

Vue.js

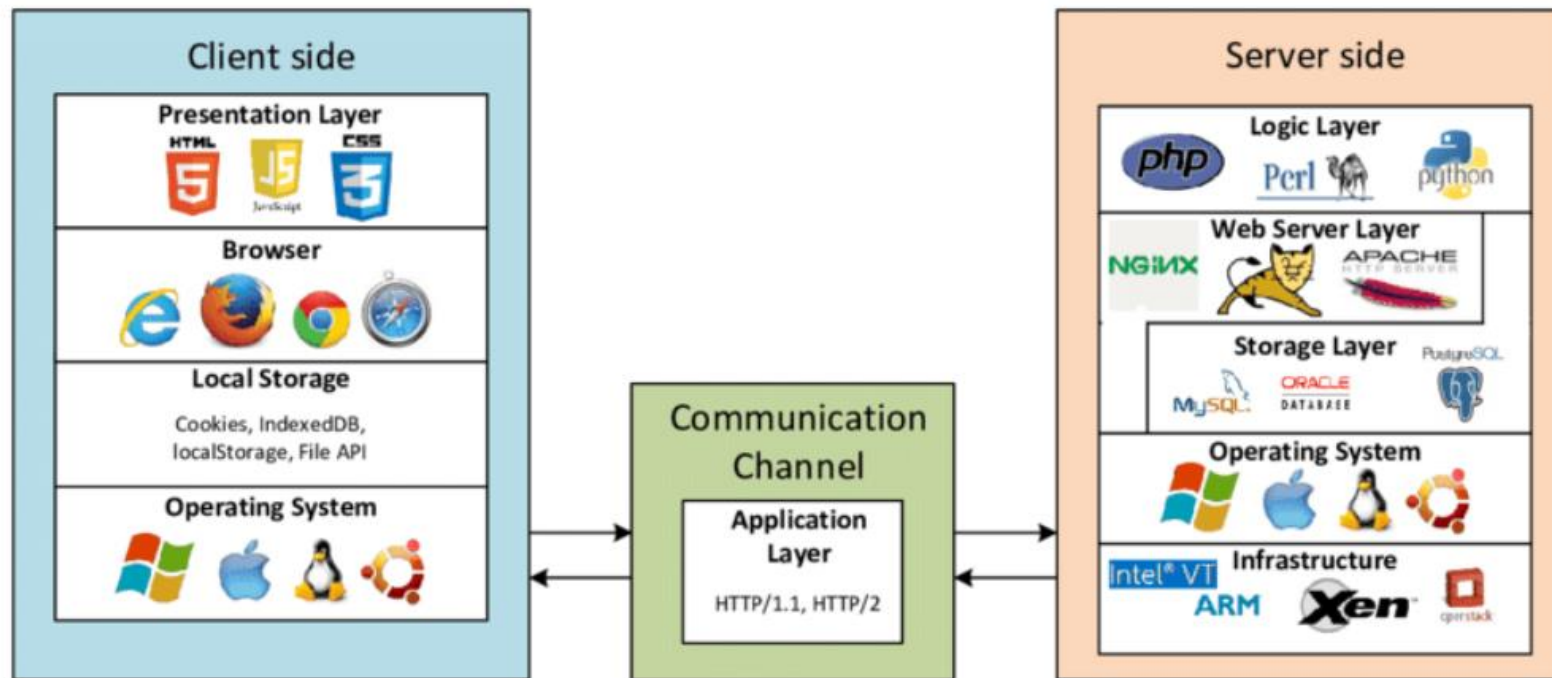
Kim Hye Kyung

topickim@naver.com

| Web Application Architecture

Web Application Architecture

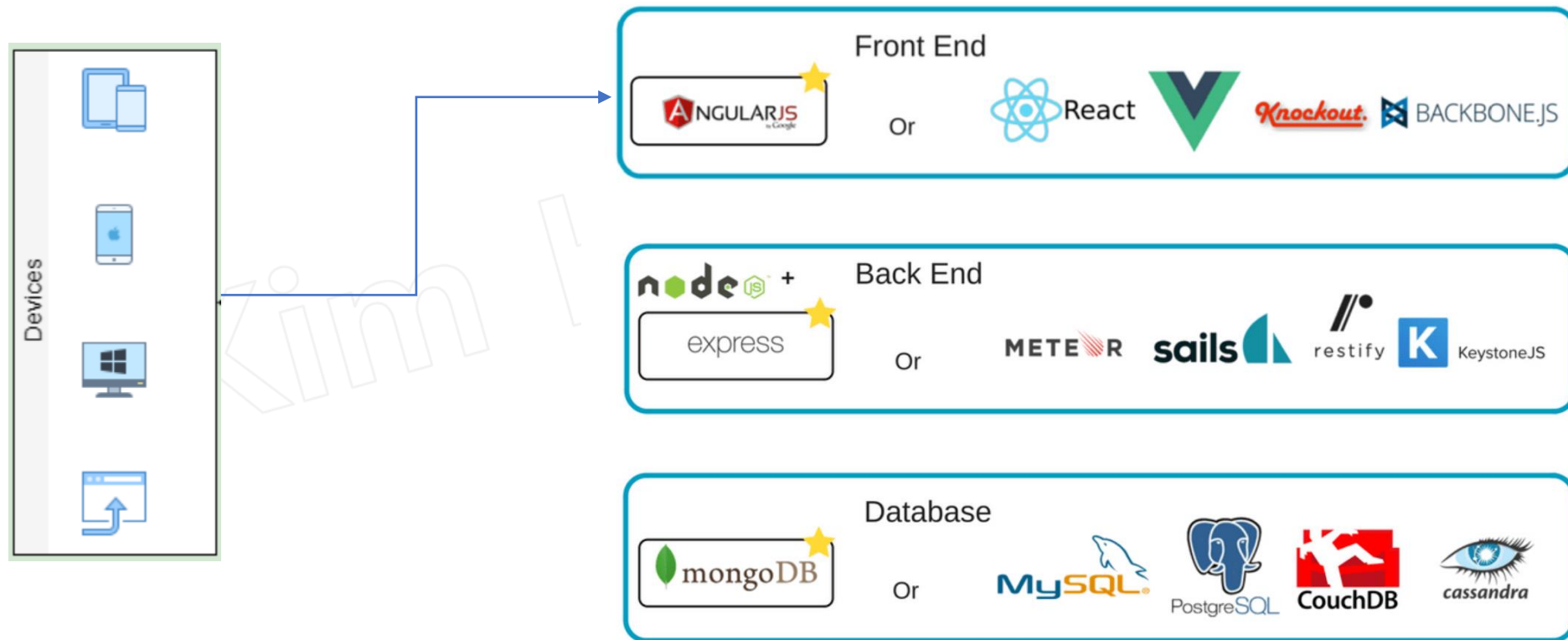
- A Modern Web Application Architecture and Its Running Environments.



https://www.researchgate.net/figure/A-Modern-Web-Application-Architecture-and-Its-Running-Environments_fig1_280714821

Web Application Architecture

Full Stack JavaScript Tools and Technologies



| Vue.js 소개

Vue.js란?



웹 페이지 화면을 개발하기 위한 프론트엔드 프레임워크

- 뷰 레이어에 특화된 라이브러리
- 뷰 만을 다루는 단순한 라이브러리

2014년 2월에 공식 배포

구글 직원이 앵귤러로 개발 하다 앵귤러의 명시적인 데이터 바인딩 기술을 적용하여 더 간소화된 framework로 진화해서 개발

용어 정리

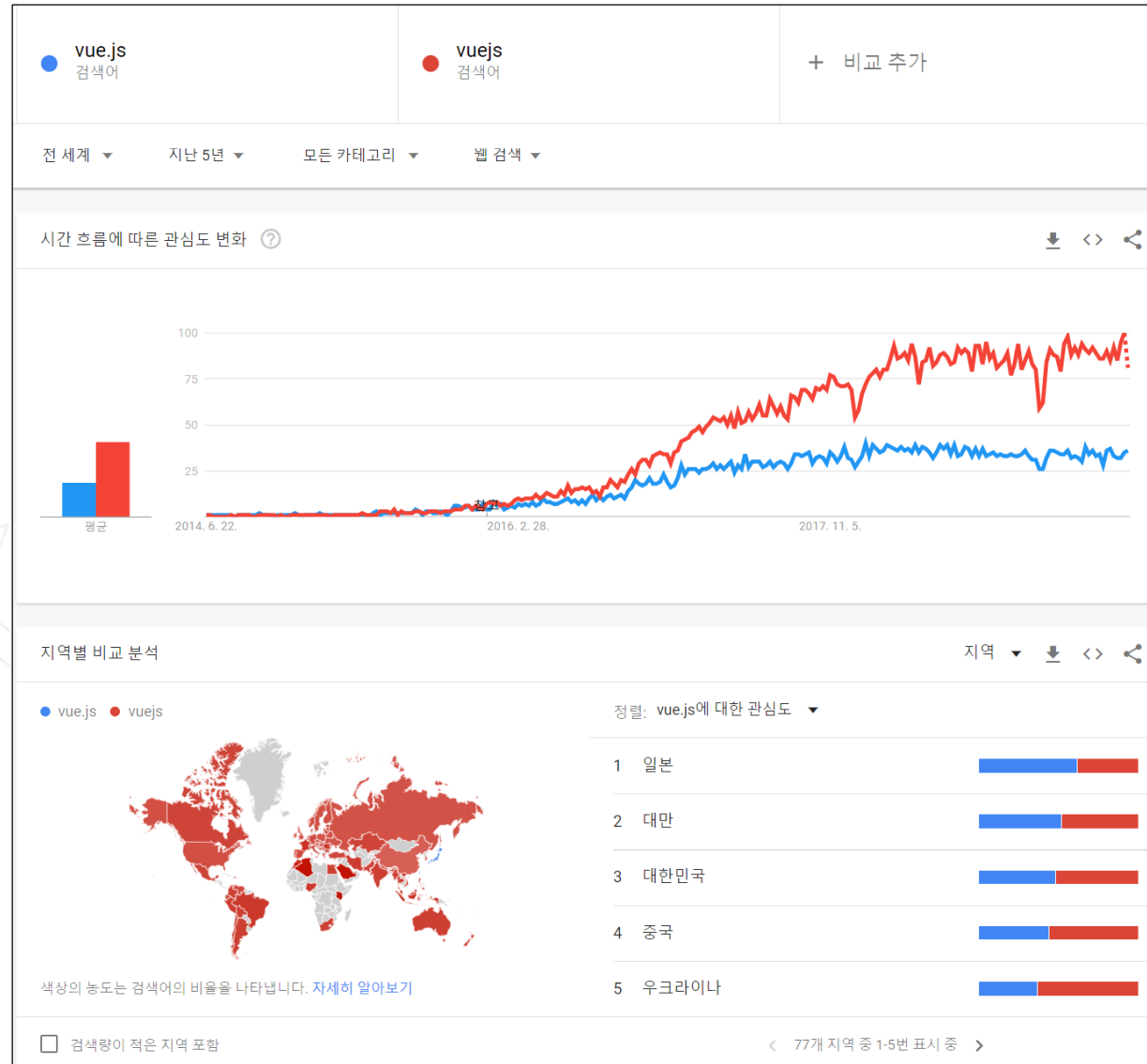
- 프레임워크
 - 개발자들이 개발 생산성을 높이기 위해 일정한 틀과 규칙에 따라 개발하도록 미리 구조를 정의해 놓은 도구
- 라이브러리
 - 자주 사용되는 기능들을 재사용할 수 있도록 정리해서 모아 놓은 기술

Vue.js 참고 사이트

- 한국 공식 사이트
 - <https://kr.vuejs.org/>
- API
 - <https://kr.vuejs.org/v2/api/>

Vue.js란?

