vue.js 알아보기

Subjects

- 웹의 발전과 Vue
- Vue.js 알아보기
- Angular vs React vs Vue

2019.03.18

Doit! Vue.js

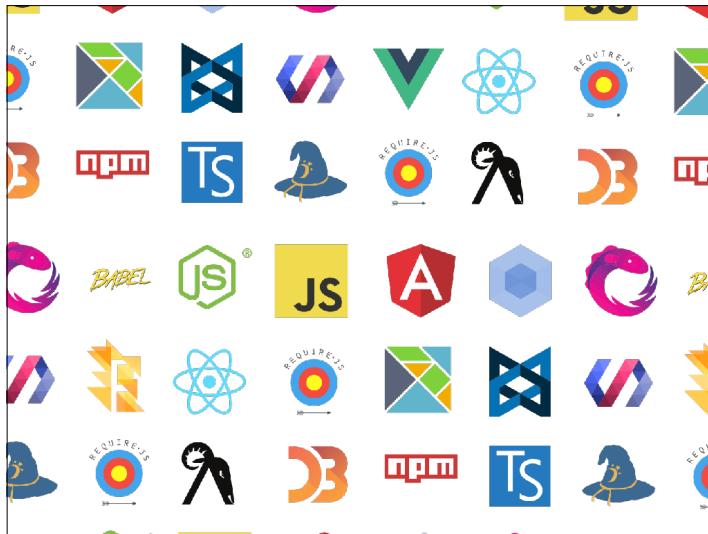
01 웹의 발전과 Vue

2019.03.18

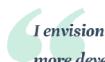
Doit! Vue.js

현재에 프런트엔드 개발은 '라이브러리와 프레임워크를 사용한 개발'을 의미한다. 따라서 어떤 라이브러리와 프레임워크가 프로젝트에 가장 잘 맞을지 고민하고 적용하는 과정은 오늘날 프런트엔드 개발의 주요한 모습이자 중요한 요소.

다양한 라이브러리와 프레임워크가 등장해 저마다의 장점을 내세우지만 현재는 리액트, 뷰, 앵귤러가 주요 라이브러리와 프레임워크로 자리를 잡았다고 할 수 있다.



Vue.js의 비전



I envision Vue's goal to be helping more developers enjoy building apps on the Web.



Evan You
Creator of Vue.js

"Vue.js의 초점은 더 많은 사람들이 쉽게 웹 앱을 개발하도록 도와주는데 있다."

-에반 유(Vue.js 창시자)

빠르게 변하는 프런트엔드 시장.

그중에서도 가장 두드러진 성장을 보여주는 vue.js

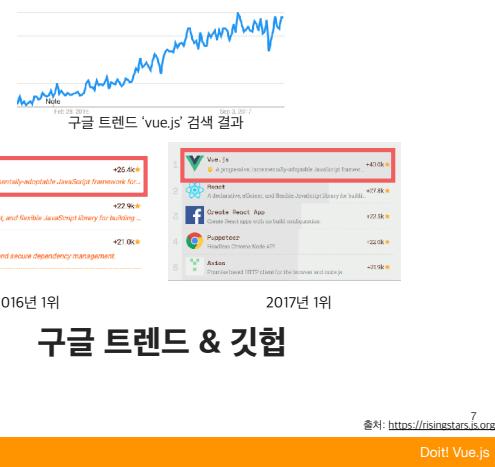
HTML, CSS, javascript만 아는 웹개발자도 쉽게 배우도록 만들어진 웹페이지 화면 개발 위한 프런트엔드 프레임워크.

구글 개발자 에반유, 앵귤러 하다가 방대한 프레임월 이해 부담, 명시적 바인딩 같은 필수요소만으로 화면 구현 시작했고 오픈소스화하면서 사람들이 살을 붙여나감. 그렇게 뷰가 탄생.

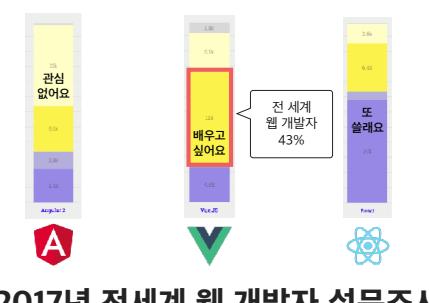
2019.03.18

Doit! Vue.js

통계 Vue에 대한 관심

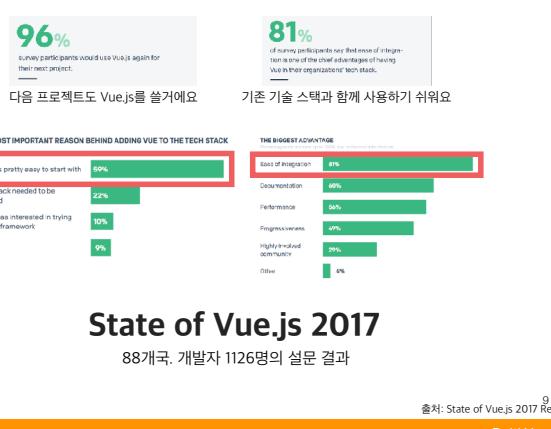


통계 Vue에 대한 관심



2017년 전세계 웹 개발자 설문조사

통계 Vue에 대한 관심



사용성 얼마나 편리한가

jQuery

```
<body>
<div>
  <p></p>
  <button>click me</button>
</div>
```

CDN 라이브러리 로딩

```
<script src="https://code.jquery.com/jquery-3.3.1.js"></script>
```

```
</body>
```

