



Codingapple

사용하는 이유

1. 데이터바인딩/ 자료 iteration/ DOM 조작이 매우쉬워짐
2. 요소를 실시간으로 반응시켜야할 때
3. Component 기반 개발방식을 적용할 때
4. SPA를 만들 때



The Progressive
JavaScript Framework



WHY VUE.JS?

GET STARTED



GITHUB

설치는 그냥 흔한 JS 라이브러리 설치하듯 하면 됩니다.

(복붙) `<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>`

이제 공식 튜토리얼을 해설과 함께 따라해보면 됩니다.

1. 데이터바인딩 / iteration

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

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

가장 간단한 Text 바인딩

- 새로운 Vue instance 생성
- el : 바인딩 원하는 element id 지정
- data : 본격적으로 데이터와 함수가 담기는 공간

2. 데이터바인딩 / iteration (2)

```
<div id="app">  
  <span v-bind:class="style"></span>  
</div>
```

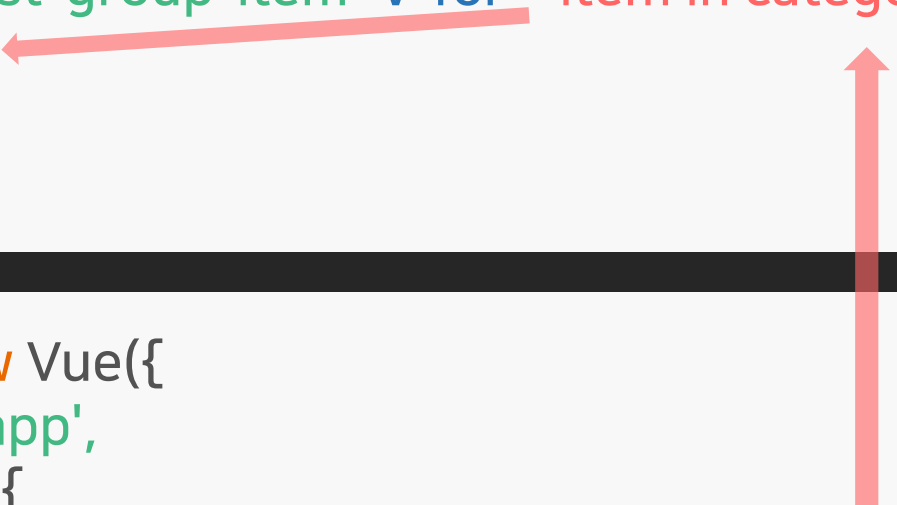
```
var app = new Vue({  
  el: '#app',  
  data: {  
    style : 'red-button'  
  }  
});
```

가장 간단한 attribute 바인딩

```
<span class="red-button">
```

3. 데이터바인딩 / iteration (3)

```
<ul class="list-group">
  <li class="list-group-item" v-for="item in category">
    {{item}}
  </li>
</ul>
```



```
var app = new Vue({
  el: '#app',
  data: {
    category: ['Dress', 'Outer', 'Skirt', 'Innerwear']
  }
});
```

데이터 iteration (for 문안쓰고
도 이렇게 쉽게 for문 기능을!)

category라는 자료의
value(array)를 HTML에 쉽게
iterate할 수 있음.

4. Two way data binding과 실시간 반응성

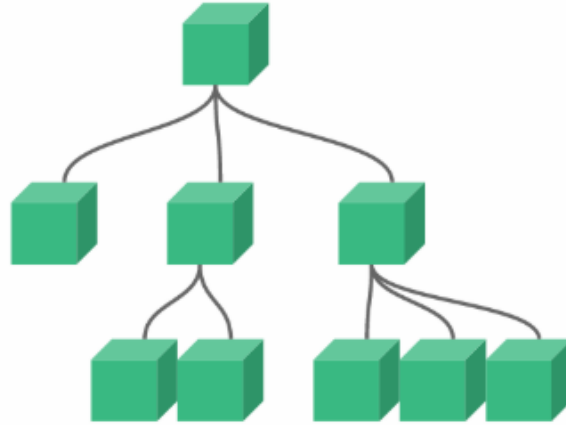
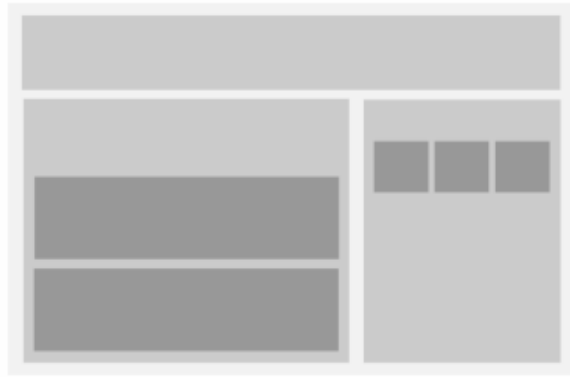
```
<div id="app">  
  <p>{{ message }}</p>  
  <input v-model="message">  
</div>
```

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

v-model 이라는 directive를 이용해서 사용자의 input으로 JS데이터를 실시간으로 변경시킬 수 있음 (= Two way binding)

(One way binding은 흔히말하는 JS -> HTML 순의 데이터바인딩)

5. Component 기반 개발방식



초록색 네모가
전부 component

웹사이트를 저렇게 Component화 해놓은 뒤,
페이지마다 component 들을 첨부하는 방식으로 웹개발진행.

5. Component 기반 개발방식

<배경/>

<카드/>

<버튼 1/>

<버튼 2/>

<카드/>

<버튼 1/>

<카드/>

5. Component 기반 개발방식

```
<배경>  
  <카드/>  
</배경>
```

Component화 해놓고 원하는 페이지에서 조립하여 웹을 개발합니다.

장점1. 유지보수가 쉬움

장점2. 페이지수가 많은 큰 프로젝트일 때 좋음

장점3. HTML 레이아웃의 쉬운 재사용

6. Component 기반 개발방식 가장 쉬운 예

```
<div id="app3">  
  <blog-post></blog-post>  
</div>
```

```
Vue.component('blog-post', {  
  template: '<h3>How to make dinner</h3>'  
});  
  
new Vue({  
  el: '#app3',  
  data: {}  
});
```

props라는 부분은
attribute 데이터를 가져와서
template에 박아넣음.

7. 세부내용이 달라지는 Component

```
<div id="app3">  
  <blog-post title="My journey with Vue"></blog-post>  
  <blog-post title="Bloggng with Vue"></blog-post>  
  <blog-post title="Why Vue is so fun"></blog-post>  
</div>
```

```
Vue.component('blog-post', {  
  props: ['title'],  
  template: '<h3>{{ title }}</h3>'  
});
```



```
new Vue({  
  el: '#app3',  
  data: {}  
});
```

props라는 부분은
attribute 데이터를 가져와서
template 내에 바인딩 될 수 있게
도와줌.