- 구글 트렌드



Vue.js 의 장점

- 학습이 용이
- 리액트, 앵귤러에 비해 성능이 우수
- 리액트, 앵귤러의 장점 보유

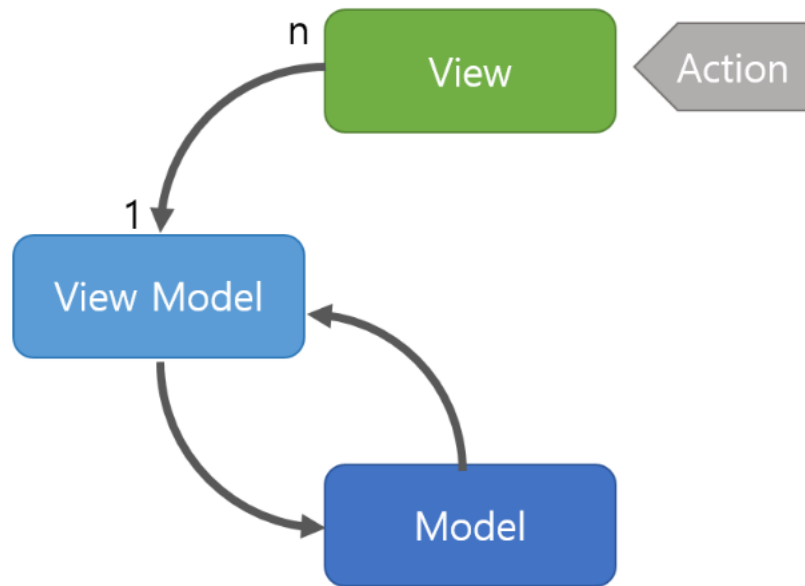
Kim Hye Kyung

Vue.js Architecture

- MVVM pattern

MVVM 패턴 기반의 ViewModel 화면 단 library

- MVVM이란?
 - Model + View + View Model를 합친 용어
 - Model과 View는 다른 패턴과 동일



Model : 어플리케이션에서 사용되는 데이터와 그 데이터를 처리하는 부분

View : 사용자에서 보여지는 UI 부분

View Model : View를 표현하기 위해 만든 View를 위한 Model
View를 나타내 주기 위한 Model이자 View를 나타내기 위한 데이터 처리를 하는 부분

MVVM 패턴 기반의 ViewModel 화면 단 library

MVVM 동작 순서

1단계 : 사용자의 Action들은 View를 통해 유입

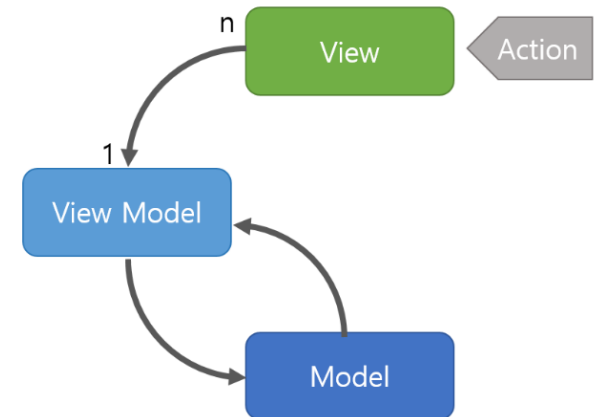
2단계 : View에 Action이 들어오면, Command 패턴으로 View Model에 Action을 전달

3단계 : View Model은 Model에게 데이터를 요청

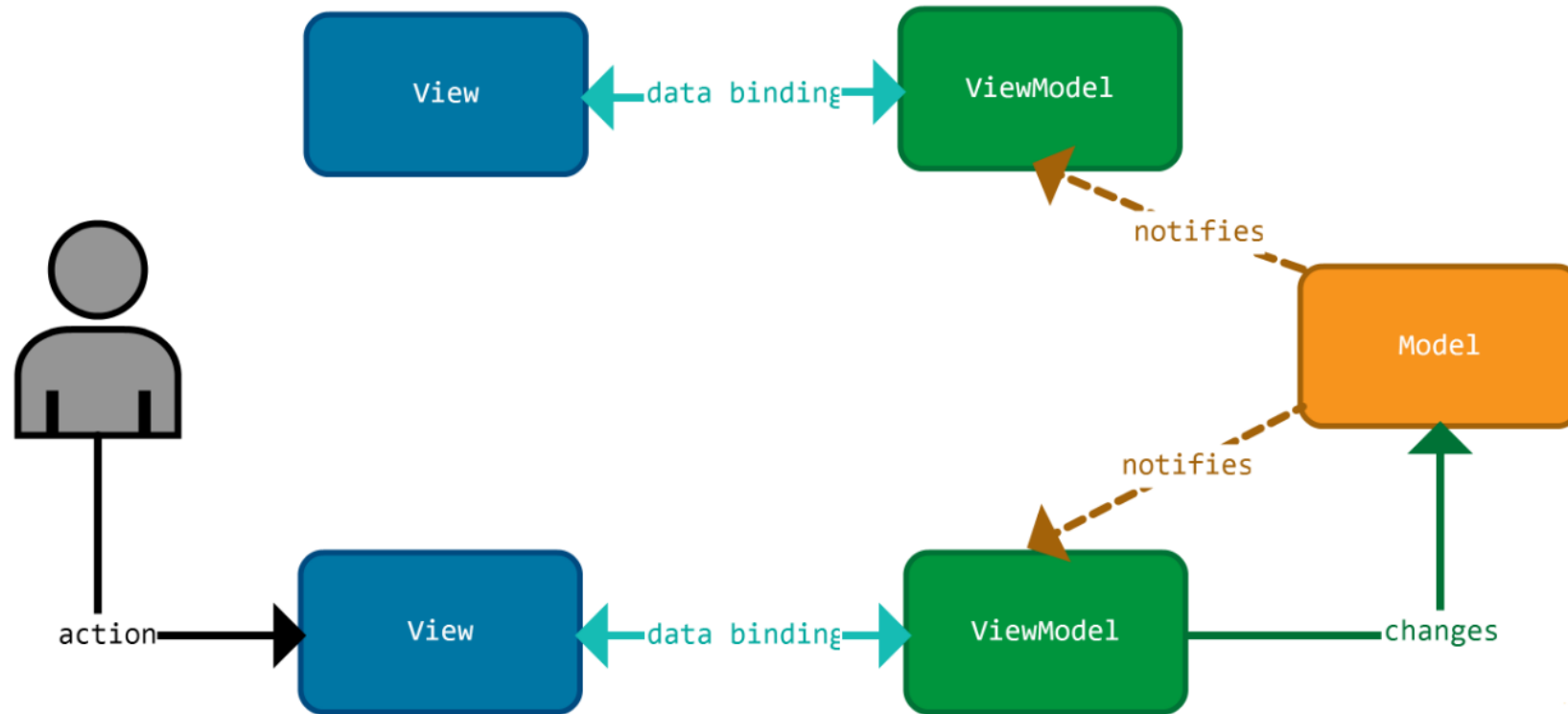
4단계 : Model은 View Model에게 요청받은 데이터를 응답

5단계 : View Model은 응답 받은 데이터를 가공하여 저장

6단계 : View는 View Model과 Data Binding하여 화면에 출력



Data Binding 이란?



MVVM 패턴 기반의 ViewModel 화면 단 library

- MVVM 특징
 - MVVM 패턴은 Command 패턴과 Data Binding 두 가지 패턴을 사용하여 구현되었음
 - Command 패턴과 Data Binding을 이용하여 View와 View Model 사이의 의존성을 삭제
 - View Model과 View는 1:n 관계
 - 화면 요소들을 제어하는 코드와 데이터 제어 로직을 분리하여 코드를 더 직관적으로 이해하고 유지보수가 용이하게 하기 위함

MVVM 패턴 기반의 ViewModel 화면 단 library

- MVVM 장점과 단점

- 장점

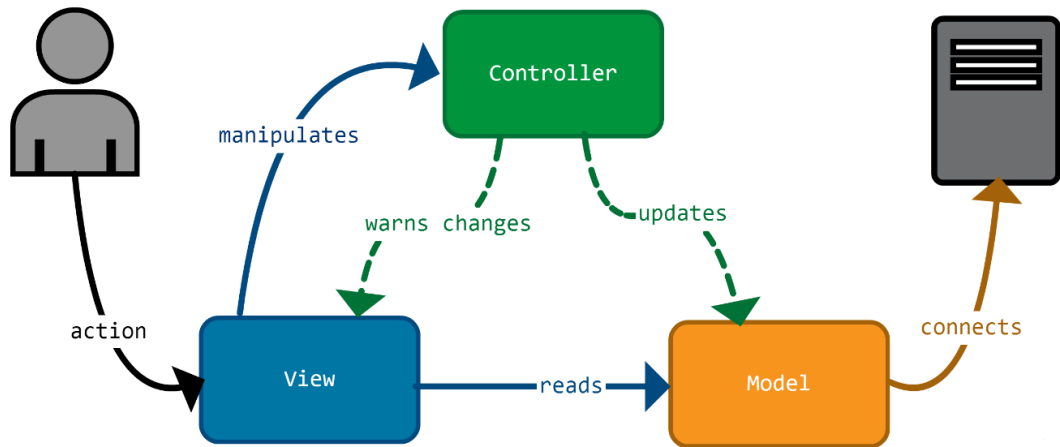
- MVVM 패턴은 View와 Model 사이의 의존성이 없음
 - Command 패턴과 Data Binding을 사용하여 View와 View Model 사이의 의존성 없음
 - 각각의 부분은 독립적이기 때문에 모듈화 하여 개발할 수 있음