Vue.js

```
<div id="app">
  <{{ message }}>
  <button v-on:click="clickMe">click me</button>
</div>
```

CDN 라이브러리 로딩

```
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
```

```
new Vue({
  el: '#app',
  data: {
    message: 'Hello World'
  },
  methods: {
    clickMe: function() {
      alert('hi');
    }
  }
})
```

jQuery vs Vue.js

file:///Users/gihyojoshua... ☆ ▾ ▾

Hello World

click me

10

2019.03.18

뷰의 또 다른 장점은 리액트와 앵귤러에 비해 우수한 퍼포먼스 보여줌에도 이들보다 배우기가 쉽다는 점. 이에대한 자세한 설명은 뒤에서.

프로젝트 적용 시에 마치 제이쿼리 사용하는 것처럼 CDN으로 불러와서 쉽게 적용 가능. 원한다면 웹팩으로도 구성 할 수 있다.

사용성 얼마나 편리한가



Hello World

Getting Started

The easiest way to get started with React is to use this on CodePen. You don't need to install anything; you can follow along as we go through examples. If you'd like to jump right into a framework as your first step - grasp the basics then come back! Prior experience with other frameworks helps, but is not required.

The smallest React example looks like this:

```
ReactDOM.render(
  <h1>Hello world!</h1>
  document.getElementById('root')
)
```

It renders a header saying "Hello, world!" on the page.

The next few sections will gradually introduce you to using React. We'll be building blocks of React apps: elements and components. Once you understand how to use them, you can start creating complex apps from small, reusable pieces.

Getting Started

V

The official guide assumes intermediate level knowledge of HTML, CSS, and JavaScript. If you are totally new to frontend development, it might not be the best idea to jump right into a framework as your first step - grasp the basics then come back! Prior experience with other frameworks helps, but is not required.

The easiest way to try out Vue.js is using the JSFiddle Hello World example. Feel free to open it in another tab and follow along as we go through some basic examples. Or, you can create an `index.html` file and include Vue with:

```
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
```

The [Installation](#) page provides more options of installing Vue. Note: We do not recommend that beginners start with `vue cli`, especially if you are not yet familiar with Node.js-based build tools.

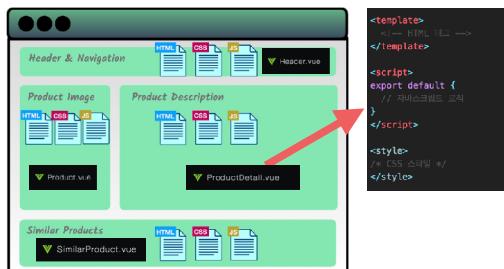
입문자를 배려한 가이드 문서

11

2019.03.18

또 다른 장점은 높은 수준의 도큐먼테이션. 메뉴얼 한글화도 잘 되어있고 정리도 잘 되어있어서 한국 개발자들이 시작하기 좋은 환경.

사용성 얼마나 편리한가



컴포넌트별 코드 관리

12

2019.03.18

Doit! Vue.js

02 Vue.js 알아보기

2019.03.18

Doit! Vue.js

Vue.js란?



작은 화면단 라이브러리 역할부터
큰 규모의 웹 애플리케이션 개발을 돋는 프레임워크 역할까지
점진적으로 적용할 수 있는 프런트엔드 프레임워크

14

예반 유가 2017년 컨퍼런스에서 피티하며 사용한 그림.

뷰 코어 라이브러리는 화면단 데이터 표현에 노간한 기능들을 중점적으로 지원, 하지만 프레임워크의 기능인 라우터, 상태관리, 테스팅 등을 쉽게 결합할 수 있는 형태로도 제공된다.

즉, 라이브러리 역할 + 프레임워크 역할도 할 수 있다는 의미이며 따라서 공식 사이트에서도 뷰를 점진적인 프레임워크라고 부른다.

2019.03.18

Doit! Vue.js

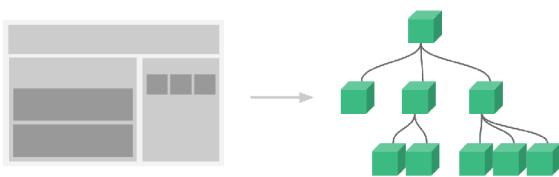
Vue.js 프레임워크 특징 3가지

15

2019.03.18

Doit! Vue.js

1. 컴포넌트 기반 개발 방식



화면을 여러 개의 작은 단위로 쪼개어 개발
재사용성 ↑ 구현속도 ↑ 코드 가독성 ↑

16

2019.03.18

Doit! Vue.js

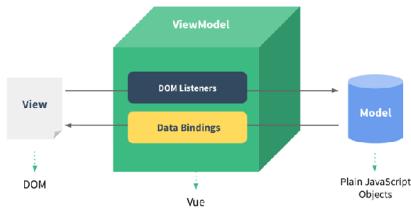
최신 프론트엔드 프레임워크인 리액트, 앵귤러 모두 컴포넌트 기반 개발방식을 추구함.
뷰 역시 컴포넌트 기반 프레임워크의 컴포넌트 조합하여 화면 구성.