- 단점

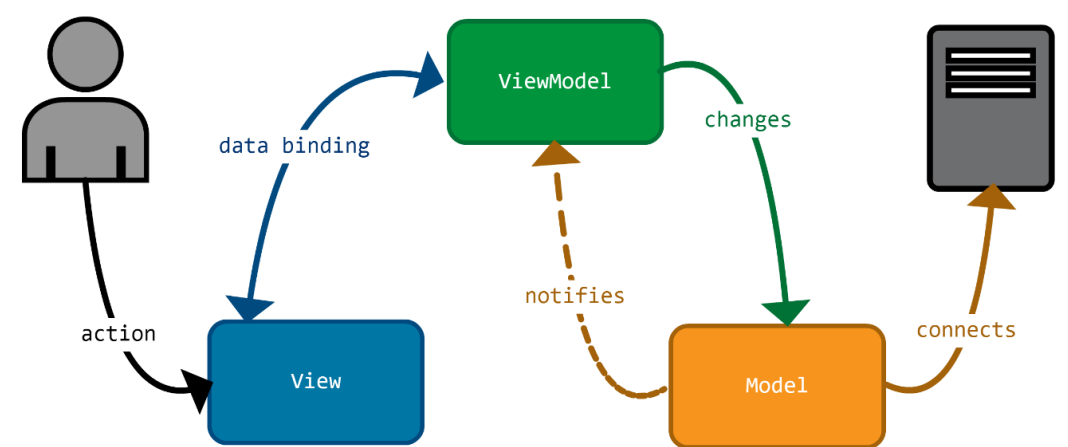
- MVVM 패턴의 단점은 View Model의 설계가 쉽지 않음

쉬어가기

- 각 pattern들에 대해 비교해 보기



MVC



MVVM

| Vue.js 특징

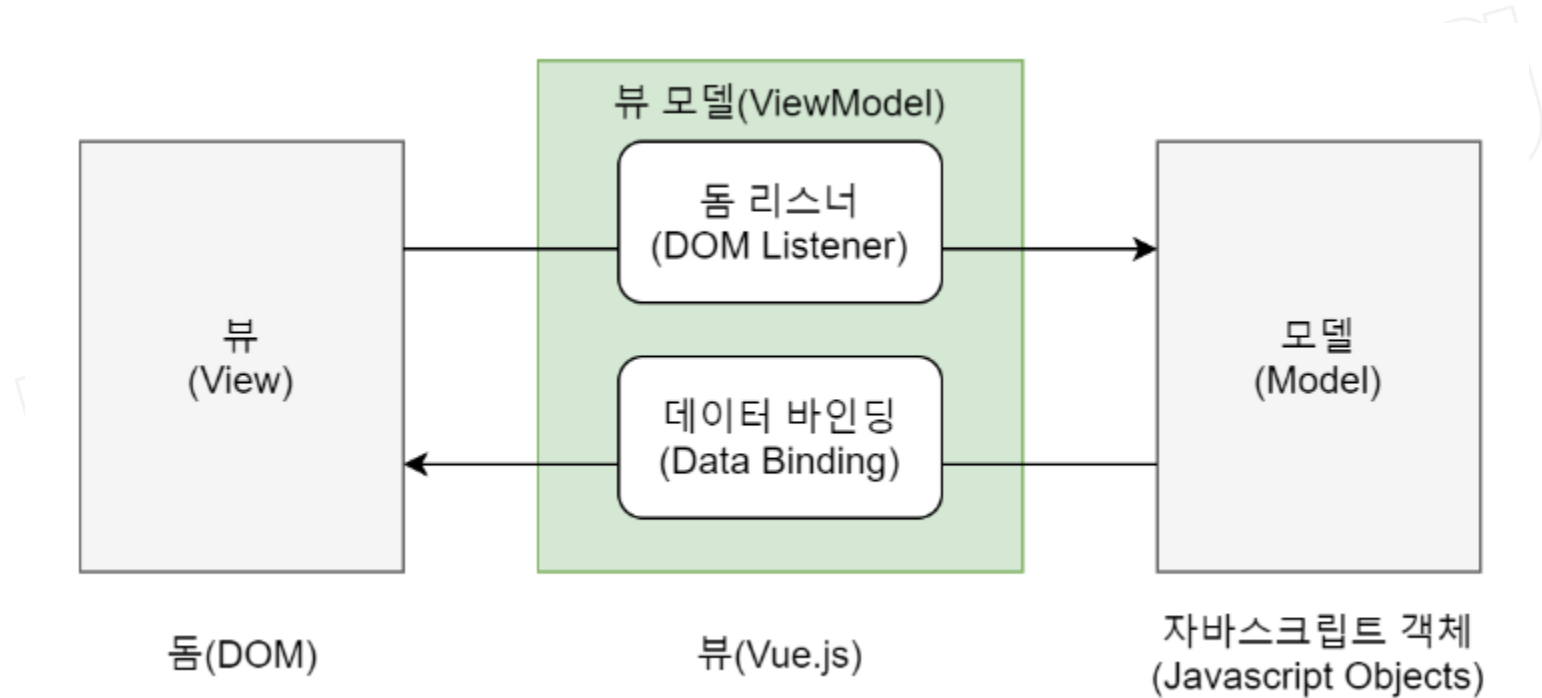
Vue.js 특징

MVVM 패턴 기반의 ViewModel 화면 단 library

컴포넌트 기반 Framework

MVVM 패턴 기반의 ViewModel 화면 단 library

- MVVM 구조에서의 Vue.js의 위치

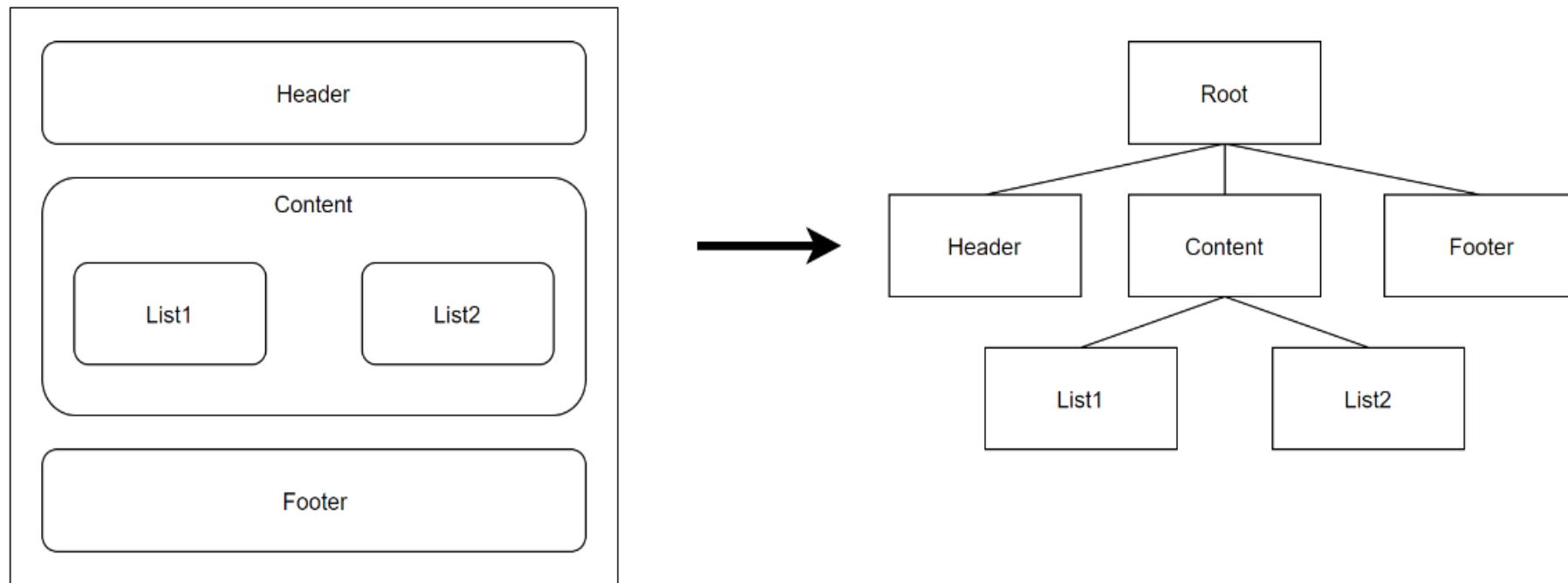


Vue.js 필요 용어

용어	설명
View	사용자에게 보이는 화면 화면의 요소가 변경되거나 조작이 일어날 때 즉각적으로 반응하여 화면의 데이터를 갱신하여 보여주는 기능
DOM	HTML 문서에 들어가는 요소 (tag, class등)의 정보를 담고 있는 tree
DOM Listener	Dom의 변경 내역에 대해 즉각적으로 반응하여 특정 로직을 수행
Model	데이터를 보유, 서버에서 가져온 데이터를 JavaScript 객체 형태로 저장
ViewModel	View와 Model의 접점, DOM Listener와 Data Binding을 제공하는 영역

컴포넌트 기반 Framework

- 레고 블록 조합하듯 Vue의 컴포넌트를 조합하여 화면 구성이 가능



화면을 컴포넌트로 구조화한 컴포넌트 간 관계도

| Vue.js 개발 환경 구축

개발 환경 필요 요소

- 크롬 브라우저
 - ECMAScript5 기능을 사용하기 때문에 IE8 이하 버전을 지원하지 않음
- 텍스트 에디터
 - Vue.js
- Node.js
 - <https://nodejs.org/en/>
 - Server Side에서 Java Script를 실행할 수 있는 실행 환경
 - CLI(Command Line Interface)를 이용하여 Vue.js 개발시에 필요
 - Vue CLI로 생성한 Project에서 프로토타이핑을 할 경우에도 Node.js 서버를 사용
 - Node package manager인 NPM도 자동 설치됨
- Vue 개발자 도구(Vue.js, devtools, 크롬 확장 플러그인)

개발 환경 구축

- Node.js



npm

Node.js와 npm(Node Package Manager)은
JavaScript가 거대한 오픈소스 생태계를 확보하는 데 결정적인 역할

개발 환경 구축

- NPM

- 노드 설치할 때 함께 설치됨
- 패키지의 저장소뿐 아니라 프로젝트에 필요 패키지를 추가하는 등 다양한 명령을 제공하는 패키지 관리 도구



```
C:\Users\WKimhyekyung>node -v  
v10.15.3
```

```
C:\Users\WKimhyekyung>npm -v  
6.4.1
```

```
C:\Users\WKimhyekyung>node -v && npm -v  
v10.15.3  
6.4.1
```

개발 환경 구축

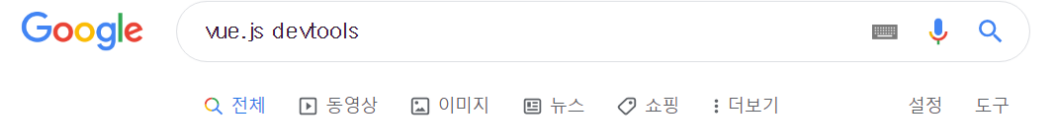
- Vue 개발자 도구 설치하기

- Vue의 크롬 플러그인

- Vue로 만든 웹 앱의 구조를 간편하게 디버깅 하거나 분석할 수 있음

- 지원 브라우저

- 크롬, 사파리, 파이어 폭스



Vue.js devtools - Google Chrome

<https://chrome.google.com/.../vuejs-devtools/nhdogimeijiglipcpn...> 이 페이지 번역하기

2019. 4. 14. - Chrome and Firefox DevTools extension for debugging Vue.js applications.

Vue.js devtools

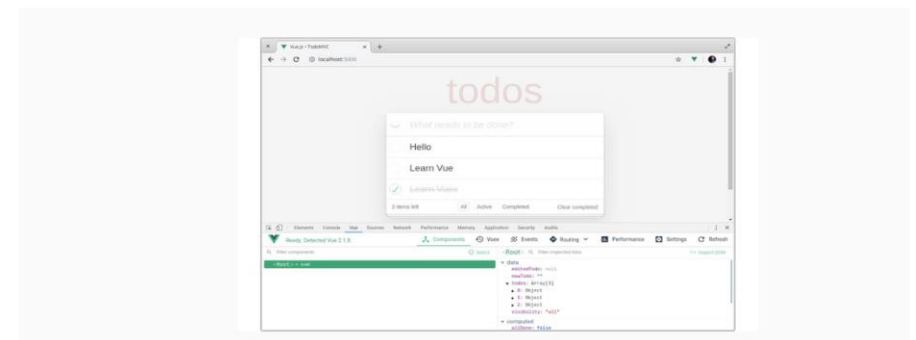
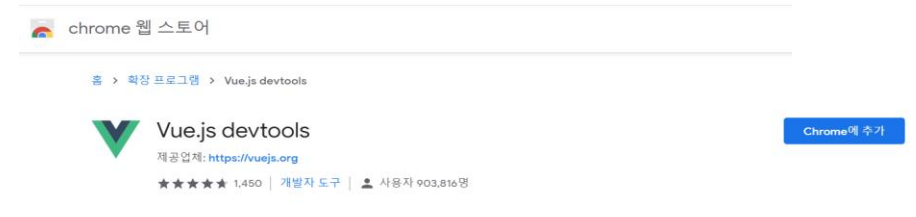
Chrome and Firefox DevTools extension for debugging Vue.js ...

[google.com 검색결과 더보기 »](#)

크롬 브라우저에서 Vue 앱 디버깅하기 - Vue.js 한국 사용자 모임

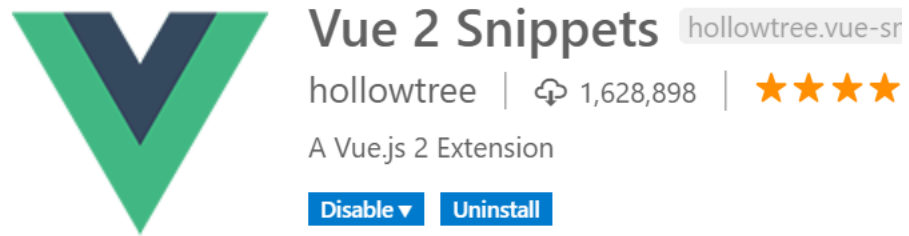
vuejs.kr/vue/2017/02/25/vue-chrome-debugging/

2017. 2. 25. - vue.js는 훌륭한 개발자 도구인 vue-devtools를 제공합니다. 현재 크롬 브라우저만 지원하고 있으며 여기에서 받으실 수 있습니다. vue-devtools.



개발 환경 구축

- VSC인 경우 추가



Details Contributions

Vue 2 Snippets for Visual Studio Code

개발 환경 구축

- Vue.js 설치 방법

방법1 : <script> tag 에 추가
가장 간단한 방법

library 다운 받고 src에 설정

CDN

<https://unpkg.com/vue>

방법2 : NPM을 통해 Vue.js 설치

대규모 서비스 구축 시 사용 권장
WebPack 등과 같은 번들러와 함께 사용 할 수 있음

방법3 : vue-cli를 사용하여 Vue.js 프로젝트 생성

Vue.js 프로젝트 생성을 지원하기 위해 [공식 CLI\(Command Line Interface\)](#)를 제공

개발 환경 구축

- 공식 사이트 참조 내용

HTML

```
<!-- 도움되는 콘솔 경고를 포함한 개발 버전 -->  
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

혹은:

HTML

```
<!-- 상용버전, 속도와 용량이 최적화됨. -->  
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
```

| Vue.js 프로젝트 만들기

Vue 개발 단계

HTML 파일 생성

Vue 소스 코드 추가

브라우저로 실행

Vue.js 구조 및 동작 이해를 위한 실습 단계

- 실습 단계
 - step01 – 기본
 - step02 – 조건, 반복등 응용
 - step03 – component 맛보기 등으로 구성

인스턴스 & 컴포넌트

화면 개발을 위한 필수 단위

인스턴스 & 컴포넌트

Vue 인스턴스

Vue 컴포넌트

Vue 컴포넌트 통신

Vue 인스턴스

- Vue로 화면을 개발하기 위해 필수적으로 생성해야 하는 기본 단위
- Vue 인스턴스 생성 문법
 - Vue 생성자로 생성

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

생성자 사용 사유

- 필요 옵션과 기능들을 정의해 놓고 확장이 용이하게 하기 위함
- 즉 속성과 메소드를 재사용하기 위함

Vue 인스턴스 옵션 속성

- Vue 생성자 속성과 옵션들
 - <https://kr.vuejs.org/v2/api/#data>