(장점)

코드 재사용이 쉬움

프레임워크 자체에서 컴포넌트 방식 추구할 경우 모두가 정해진 방식대로 컴포넌트 사용하므로 빠르게 구현할 수 있음 + 낭 코드 보기 수월해
뷰의 경우 HTML 코드에서 화면 구조 직관적으로 파악 가능

2. MVVM 패턴



화면 UI 코드와 백엔드 데이터 처리 코드를 분리

17

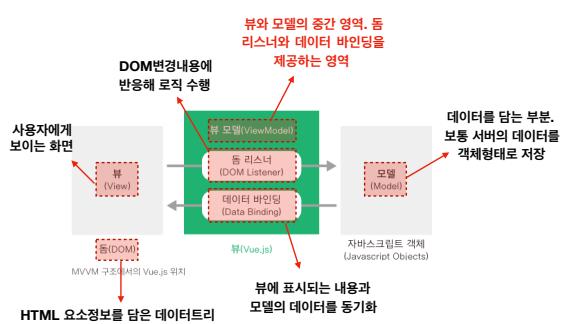
2019.03.18

Doit! Vue.js

뷰는 UI 화면개발 방법 중 하나인 MVVM(Model View ViewModel) 패턴에서 뷰 모델에 해당하는 화면단 라이브러리이다.

MVVM - 화면을 모델 / 뷰 / 뷰모델로 구조화하여 개발하는 방식을 의미. 화면의 요소를 제어하는 코드와 데이터 제어 로직을 분리하여 코드를 더 직관적으로 이해할 수 있고, 추후 유지보수 편하다는 장점!

2. MVVM 패턴



구조도의 용어. 여기서 뷰는 뷰모델에 해당하는 화면단 라이브러리!!!

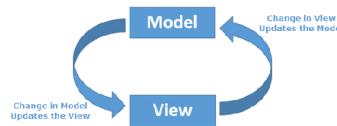
즉, 뷰는 화면의 요소가 변경되거나 조작이 일어날 때 즉각적으로 반응하여 화면의 데이터를 갱신하여 보여주는 역할을 한다. 이처럼 화면의 표현에 주로 관여하므로 화면단 라이브러리라고 함.

18

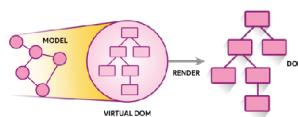
2019.03.18

Doit! Vue.js

3. React와 Angular의 장점을 흡수



Two-way data binding



Virtual DOM

뷰는 앵글러의 양방향 데이터 바인딩과 리액트의 단방향 데이터 흐름의 장점을 모두 결합한 프레임워크. 더해 빠른 화면 렌더링을 위해 리액트의 가상 dom 렌더링 방식을 사용.

- 양방향 데이터 바인딩 : 화면에 표시되는 값과 프레임워크의 모델 데이터 값이 동기화되어 한쪽이 변경되면 다른 한 쪽도 자동으로 변경되는 것을 말함.
- 단방향 데이터 흐름 : 컴포넌트의 단방향 통신을 의미. 컴포넌트간에 데이터를 전달할 때 항상 상위 컴포넌트에서 하위 컴포넌트 한 방향으로만 전달하게끔 프레임워크가 구조화되어있음
- 가상dom 렌더링 방식 : 가상dom을 활용하여 특정 dom 요소를 추가/삭제하는 변경이 일어날 때 화면 전체를 다시 그리지 않고 프레임워크에서 정의한 방식에 따라 화면을 갱신한다. 따라서 브라우저 입장에서는 성능 부하가 줄어들어 일반 렌더링 방식보다 더 빠르게 화면을 그릴 수 있다.

19

2019.03.18

Doit! Vue.js

03 Angular vs Vue vs React

2019.03.18

Doit! Vue.js

프레임워크별 필요 기술스택



- 앵글러의 경우 컴포넌트 기반의 앵글러2로 진화하면서 타입스크립트, ES6 등 가장 많은 학습 필요한 프레임워크가 됨.

- 리액트 역시 학습을 위해서는 ES6, JSX라는 장벽 존재.

타입스크립트

: 기존 자바스크립트에 엄격한 타입 체크를 도입한 언어. 앵글러2의 표준

ES6

: 자바스크립트 최신 스펙.

JSX

: 자바스크립트 문법의 확장. 리액트 컴포넌트들은 일반적으로 JSX를 사용하여 작성됨.

Babel

: 자바스크립트 컴파일러. 최신버전의 자바스크립트 문법은 브라우저가 이해하지 못하기 때문에 비벨이 브라우저가 이해할 수 있는 문법으로 변환해준다.

21

2019.03.18

Doit! Vue.js

Vue vs React



	Vue	React
데이터 바인딩 방식	데이터 바인딩을 프레임워크가 어느정도까지 알아서 해준다.	기본적으로 모든 페이지는 rendering이 되며, 예외의 경우에 대해 개발자가 코드상에서 직접 변경
작용사이트	Alibaba, Baidu, Tencent, Xiaomi, Didi Chuxing, Gitlab 등	instagram, netflix, dropbox, pinterest, twitter, naver facebook 등
Bundle Size	약 20.9KB	약 31.8KB
Performance	속도는 비슷하며, 메모리 소비는 Vue가 React에 비해 적다	

출처: <http://www.stefankrause.net/s-frameworks-benchmark2/table.html> 22

2019.03.18

Doit! Vue.js

02

Vue.js 기술 상세

Subjects

- 뷰 인스턴스 & 컴포넌트
- 뷰 라우터
- 뷰 HTTP통신
- 뷰 템플릿

2019.03.18

Doit! Vue.js

01

뷰 인스턴스 & 컴포넌트

2019.03.18

Doit! Vue.js

뷰 인스턴스

뷰 인스턴스의 형식

```
new Vue({
  ...
});
```

- 뷰로 화면을 개발하기 위한 필수단위.
- 뷰 생성자로 생성한다.
- 미리 정해져 있는 인스턴스 옵션 속성을 설정하여 사용한다.