```
11 <body>
12   <div id="app">
13     {{ message }}
14   </div>
15   <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
16   <script>
17     new Vue({
18       el: '#app',
19       data: {
20         message: 'Hello Vue.js'
21       }
22     });
23   </script>
24 </body>
--
```

Vue.use
Vue.mixin
Vue.compile
Vue.version

옵션 / 데이터

data
props
propsData
computed
methods
watch

옵션들 / DOM

el
template
render
renderError

옵션 / 라이프사이클 훅

beforeCreate
created
beforeMount
mounted
beforeUpdate
updated
activated
deactivated
beforeDestroy
destroyed
errorCaptured

옵션 / 에셋

옵션 / 데이터

data

- 타입: `Object` | `Function`
- 제한: 컴포넌트에서 사용될 때만
- 상세:

Vue 인스턴스의 데이터 객체입니다. "객체" 형태로 만듭니다. 객체는 반드시 토타입 속성과 같은 기본 객체는 무태를 유지하는 동작은 관찰하지 않

일단 관찰되어지면, 루트 데이터 객터스 생성 이전에 모든 루트 수준의

인스턴스가 생성된 이후 원래 데이터는 데이터 객체에 있는 모든 속니다.

또는 `$` 로 시작하는 속성은 Vue 인스턴스에서 프록시 되지 않습니다

컴포넌트를 정의할 때 `data` 는 데이터 객체를 사용하면 생성된 모 `data` 함수를 제공함으로써 새로운 복사본을 반환할 수 있습니다.

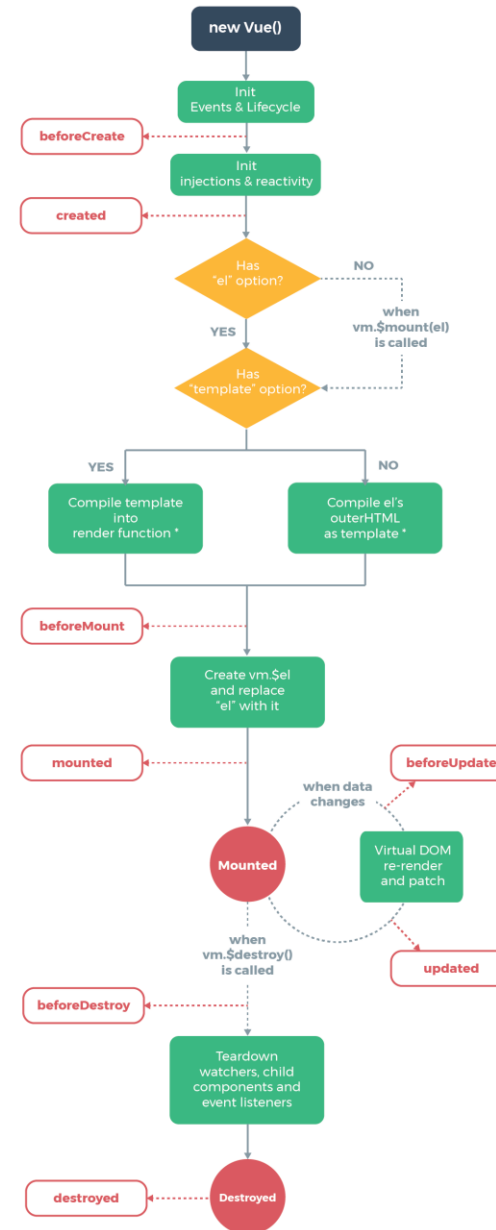
필요한 경우, `vm.$data` 를 JSON.p

Vue 인스턴스의 유효 범위

- 인스턴스의 유효 범위란?
 - Vue 인스턴스는 생성 후 HTML의 범위 내에서만 옵션 속성들이 적용
- 지역 컴포넌트와 전역 컴포넌트의 차이점을 이해하기 위한 필수 이론
- 인스턴스의 유효 범위
 - el 속성에 의해 정해짐
- 실행 Process
 - 1단계 : Vue 라이브러리 파일 로딩
 - 2단계 : 인스턴스 객체 생성(옵션 속성 포함)
 - 3단계 : 특정 화면 element에 인스턴스를 연결
 - 4단계 : 인스턴스의 내용이 화면 element로 변환
 - 5단계 : 변환된 화면이 사용자에게 노출

Vue 인스턴스 라이프 사이클

- <https://kr.vuejs.org/v2/guide/instance.html>



* template compilation is performed ahead-of-time if using a build step, e.g. single-file components

Vue 인스턴스 라이프 사이클

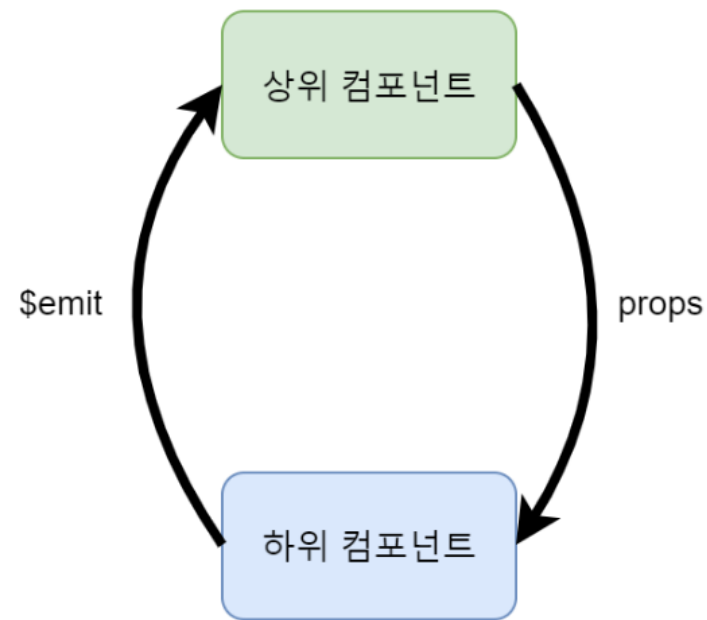
단계	설명
beforeCreate	인스턴스가 생성되고 최초로 실행되는 단계. data 속성과 methods 속성이 아직 인스턴스에 정의되지 않음, 화면 요소에도 접근할 수 없음
created	data, methods 속성이 정의된 단계. this.data 와 같이 사용할 수 있고 methods를 사용하여 로직을 실행할 수 있음 단, 화면 요소가 적용되기 전 따라서 template 속성에 정의된 요소에는 접근할 수 없음 data, methods 속성에 접근 할 수 있는 첫 단계로 서버에 데이터를 요청하여 받아오는 로직을 수행하기에 적합
beforeMount	template 속성에 지정한 마크업 속성을 render() 함수로 변환한 후 el 속성에 지정한 화면 요소에 인스턴스를 연결하기 전에 호출되는 단계
mounted	el 속성에서 지정한 화면 요소에 인스턴스가 연결되고 호출되는 단계 template 속성에 정의된 요소에 접근할 수 있어 화면 요소를 제어하는 로직을 수행하기 좋은 단계
beforeUpdate	Vue의 특징은 코드로 인한 데이터의 변화에 따라 화면이 즉각 반응하여 갱신되는 반응성이 있음 관찰되고 있는 데이터가 변화되어 화면으로 그려지기 직전에 호출되는 단계로, 변경 예정된 데이터의 값에 대한 로직을 수행하기 좋은 단계 단, 이 단계에서 값을 변경하는 로직을 넣더라도 화면이 반응하지 않음
update	데이터가 변경된 후 화면에 반영된 후 실행되는 단계. 데이터 변경 후 화면 요소에 대한 제어 로직을 추가하기적합
beforeDestroy	Vue 인스턴스가 소멸하기 직전에 단계. 아직 인스턴스에 접근할 수 있기 때문에 Vue 인스턴스의 데이터를 삭제하기 적합한 단계
destroyed	Vue 인스턴스가 소멸하고 나서 호출되는 단계. 인스턴스에 정의한 모든 속성이 제거되고 하위로 선언한 인스턴스까지 모두 소멸

Vue 컴포넌트 통신

1	컴포넌트 간 통신과 유효 범위
2	상위 하위 컴포넌트 관계
3	상위에서 하위 컴포넌트로 데이터 전달하기
4	하위에서 상위 컴포넌트로 이벤트 전달하기
5	관계 없는 컴포넌트 간 통신-이벤트 버스

Vue 컴포넌트간 통신

- 1. 컴포넌트 간 통신과 유효 범위
 - 각 컴포넌트의 유효 범위가 독립적 따라서 다른 컴포넌트의 값을 직접적으로 참조 불가
- 2. 상위 하위 컴포넌트 관계
 - Vue의 컴포넌트간 데이터 전달 방법의 기본
 - props 와 emit 사용
 - 상위(부모) -> 하위(자식) 컴포넌트간 데이터 전달 방법
 - 하위에서 상위로 : 이벤트 발생
 - 상위에서 하위로 : props 속성으로 데이터 전달



Vue 컴포넌트간 통신

- 3. 상위에서 하위 컴포넌트로 데이터 전달
 - 하위 컴포넌트에서 등록한 props 속성을 v-bind 라는 속성으로 상위 컴포넌트의 data 속성명 사용

```
//1. 속성 이름은 반드시 소문자
Vue.component('child-component', {
  props: ['속성이름']
});

//2. 하위 컴포넌트에서 등록한 props 속성을 v - bind 라는 속성으로 상위 컴포넌트의 data 속성명 입력
< child - component v - bind: 속성이름 = "상위컴포넌트 data 속성명" ></child - component >

//3. 예시
< div id = "app" >
  <my-component1 v-bind: propsdata="parentData"></my-component1>
  <my-component2></my-component2>
</div >
```

Vue 컴포넌트간 통신

- 4. 하위에서 상위 컴포넌트로 이벤트 전달하기
 - 하위 컴포넌트는 상위 컴포넌트로 이벤트를 발생하여 상위 컴포넌트의 method 호출 가능
 - 하위 컴포넌트로 내려보내는 props의 값을 조정할 수도 있음
 - 이벤트 발생과 수신은 \$emit과 v-on 속성 사용

//이벤트 발생과 수신은 \$emit과 v-on 속성을 사용

//1. 하위 컴포넌트 method에서 \$emit을 사용하여 이벤트를 발생시킴

```
this.$emit('이벤트명');
```

//2. 하위 컴포넌트에서 발생시킨 이벤트명으로 상위 컴포넌트 method 실행

```
<child-component v-on:이벤트명="상위컴포넌트 method명"></child-component>
```

Vue 컴포넌트간 통신

- 4. 하위에서 상위 컴포넌트로
이벤트 전달하기

```
//3. 예시
<div id="app">
  <!-- 부모 인스턴스의 viewData 메소드를 호출 -->
  <child-component v-on:show-log="viewData"></child-component>
</div>

Vue.component('child-component', {
  //child-component에서 template에 있는 v-on속성과 click이벤트를 연결
  //click 이벤트는 child-component의 showLog라는 method를 실행
  template : '<button v-on:click="showLog">데이터 보기</button>',
  methods : {
    showLog : function(){
      this.$emit('show-log');
    }
  }
});

new Vue({
  el : "#app",
  data : {
    message : '상위 컴포넌트에게 전달될 하위 데이터'
  },
  methods: {
    viewData : function(){
      console.log(this.message);
    }
  }
});
```

Vue 컴포넌트간 통신

- 5. 관계 없는 컴포넌트 간 통신-이벤트 버스
 - 컴포넌트 간의 통신은 상하위 관계에서만 적용
 - 서로 다른 컴포넌트 간의 통신 불가능
 - 해결책
 - 이벤트 버스 활용
 - 개발 방식
 - 애플리케이션 로직의 인스턴스 생성
 - 애플리케이션 로직과 무관한 새로운 인스턴스 1개 생성
 - 새 인스턴스를 이용하여 이벤트 전달
 - 보내는 컴포넌트에서는 `$emit()`
 - 받는 컴포넌트에서는 `$on()` 구현

Vue 컴포넌트간 통신

- 5. 관계 없는 컴포넌트 간 통신-이벤트 버스
 - 버튼 클릭시 showLog() 실행되었을 때
eventBus의 이벤트가 발생
 - 발생한 이벤트는 상위 컴포넌트의
created() 내의 eventBus.\$on()에 전달되어짐
 - 이벤트 버스 활용시 props속성없이도
다른 컴포넌트 간에 직접적인 데이터 전달 가능

```
//1. 이벤트 버스를 위한 Vue 인스턴스를 생성, 컴포넌트간 통신 매개체  
var eventBus = new Vue();
```

```
//2. 하나의 컴포넌트에서 eventBus 인스턴스를 이용해 $emit을 사용  
// 작성된 이벤트명에 대한 이벤트를 발생하고 데이터를 전달
```

```
methods: {  
  메소드명: function() {  
    eventBus.$emit('이벤트명', 데이터, ...);  
  }  
}
```