인스턴스 옵션 정의 및 인스턴스 생성

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

- new Vue로 인스턴스를 생성할 때 Vue를 생성자라고 한다.
- 뷰 생성자는 뷰 라이브러리를 로딩하고 나면 접근할 수 있다.
- 뷰 인스턴스 옵션 속성은 인스턴스 생성 시 재정의할 data, el, template등의 속성을 의미.
- 오른쪽 코드는
 - 1) 생성자로 뷰 인스턴스를 생성
 - 2) el 속성으로 뷰 인스턴스가 그려질 지점을 지정
 - 3) data속성에 메시지 값은 정의하여 화면의 {{message}}에 연결
- el에는 css 선택자 규칙에 맞춰 들어감.

속성

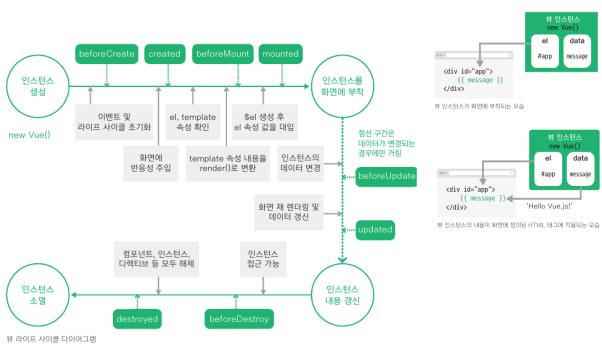
속성	설명
template	화면에 표시할 HTML등의 마크업 요소를 정의하는 속성
methods	이벤트 처리와 같이 화면 로직 제어와 관계된 메서드를 정의하는 속성
created	뷰 인스턴스가 생성되자마자 실행할 로직을 정의할 수 있는 속성

25

2019.03.18

DoIt! Vue.js

뷰 인스턴스 인스턴스 라이프 사이클



26

인스턴스가 생성된 후 화면에 부착/ 소멸되기까지의 과정. 인스턴스가 위의 라이프 사이클 중 어떤 단계/ 어떤 상태에 있느냐에 따라 호출할 수 있는 속성이 달라진다.

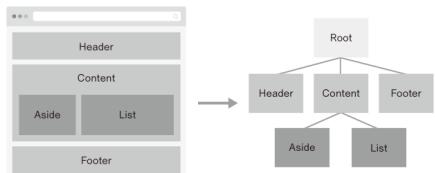
크게 인스턴스의 '생성' '부착' '갱신' '소멸'의 4가지 단계로 구분된다.

- **created :** data 속성과 메서드 속성에 접근할 수 있는 가장 첫 라이프사이클 단계. 서버에 데이터 요청해 받아오기 좋다.
아직 화면에 인스턴스가 부착되기 전이므로 템플릿 속성에 정의된 둘 요소로 접근 불가능.
- **mounted :** el속성에서 지정한 화면위치에 인스턴스 부착되고 나면 호출되는 단계.
화면요소 DOM에 접근할 수 있으므로 화면을 제어하는 로직 수행하기 좋음.
- **beforeUpdate :** 화면에 인스턴스 부착되면 인스턴스에 정의한 속성들이 화면에 치환되면서 뷰 반응성 제공위해 \$watch 속성으로 감시된다(데이터 관찰).
이때 관찰하고 있는 데이터 변경되면 가상돔으로 화면을 다시 그리기 전에 beforeUpdate 호출됨. 변경 예정인 데이터에 접근할 수 있어서 이와 관련된 로직 미리 넣을수있다.
- **updated :** 데이터 변경되고 나서 가상돔으로 다시 화면을 그리고나면 실행되는 단계. 변경 후 화면제어요소와 관련된 로직 추가하기 좋음.

2019.03.18

DoIt! Vue.js

뷰 컴포넌트



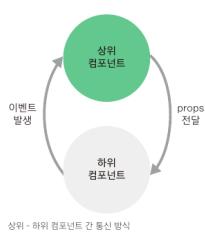
27

2019.03.18

DoIt! Vue.js

컴포넌트 통신

- 부는 한 화면을 여러 컴포넌트로 구성한다. 따라서 같은 웹페이지라도 서로간에 직접 데이터를 공유할 수 없다.
- 이는 컴포넌트마다 독립적인 유효범위(scope)를 갖기 때문.



- 부의 기본적인 데이터 전달방법
: 상위(부모) -> 하위(자식)
- 상위 컴포넌트에 등록된 하위 컴포넌트로 **props**라는 특수한 속성 전달하여 값 전달.
- 하위 컴포넌트에서는 **이벤트**를 발생시켜 상위 컴포넌트에 신호를 보낸다.
 - \$emit : 이벤트 발생
 - v-on : 이벤트 수신
- 같은 레벨 간 컴포넌트 통신에는 이벤트버스 활용

28

2019.03.18

Doit! Vue.js

이벤트를 발생시키고 수신하기

event2 이벤트는 <child>에 정의한 v-on:event2에 전달되고, v-on: event2의 대상 메서드인 최상위 컴포넌트의 메서드 event3 가 실행된다.

event1() 메서드 인의 this.\$emit('event2')가 실행되면서 event2 이벤트가 발생

버튼 클릭시 클릭 이벤트 v-on: click="event1"에 따라 event1() 메서드 실행.

event3()는 컴포넌트 data 속성의 message값을 바꾼다. 데이터가 바뀌면 화면도 반응하여 바뀌게 된다.

```
<body>
  <div id="app">
    {{message}}
    <child v-on:even2="event3"></child>
  </div>
<script src="https://cdn.jsdelivr.net/npm/vue@2.5.2/dist/vue.js"></script>
<script>
  Vue.component('child',{
    template: <button v-on:click="event1">i am a child component</button>,
    methods: {
      event1: function(){
        this.$emit("event2")
      }
    }
  });

  var app = new Vue({
    el: '#app',
    data: {
      message: "I am a parent component"
    },
    methods: {
      event3: function(){
        this.message = "Received event!"
      }
    }
  });
</script>
</body>
```

29

2019.03.18

Doit! Vue.js

* 인스턴스 생성 전에 컴포넌트를 먼저 등록해야함

02 뷰 라우터

2019.03.18

Doit! Vue.js

- props 사용

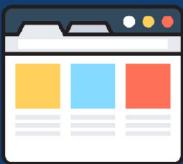
- 1) 상위 컴포넌트에 하위 컴포넌트 등록.
- 2) 하위컴포넌트에 props 속성 정의
- 3) 하위컴포넌트 HTML 태그에 v-bind 속성 추가

- 이벤트 사용

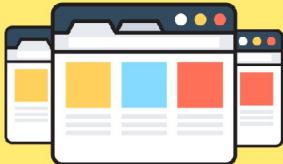
- 1) 하위 컴포넌트에서 이벤

라우팅이란?

Multiple Page Applications



VS



Single Page Applications



31

2019.03.18

Doit! Vue.js

- 웹에서의 하우팅이란 웹 페이지 간의 이동을 말한다. 상용 웹 앱의 경우 여러 페이지로 구성되어 있기 때문에 페이지 간에 이동을 돋는 라우터를 사용해야 한다.

- 라우팅은 현대 웹 앱의 형태 중 하나인 싱글 페이지 애플리케이션에서 주로 사용하고 있다.

- 싱글 페이지 애플리케이션

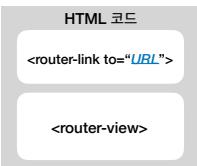
: 페이지를 이동할 때마다 서버에 웹 페이지를 요청하여 새로 갱신하는 것이 아니라 미리 해당 페이지들을 받아 놓고 페이지 이동 시에 클라이언트의 라우팅을 이용하여 화면을 갱신하는 패턴을 적용한 애플리케이션.

뷰 라우터

- 뷰 라우터는 뷰에서 라우팅 기능을 구현할 수 있도록 지원하는 공식 라이브러리이다.
- 뷰 라우터를 구현하기 위해 필요한 특수 태그와 기능은 아래와 같다.

태그	설명
<router-link to="URL값">	페이지 이동 태그. 화면에서는 <a>로 표시되며, 클릭하면 to에 지정한 URL로 이동합니다.
<router-view>	페이지 표시 태그. 변경되는 URL에 따라 해당 컴포넌트를 뿌려주는 역할입니다.

뷰 라우터 사용



32

- 뷰 라우터 정의 >> new VueRouter로 생성

- 뷰 인스턴스에 라우터 추가할 때 \$mount('#app') 과 같이 사용. el 속성과 동일하게 인스턴스를 화면 특정 위치에 붙이는 역할 한다. el 넣지 않고 인스턴스 생성했더라도 mount() 활용해서 강제로 붙일 수 있음

2019.03.18

Doit! Vue.js

네스티드 라우터 & 네임드 뷰

- 실제 웹 앱은 화면이 여러개로 분할된 경우가 많다.
- 네스티드 라우터와 네임드 뷰는 여러 개의 컴포넌트를 동시에 표시할 수 있는 라우터이다.

네스티드 라우터의 구조



- 상위 컴포넌트 1개에 하위 컴포넌트 1개를 포함하는 구조

- URL에 따라서 컴포넌트의 하위 컴포넌트가 다르게 표시된다.

- 링크되는 기본 컴포넌트(user) 내에도 하위 컴포넌트(post, profile) 내용을 표시할 <router-view>가 있다.

- 한번에 많은 수의 컴포넌트를 표시하는 데에는 한계가 있다.

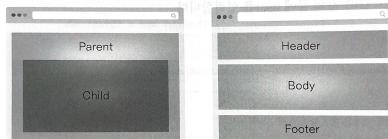
33

2019.03.18

Doit! Vue.js

네스티드 라우터 & 네임드 뷰

- 실제 웹 앱은 화면이 여러개로 분할된 경우가 많다.
- 네스티드 라우터와 네임드 뷰는 여러 개의 컴포넌트를 동시에 표시할 수 있는 라우터이다.