```
//3. eventBus.$on을 이용하여 이벤트명에 대한 이벤트 발생을 감지  
// 전달된 데이터를 function에서 사용
```

```
methods: {  
  메소드명2: function() {  
    eventBus.$on('이벤트명', function(value, ...){  
      ...  
    });  
  }  
}
```

| Vue 라우터

Vue 라우터

1	라우팅이란?
2	Vue 라우터
3	Nested 라우터
4	Nested Vue

라우팅이란?

- 웹 페이지 간의 이동 방법 의미
- 현대 웹 앱 형태 중 하나인 싱글 페이지 애플리케이션(SPA, Single Page Application)에서 주로 사용
- 장점
 - 화면 간의 전환이 매끄러움
 - 요청, 응답 진행 시 발생하는 깜빡임 현상 해소
 - 애플리케이션의 사용자 경험을 향상
 - 빠르게 화면 조작 가능

리액트와 앵귤러 모두 라우팅을 이용한 화면 전화 기술 적용,
Front end framework 사용 없이도 HTML 파일들로 Java Script library들로 라우팅 방식의 페이지 이동 구현 가능

라우팅이란?

- Vue에서 라우팅 기능을 구현 할 수 있도록 지원하는 공식 library
- Vue로 만든 페이지 간에 자유롭게 이동 가능

태 그	설 명
<code><router-link to="url"></code>	페이지 이동 태그 브라우저에는 <code><a></code> tag 처럼 표시 to 속성에 지정한 url로 이동
<code><router-view></code>	페이지 표시 태그 변경되는 url에 따라 해당 컴포넌트를 화면에 출력하는 영역

Vue 라우터

- \$mount() 란 ?
 - el 속성과 동일
 - 인스턴스를 화면에 적용하는 역할

웹 앱 구현 시 다중 화면

- 화면이 여러 개의 컴포넌트로 분할된 경우

네스티드 라우터

라우터로 페이지를 이동할 때 최소 2개 이상의 컴포넌트를 화면에 나타낼 수 있음

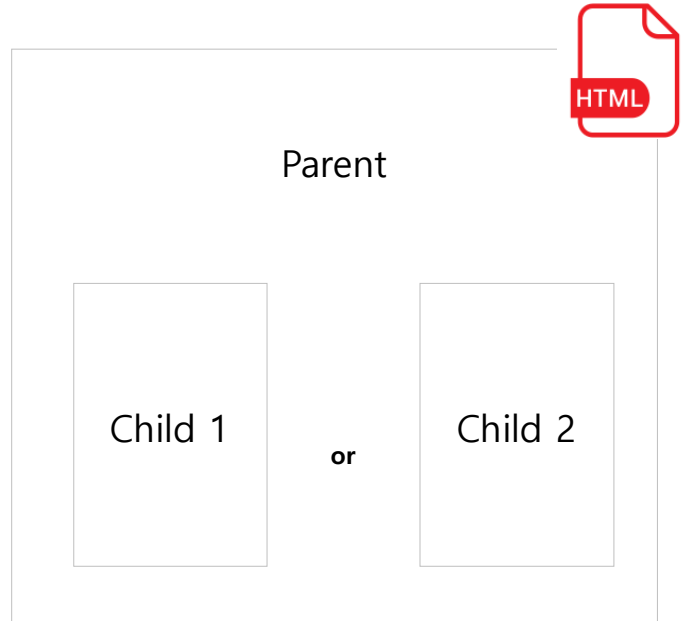
상위 컴포넌트 한 개에 하위 컴포넌트 한 개를 포함하는 구조

한번에 많은 컴포넌트 표시할 경우엔 한계가 있음

네임드 Vue

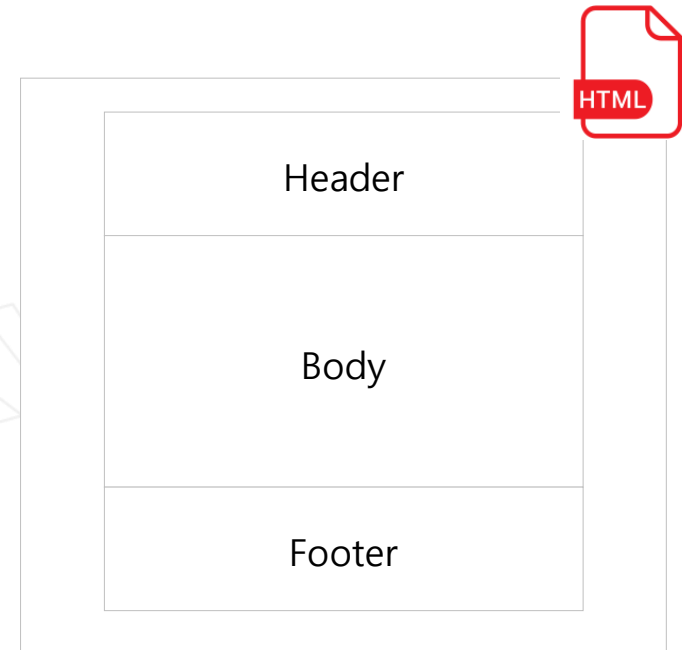
특정 page로 이동했을 경우 여러 개의 컴포넌트를 동시에 표시하는 라우팅 방식

Vue 라우터 - 웹 앱 구현 시 다중 화면



Nested Router

상위 하위 구조로 표현하는 routing 방식
상위 컴포넌트가 하위 컴포넌트 포함하는 형식

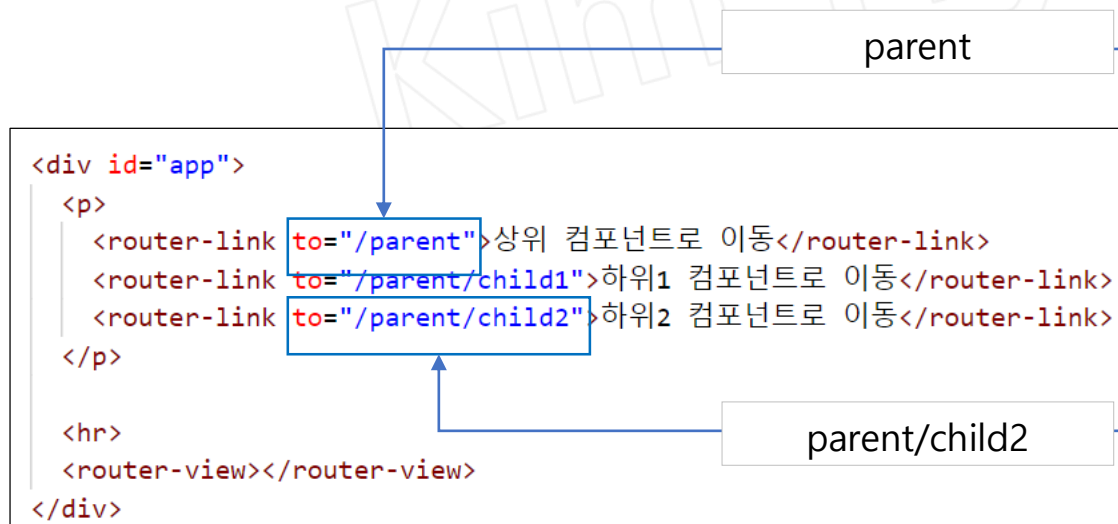


Named Router

여러 개의 컴포넌트를 동시에 표현하는 routing 방식
같은 레벨에서 여러 개의 컴포넌트를 한번에 표시

Nested 라우터

- 라우터로 페이지를 이동할 때 최소 2개 이상의 컴포넌트를 화면에 나타낼 수 있음
- 화면을 구성하는 컴포넌트 수가 적을 경우 유용
- 상위 컴포넌트 1개에 하위 컴포넌트 N개로 구성 가능



```
//상위 컴포넌트
var parent = {
  template: `
    <div>
      상위 Component
      <router-view></router-view> <!-- 하위 컴포넌트 출력 위치 -->
    </div>
  `
};

//하위 컴포넌트 : child1, child2
var child1 = { template: '<p>하위1 Component</p>' };
var child2 = { template: '<p>하위2 Component</p>' };

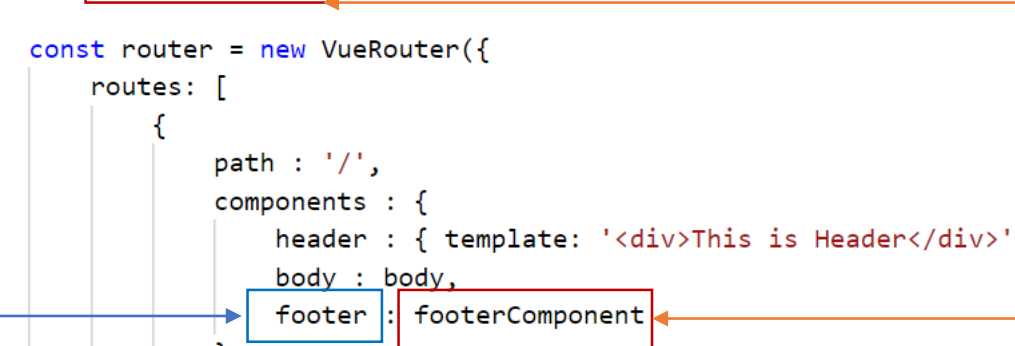
//parent 컴포넌트 라우터의 자식으로 child1, child2으로 구성
var routes = [
  {
    path: '/parent',
    component: parent,
    children: [
      {
        path: 'child1',
        component: child1
      },
      {
        path: 'child2',
        component: child2
      }
    ]
  }
];
```

네임드 Vue

- 특정 페이지로 이동시 여러 개의 컴포넌트를 동시에 표시하는 라우팅 방식
- 같은 레벨에서 여러 개의 컴포넌트를 한번에 표시

```
<div id="app">  
  <router-view name="header"></router-view>  
  <router-view name="body"></router-view>  
  <router-view name="footer"></router-view>  
</div>
```