- 네임드 뷰는 특정 페이지로 이동했을 때 **여러개의 컴포넌트를 동시에 표시하는** 라우팅 방식이다.
- 네스티드 라우터는 상위 컴포넌트가 하위 컴포넌트를 포함하는 형식이라면 **네임드 뷰는 같은 레벨에서 여러개의 컴포넌트를 한번에 표시한다.**
- 이를과 같이 <router-view>에 name이 지정된 형태이다.

34

2019.03.18

Doit! Vue.js

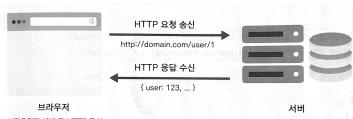
03 뷰 HTTP 통신

2019.03.18

Doit! Vue.js

웹 앱의 HTTP 통신

HTTP는 브라우저와 서버 간에 데이터를 주고받는 통신 프로토콜이다.



- 사용자와 상호작용에 따라 동적으로 데이터를 화면에 표시해야 하는 현재, 웹 앱의 HTTP 통신구현은 필수다.
- 웹 앱 HTTP 통신의 대표적 사례는 제이쿼리의 ajax



- 서버에서 받아온 데이터를 표시할 때 화면 전체를 갱신하지 않고도 화면의 일부분만 변경할 수 있도록 하는 자바스크립트 기법
- React, Angular 등에서도 Ajax를 사용하고 있다.
- 뷰에서도 마찬가지로 Ajax를 위한 라이브러리 제공.

36

2019.03.18

Doit! Vue.js

Axios(액시오스)

- 현재 뷰 커뮤니티에서 가장 많이 사용되는 HTTP 통신 라이브러리
- 별도로 구현 필요 없이 제공되는 API 만으로도 간편하게 원하는 기능을 구현할 수 있다.

API 유형	처리 결과
axios.get('URL').then().catch()	<ul style="list-style-type: none">• 해당 URL에 대해 HTTP GET 요청을 보낸다.• 데이터를 정상적으로 받아오면 then()의 로직이, 오류가 발생하면 catch()의 로직이 실행된다.
axios.post('URL').then().catch()	<ul style="list-style-type: none">• 해당 URL에 대해 HTTP POST 요청을 보낸다.• then()과 catch() 로직의 동작은 동일
axios({options})	<ul style="list-style-type: none">• HTTP 요청에 대한 자세한 속성을 직접 정의하여 보낸다.• (ex) 요청을 보낼 URL, 요청방식, 보내는 데이터 유형..

- 액시오스의 응답 데이터는 일반 문자열이 아닌 객체 형태이기 때문에 별도로 JSON.parse()를 사용하여 변환할 필요가 없다는 점에서 편리하다.
- 이외에도 초기에 뷰 코어 팀에서 공식적으로 권하는 HTTP 통신 라이브러리였던 뷰 리소스를 사용할 수 있다.(2016년 예반의 공식적 지원 중단됨.)

37

2019.03.18

Doit! Vue.js

04

뷰 템플릿

2019.03.18

Doit! Vue.js

뷰 템플릿이란?

- HTML, CSS 등의 마크업 속성과 뷰 인스턴스에서 정의한 데이터, 로직들을 연결하여 사용자가 브라우저에서 볼 수 있는 형태의 HTML로 변환해주는 뷰 인스턴스 속성이다.

```
<script>
  new Vue({
    template: '<p>Hello {{msg}}</p>'
  });
</script>
```

라이브러리 내부적으로 **template** 속성에서 정의한 데이터

를 가상 둘기반의 **render()** 함수로 변환한다. 변환과정에

서 뷰의 반응성이 화면에 더해지며, 변환된 render() 함수는
화면을 그리게 된다.

39

2019.03.18

Doit! Vue.js

액시오스와 뷰 리소스 모두 CDN을 이용하여 설치하는 방법과 NPM을 이용하여 설치하는 방법 모두 가능하다.

- NPM : Node Package Manager >> Node.js에서 사용 가능한 모듈들을 패키지화하여 모아놓은 것.
- CDN : Content Delivery Network >> 콘텐츠 전송 네트워크. JS라이브러리와 같은 정적 파일을 제공하는 데에 최적화되어 있으며 파일을 매우 빠르게 전달. 웹 페이지에서는 CDN을 이용해 JS라이브러리를 가져올 수 있다.

템플릿에서 사용하는 주요 속성과 문법

01 데이터바인딩

`{} 콧수염괄호` (이중 중괄호)
뷰 인스턴스를 HTML 태그에 연결하는 가장 기본적인 텍스트 삽입방식
v-bind
HTML 속성값에 뷰 데이터값을 연결할때 사용하는 데이터 연결방식

이 외에도 고급 템플릿 기법 `computed`와 `watch`가 있다.
`computed` 속성의 로직은 데이터 연산을 정의하는 영역으로 값이 변경되면 자동으로 호출된다(`methods` 속성의 함수는 호출할때만 수행됨)
`watch` 속성은 데이터 변화를 감지하여 자동으로 특정 로직을 수행한다.
`computed` 속성과 유사하지만 `computed`에 비해 데이터 호출과 같이 시간이 상대적으로 더 많이 소모되는 비동기 처리에 적합하다.

02 디렉티브

뷰 디렉티브란 **HTML 태그 안에 v- 접두사를 갖는 모든 속성을 의미한다.**
뷰의 데이터 값 변경시 화면 요소들이 반응하여 갱신된다. 이처럼 화면 요소를 직접 제어할 필요 없이 뷰의 디렉티브 활용하여 조작 가능.
(ex) `v-if`, `v-for`, `v-show`, `v-bind`, `v-on`, `v-model` 등

03 이벤트처리

`v-on` 디렉티브와 `methods` 속성을 활용한 이벤트 처리

40

2019.03.18

Doit! Vue.js

03

Vue.js로 화면개발하기

Subjects

1. Vue 프로젝트 구성방법
2. Vue.js로 구현한 할일관리 앱

2019.03.18

Doit! Vue.js

01

Vue 프로젝트 구성방법

2019.03.18

Doit! Vue.js

싱글 파일 컴포넌트 체계

- .vue 파일로 프로젝트 구조를 구성하는 방식. .vue 파일 1개는 컴포넌트 1개와 동일하다.

```
<template>
  <!– HTML태그내용 –>
</template>

<script>
export default {
  // 자바스크립트 내용
}
</script>

<template>
  /* CSS 스타일 내용 */
</template>
```

화면에 표시할 요소들을 정의하는 영역.
예) HTML + 뷰 데이터 바인딩

뷰 컴포넌트의 내용을 정의하는 영역
예) template, data, methods 등

템플릿에 추가한 HTML태그의 CSS
스타일을 정의하는 영역

43

2019.03.18

Doit! Vue.js

개발환경 설정

1. [Node.js 설치](#) 

2. [Vue CLI 설치](#) 

3. [Vue 개발자 도구 설치](#) 

4. [VisualStudio Code 설치](#) 

5. [Git 설치](#) 

44

2019.03.18

Doit! Vue.js

뷰 CLI 도구

- vue 개발자들이 편하게 프로젝트를 구성할 수 있도록 뷰 코어 팀에서 제공하는 도구.
- 커맨드 창에서 명령어로 뷰 개발을 위한 프로젝트 구성과 관련된 여러 작업을 수행할 수 있다.

```
$ vue init webpack my-project
```

뷰 CLI 명령어 사용한
프로젝트 생성



npm install 명령어로 다운받은 라이브러리가
존재하는 위치

.vue파일을 비롯한 앱 동작 로직이 들어가는 위치

뷰 만든 웹앱의 시작점. npm run dev 실행시 로딩되는 파일

npm설정파일. 뷰 애플리케이션이 동작하는 데 필요한
라이브러리들을 정의하는 파일

45

2019.03.18

Doit! Vue.js

이러한 싱글파일 컴포넌트 체계 사용 위해서는 .vue파일을 웹 브라우저가 인식할 수 있는 형태의 파일로 변환해주는 웹팩이나 브라우저리 파이와 같은 도구가 필요하다.

** 웹팩 **

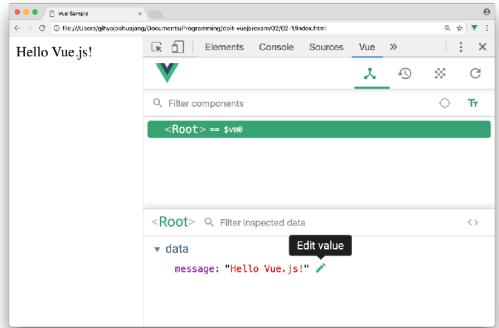
웹 앱의 자원들을 자바스크립트 모듈로 변환해 하나로 묶어 웹 성능을 향상시켜주는 자바스크립트 모듈 번들러이다.

각각의 도구 사용법 다시 공부하는 부담 덜어주기 위해 뷰 개발자들이 편하게 프로젝트 구성할 수 있도록 뷰 코어 팀에서 CLI도구를 제공한다.

뷰 프로젝트 생성 후 CLI 안내문에 따라 npm install을 입력하여 관련 라이브러리 모두 다운하면 그림과 같은 폴더 구조 생성됨.

뷰 개발자 도구

- 뷰 개발에 도움을 주는 뷰 크롬 플러그인. 뷔로 만든 웹 앱의 구조를 디버깅, 분석할 수 있다.
- 크롬, 사파리, 파이어폭스 등에서 모두 지원된다.



46

2019.03.18

Doit! Vue.js

02

Vue 로 구현한 할일관리 앱

2019.03.18

Doit! Vue.js

애플리케이션 설계



48

2019.03.18

Doit! Vue.js

서비스에 필요한 데이터 조작 및 컴포넌트 구조화/통신을 구현해 볼 수 있는 어플리케이션. 가장 대표적인 뷔 관련 예제 프로젝트.

하나의 페이지지만 앱 이름 보여주는 헤더, 할일 넣는 인풋, 할일 목록 리스트, 전체 삭제하는 footer 의 4개 컴포넌트로 이루어져있다. 이렇게 화면 1개를 여러개의 컴포넌트로 쪼개 놓음으로서 반응성이 더 좋은 뷔 애플리케이션 설계할 수 있을 뿐만 아니라 다른 페이지에서 해당 컴포넌트 재사용 용이.

컴포넌트 기반 프레임워크에서는 컴포넌트 단위를 작게 설계하도록 권한다.

컴포넌트는 관례상 src/components 폴더에서 관리한다. 애플리케이션의 규모가 커져서 기능별로 관리하는 경우에는 components/기능/컴포넌트.vue 같은 식으로 관리함.