```
// var h = { template: '<div>This is Header</div>' };  
var body = { template: '<div>This is Body</div>' };  
var footerComponent = { template: '<div>This is Footer</div>' };  
  
const router = new VueRouter({  
  routes: [  
    {  
      path: '/',  
      components: {  
        header: { template: '<div>This is Header</div>' },  
        body: body,  
        footer: footerComponent  
      }  
    }  
  ]  
});
```



| Vue HTTP 통신

Vue와 Ajax

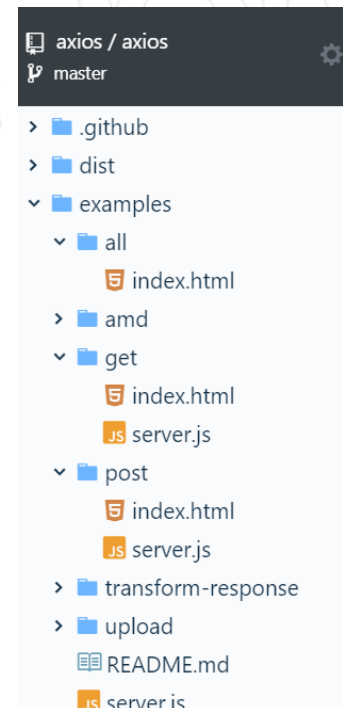
- Axios
 - Vue의 비동기 개발 지원하는 library
 - Vue Framework의 필수 library
 - Vue 커뮤니티에서 가장 많이 사용하는 HTTP 통신 라이브러리
 - 별도의 로직을 구현할 필요 없이 주어진 API만으로 간편하게 비동기 로직 구현이 가능

| Axios

Kim Hye Kyung

Axios 개요

- HTTP 통신을 위한 Java Script library[client library]
- XMLHttpRequest[XHR] 요청을 매우 쉽고 간결하게 처리
- Vue 커뮤니티에서 가장 많이 사용 및 권고하고 있는 HTTP client 통신 library
- 장점
 - 구형 브라우저 지원
 - CSRF 보호 기능이 내장되어 있음
 - Cross-site request forgery[CSRF] page 참조
 - **JSON 데이터 자동 변환**
- 참고 사이트
 - <https://github.com/axios/axios>
 - example 디렉토리 예제들 보기



Axios & Promise

- Promise 기반의 API 형식으로 개발
 - 다양하게 제공되는 API를 사용해서 간편하게 Ajax 구현이 가능

Promise란?

ES6에서 비동기 처리를 위한 또 다른 패턴으로 프로미스(Promise)를 도입

Promise는 전통적인 콜백 패턴이 가진 단점을 보완하며 비동기 처리 시점을 명확하게 표현

Cross-site request forgery[CSRF]

- 사이트간 요청 위조(크로스 사이트 요청 위조)
- 웹 사이트 취약점 공격의 하나로 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위인 수정, 삭제, 등록등을 특정 웹사이트에 요청하게 하는 공격 의미
- 유명 경매 사이트인 옥션에서 발생한 개인정보 유출 사건에서 사용된 공격 방식 중 하나
- 참고

- 위키백과

공격 과정 [편집]

1. 이용자는 웹사이트에 로그인하여 정상적인 쿠키를 발급받는다
2. 공격자는 다음과 같은 링크를 이메일이나 게시판 등의 경로를 통해 이용자에게 전달한다.
`http://www.geocities.com/attacker`
3. 공격용 HTML 페이지는 다음과 같은 이미지태그를 가진다.

```
<img src= "https://travel.service.com/travel_update?.src=Korea&.dst=Hell">
```

해당 링크는 클릭시 정상적인 경우 출발지와 도착지를 등록하기위한 링크이다. 위의 경우 도착지를 변조하였다.

4. 이용자가 공격용 페이지를 열면, 브라우저는 이미지 파일을 받아오기 위해 공격용 URL을 연다.
5. 이용자의 승인이나 인지 없이 출발지와 도착지가 등록됨으로써 공격이 완료된다. 해당 서비스 페이지는 등록 과정에 대해 단순히 쿠키를 통한 본인확인 밖에 하지 않으므로 공격자가 정상적인 이용자의 수정이 가능하게 된다.

Axios(엑시오스) 설치 및 요청 방식

Axios CDN

```
<script src="https://unpkg.com/axios/dist/axios.min.js"> </script>
```

```
axios.get("URL 주소").then().catch();
```

해당 URL로 HTTP Get 요청을 보냄
서버가 정상 응답시 then() 실행
오류 발생시 catch() 실행

```
axios.post("URL 주소").then().catch();
```

해당 URL로 HTTP Post 요청을 보냄
서버가 정상 응답시 then() 실행
오류 발생시 catch() 실행

주의 사항
**- Cross-Site HTTP Requests 를 위한
설정 필수**

```
axios({  
  method : 'get',  
  url : 'URL 주소',  
  ....  
});
```

옵션 속성

HTTP 요청에 대한 상세한 속성들을 직접 정의
데이터 요청을 보낼 URL, HTTP 요청 방식, 보내는 데이터 유형, 기타 등등

Ajax Programming 특징

- CORS
 - 웹 브라우저에서 외부도메인 서버와 통신하기 위한 방식을 표준화한 스펙
 - JavaScript Ajax 크로스 도메인 요청의 기본 특징
 - 웹 개발시 자바스크립트로 외부 서버의 경로로 요청시에는 에러 발생

No 'Access-Control-Allow-Origin' header is present on the requested resource.
Origin '[요청한 도메인]' is therefore not allowed access.

- 외부로 요청이 안되는 사유
 - Java Script 엔진 표준 스펙에 동일한 출처 정책이라는 보안 규칙이 있기 때문
 - 동일 출처 정책(Same-Origin Policy)
- 정책 사이트
 - <https://www.w3.org/TR/cors/#cross-origin-request-with-preflight0>

Cross Origin Resource Sharing - CORS

- HTTP 요청은 기본적으로 Cross-Site HTTP Requests가 가능
 - 태그로 다른 도메인의 이미지 파일을 가져오거나, <link> 태그로 다른 도메인의 CSS를 가져오거나, <script> 태그로 다른 도메인의 JavaScript 라이브러리를 가져오는 것이 모두 가능
- W3C에서 CORS라는 이름의 권고안 파생 사유
 - <script></script> script tag 내부에서 생성된 Cross-Site HTTP Requests는 Same Origin Policy를 적용 받기 때문에 Cross-Site HTTP Requests가 불가능
 - Ajax가 보편적으로 사용되면서 <script></script>로 둘러싸여 있는 스크립트에서 생성되는 XMLHttpRequest에 대해서도 Cross-Site HTTP Requests가 가능해야 한다는 요구 폭증

Cross Origin Resource Sharing - CORS

- CORS 적용시 POST 방식의 요청인 경우 경우 추가 설정
 - header : application/x-www-form-urlencoded

```
headers: { "Content-Type": "application/x-www-form-urlencoded" }
```

참고 : JSON 데이터 : 실습을 위한 필요사항

- JSON 데이터 제공하는 온라인 사이트
 - <https://jsonplaceholder.typicode.com/posts/1/comments>
 - <https://jsonplaceholder.typicode.com/albums/1/photos>
 - <https://jsonplaceholder.typicode.com/users/1/albums>
 - <https://jsonplaceholder.typicode.com/users/1/todos>
 - <https://jsonplaceholder.typicode.com/users/1/posts>
 - <https://api.androidhive.info/contacts/>
- JSON viewer
 - <https://jsonformatter.org/json-viewer>
- Test tool
 - Post Man

Ajax Programming을 위한 Axios : get 방식

```
<div id="dataView"></div>

<div id="dataView2"></div>

<script src="https://unpkg.com/axios/dist/axios.min.js"></script>

<script>
  axios.get("https://jsonplaceholder.typicode.com/users/1/todos?userId=1&id=2")
    .then(response => {
      data = response;
      //객체 타입 확인
      console.log(typeof(data));
      //보유하고 있는 속성들 확인
      console.log(data);
      //속성들 확인후 제공받은 속성명들 활용
      console.log(data.data[0].userId);
      //응답 받은 데이터의 첫번째 배열 요소의 userId값을 화면에 출력
      document.querySelector("#dataView").innerHTML = data.data[0].userId;
    })
    .catch(error => {console.log(error);});

```

object

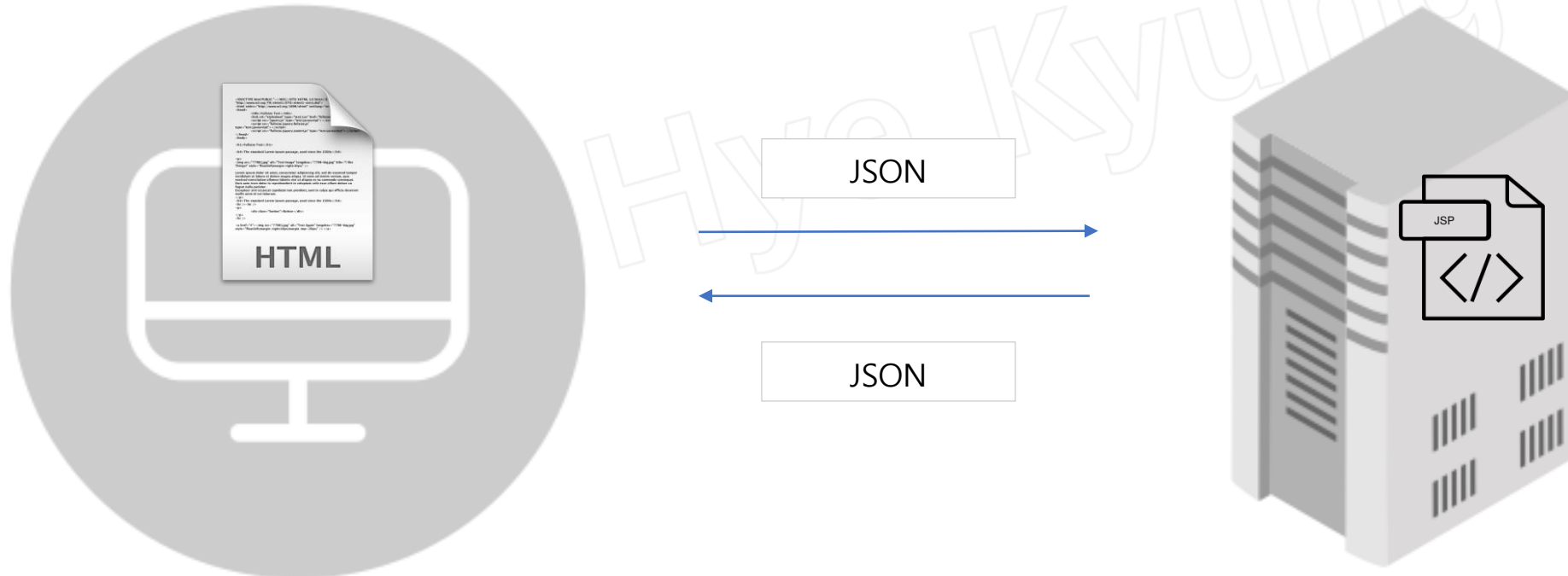
```
▼ {data: Array(1), status: 200, statusText: "", headers:
  ► config: {url: "https://jsonplaceholder.typicode.com/u
  ▼ data: Array(1)
    ► 0: {userId: 1, id: 2, title: "quis ut nam facilis et
      length: 1
      ► __proto__: Array(0)
    ► headers: {pragma: "no-cache", content-type: "applicat
    ► request: XMLHttpRequest {onreadystatechange: f, ready
      status: 200
      statusText: ""
    ► __proto__: Object
```

1