App.vue는 최상위 컴포넌트

* 컴포넌트의 data 속성은 꼭 함수로 작성해야 한다.

애플리케이션 코드 예시 할일 추가

HTML 태그 내용

```
<template>
<div class="inputBox shadow">
  <input type="text" v-model="newTodoItem" placeholder="Type what you have to do" v-on:keypress.enter="addTodo">
  </div>
</template>
```

```
<script>
export default {
  data() {
    return {
      newTodoItem: '',
      showModal: false
    }
  },
  methods: {
    addTodo() {
      if (this.newTodoItem === '') {
        var value = this.newTodoItem || this.newTodoItem.trim();
        this.$emit('addTodo', value)
        this.clearInput();
      }
    },
    clearInput() {
      this.newTodoItem = '';
    }
  }
}</script>
```

데이터의 변경(할일 추가/삭제)이 화면에 바로 반영되지 않는 구조!

```
<template>
<!-- HTML태그내용 -->
</template>

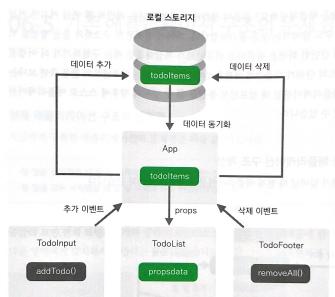
<script>
export default {
  // 자바스크립트 내용
}
</script>
```

49

2019.03.18

Doit! Vue.js

애플리케이션 구조



- 로컬 스토리지의 데이터 조회, 추가, 삭제 등의 작업을 모두 App 컴포넌트에서 한다.
- 하위컴포넌트들은 데이터 표현 혹은 데이터 조작에 대한 요청(이벤트 발생)만 담당

50

2019.03.18

Doit! Vue.js

애플리케이션 코드 예시 할일 지우기

```
<template>
<section>
  <ul v-for="(todoItem, index) in propsdata" :key="todoItem" class="shadow">
    <li>
      <span class="removeBtn" type="button" @click="removeTodo(todoItem, index)"></span>
    </li>
  </section>
</template>

<script>
export default {
  props: ['propsdata'],
  methods: {
    removeAll() {
      localStorage.clear();
      this.todoItems = [];
    },
    addTodo(todoItem) {
      localStorage.setItem(todoItem, todoItem);
      this.todoItems.push(todoItem);
    },
    removeTodo(todoItem, index) {
      localStorage.removeItem(todoItem);
      this.todoItems.splice(index, 1);
    },
    created() {
      if (localStorage.length > 0) {
        for (var i = 0; i < localStorage.length; i++) {
          this.todoItems.push(localStorage.key(i));
        }
      }
    }
  }
}</script>
```

TodoList.vue 코드

```
<script>
export default {
  props: ['propsdata'],
  methods: {
    removeTodo(todoItem, index) {
      this.$emit('removeTodo', todoItem, index);
    }
  }
}</script>
```

TodoList.vue 코드에서 App으로 이벤트 전달

51

2019.03.18

Doit! Vue.js

애플리케이션 코드 예시 할일 목록 표시

The screenshot displays two code snippets. On the left is `TodoList.vue`, which contains a template with a section for each todo item and a script block defining methods like `removeTodo`. On the right is `App.vue`, which includes a `TodoList` component and a `ClearAll` button.

```
<template>
<section>
  <ul v-for="todoItem, index" in propsdata>
    <li v-bind:key="todoItem">
      {{ todoItem }}
      <span class="removeBtn" type="button" @click="removeTodo(todoItem, index)"></span>
    </li>
  </ul>
</section>
</template>
```

```
export default {
  data() {
    return {
      todoItems: []
    };
  },
  methods: {
    clearAll() {
      localStorage.clear();
      this.todoItems = [];
    },
    addTodo(todoItem) {
      localStorage.setItem(todoItem, todoItem);
      this.todoItems.push(todoItem);
    },
    removeTodo(todoItem, index) {
      localStorage.removeItem(todoItem);
      this.todoItems.splice(index, 1);
    }
  }
}
```

```
<script>
export default {
  props: ['propsdata'],
  methods: {
    removeTodo(todoItem, index) {
      this.$emit('removeTodo', todoItem, index);
    }
  }
}</script>
```

`TodoList.vue 코드`

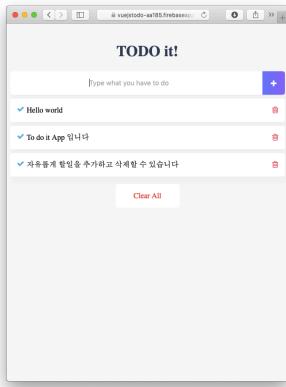
`App.vue 코드`

52

2019.03.18

Doit! Vue.js

최종 완성 앱



제작에 사용한 참고 문헌

53

2019.03.18

Doit! Vue.js

References

자료 제작에 참고한 문헌 및 사이트

- Doit! Vue.js, 장기효 저
- <https://vuejs.org/v2/guide/> : 뷰 공식 가이드
- <https://github.com/joshua1988/doit-vuejs/> : 관련 예제 코드
- <https://vuejstodo-aa185.firebaseioapp.com/> : Todolt 앱 확인 링크
- <https://codesandbox.io/>

제작에 사용한 참고 문헌

54

2019.03.18

Doit! Vue.js

감사합니다.

55

2019.03.18

Doit! Vue.js