```
▼ data: Array(1)
  ▼ 0:
    completed: false
    id: 2
    title: "quis ut nam facilis et officia qui"
    userId: 1
    ► __proto__: Object
  length: 1
  ► __proto__: Array(0)
```

Ajax Programming을 위한 Axios : post 방식

- 실습 구조
 - Server와 Client간의 JSON 포맷을 활용한 요청, 응답



Ajax Programming을 위한 Axios : post 방식

```
<script src="https://unpkg.com/axios/dist/axios.min.js"></script>
<script>
  (function () {
    var output = document.getElementById('output');

    document.getElementById('post').onclick = function () {
      var data = document.getElementById('data').value;

      //post 방식인 경우의 parameter 구성 방식
      //생략시 데이터 전송 불가
      let sendData = new URLSearchParams();
      sendData.append('data', data);

      //post 방식인 경우 추가 해야 할 headers 설정
      1 axios.post('http://localhost:8080/axiosAjaxServer/postAxios.jsp', sendData,
        { headers: { "Content-Type": "application/x-www-form-urlencoded" } })
      3 then(function (res) {
        output.id = 'output';

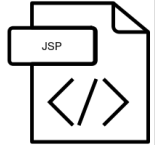
        console.log(1, res);
        console.log(2, typeof(res));
        console.log(3, res.data);
        console.log(4, res.data.inputData.name);

        let outputData = '이름 : ' + res.data.inputData.name + ", 나이 : " +
          res.data.inputData.age;

        output.innerHTML = outputData;
      })
      .catch(function (err) {
        output.id = 'output';
        output.innerHTML = err.message;
      });
    });
  })();
</script>
```



```
postAxios.jsp
1 <%@ page language="java" contentType="text/json; charset=utf-8"
2 <%
3 //post 방식의 요청 데이터에 대한 한글 인코딩 설정
4 request.setCharacterEncoding("utf-8");
5
6 //crose domain을 허용하기 위한 필수 설정
7 response.setHeader("Access-Control-Allow-Origin", "*");
8 %>
9 {"inputData": ${param.data}}
```



Console Sources Network Performance Memory Application

Filter Default levels

```
1 step02_axiosPost.html:69
  {data: {...}, status: 200, statusText: "", headers: {...}, config: {...}, ...}
  ▶ config: {url: "http://localhost:8080/axiosAjaxServer/postAxios.jsp", me...
  ▶ data: {inputData: {...}}
  ▶ headers: {content-length: "49", content-type: "text/json; charset=utf-8"}
  ▶ request: XMLHttpRequest {onreadystatechange: f, readyState: 4, timeout: ...
    status: 200
    statusText: ""
    __proto__: Object
  2 "object" step02_axiosPost.html:70
  3 {inputData: {...}} step02_axiosPost.html:71
    ▶ inputData: {name: "유재석", age: 40}
    ▶ __proto__: Object
  4 "유재석" step02_axiosPost.html:72
```

JSON 포맷으로 입력하기

```
{ "name": "유재석", "age": 40 }
```

POST 방식으로 데이터 전송

JSON 포맷으로 입력하기

```
{ "name": "유재석", "age": 40 }
```

POST 방식으로 데이터 전송

이름 : 유재석, 나이 : 40

김혜경 [topickim@naver.com]

Axios Programming 응답 데이터

- 실제 사용되는 데이터 및 다양한 데이터들이 포함되어 응답
- 주의사항
 - data 라는 property 로 받은 데이터값 활용
 - data property 데이터
 - JSON 으로 변환하지 않아도 JSON 객체로 응답 받음

| Axios & Vue Ajax Programming

Vue & Ajax Programming을 위한 Axios(엑시오스)

- get 방식 예시

```
<div id="app">
  <button v-on:click="getData">1. get 방식으로 데이터 요청 및 응답 받기</button>
  <p>get 방식으로 응답된 이름 : {{data}}</p>
</div>

<script src="https://cdn.jsdelivr.net/npm/vue@2.5.2/dist/vue.js"></script>
<script src="https://unpkg.com/axios/dist/axios.min.js"></script>
<script>
  new Vue({
    el: '#app',
    methods: {
      getData: function () {
        var self = this;
        let param = 'get방식으로 전송하는 데이터';
        axios.get('jsonResponse.jsp?msg='+param).then(function (response) {
          self.data = response.data.name;
        });
      }
    },
    data : {
      data : ''
    }
  });
</script>
```

1. get 방식으로 데이터 요청 및 응답 받기

get 방식으로 응답된 이름 :

응답된 데이터

```
{name: "김혜경", age: 20, msg: "get 방식으로 전송하는 데이터"}
```

1. get 방식으로 데이터 요청 및 응답 받기

get 방식으로 응답된 이름 : 김혜경

Vue & Ajax Programming을 위한 Axios(엑시오스)

- post 방식 예시

```
<div id="appPost">
  <button v-on:click="postData">2. post 방식으로 데이터 요청 및 응답 받기</button>
  <p>post 방식으로 응답된 이름 : {{data}}</p>
</div>

<script src="https://cdn.jsdelivr.net/npm/vue@2.5.2/dist/vue.js"></script>
<script src="https://unpkg.com/axios/dist/axios.min.js"></script>
<script>
  new Vue({
    el: '#appPost',
    methods: {
      postData: function () {

        var self = this;

        let formData = new URLSearchParams();
        formData.append('msg', "post방식으로 서버에 전송하는 데이터");

        axios.post('jsonResponsePost.jsp', formData).then(function (response){
          self.data = response.data.name;
        });

      },
      data : {
        data : ''
      }
    });
  });
</script>
```

2. post 방식으로 데이터 요청 및 응답 받기

post 방식으로 응답된 이름 :

응답된 데이터

{name: "김혜경", age: 20, msg: "post 방식으로 전송한 데이터"}

2. post 방식으로 데이터 요청 및 응답 받기

post 방식으로 응답된 이름 : 김혜경

| Vue 템플릿

Vue 템플릿

- Vue 템플릿이란?
 - HTML, CSS 등의 마크업 속성과 Vue 인스턴스에서 정의한 데이터 및 로직들을 연결하여 사용자가 브라우저에서 볼 수 있는 형태의 HTML로 변환해 주는 속성



Vue 템플릿

1	데이터 바인딩
2	Vue 디렉티브(Directive)
3	이벤트 처리
4	고급 템플릿 기법

데이터 바인딩(Data Binding)

- 데이터 바인딩은 HTML 화면 요소를 Vue 인스턴스의 데이터와 연결하는 것
- 주요 문법
 - v-bind 속성 사용
 - {{}} 템플릿 문법 사용[콧수염 괄호]

데이터 바인딩(Data Binding) : v-bind

- HTML 화면 요소를 Vue 인스턴스의 데이터와 연결하는 것
- id, class, style 등의 HTML 속성값에 Vue 데이터 값을 연결할 때 사용
- 주요 문법
 - 형식은 v-bind 속성으로 지정할 HTML 속성이나 props 속성 앞에 접두사로 붙여줌

```
<div id="app">
  <p v-bind:id="idA">id 속성을 활용한 데이터 바인딩</p>
  <p v-bind:class="classA">class 속성을 활용한 데이터 바인딩</p>
  <p v-bind:style="styleA">style 속성을 활용한 데이터 바인딩</p>
</div>

<script src="https://cdn.jsdelivr.net/npm/vue@2.5.2/dist/vue.js"></script>
<script>
  new Vue({
    el: '#app',
    data: {
      idA: 'id 속성 value',
      classA: 'class 속성 value',
      styleA: 'color: blue'
    }
  });
</script>
```

v-bind를 사용한 템플릿 구성하기

id 속성을 활용한 데이터 바인딩

class 속성을 활용한 데이터 바인딩

style 속성을 활용한 데이터 바인딩

Elements Console Sources Vue Network Performance

```
<br>
<div id="app">
  <p id="id 속성 value">id 속성을 활용한 데이터 바인딩</p>
  <p class="class 속성 value">class 속성을 활용한 데이터 바인딩</p>
  <p style="color: blue;">style 속성을 활용한 데이터 바인딩</p>
</div>
```

자바스크립트 template : {{}}

- 콧수염 괄호
- Vue 인스턴스의 데이터를 HTML 태그에 연결하는 가장 기본적인 텍스트 적용 방식
- Vue 뿐만 아니라 다른 언어나 프레임워크에서도 자주 사용되는 템플릿 문법

```
<div id="app">
  <p>{{ message }}</p>
  <p>{{ message + '입니다' }}</p>
  <p>{{ message.split('').reverse().join('') }}</p>
</div>

<script src="https://cdn.jsdelivr.net/npm/vue@2.5.2/dist/vue.js"></script>
<script>
  new Vue({
    el: '#app',
    data: {
      message: 'JavaScript Template'
    }
  });
```

JavaScript Template

JavaScript Template입니다

etalpmeT tpircSavaJ

etalpmeT tpircSavaJ

Vue 디렉티브(Directive)

- 특별한 속성
 - 표준 HTML에 독자적으로 정의한 속성으로 속성값 표현식의 변화에 따라 DOM을 조작
 - HTML tag 내에 v- 접두사를 가지는 모든 속성을 의미
- 주 용도
 - 화면의 요소를 더 쉽게 조작하기 위해 사용하는 기능
 - Vue의 데이터 값이 변경되었을 때 화면의 요소들이 Reactive하게 반응하여 변경된 데이터 값에 따라 갱신
 - 화면의 요소를 직접 제어할 필요 없이 Vue의 directive를 활용하여 화면 요소들 조작이 가능

Vue 디렉티브(Directive)

- HTML tag 내에 v- 접두사를 가지는 모든 속성을 의미
- 화면의 요소를 더 쉽게 조작하기 위해 사용하는 기능
- 화면의 요소들이 Reactive 하게 반응하여 변경된 데이터 값에 따라 갱신
- 결론
 - 화면의 요소를 직접 제어할 필요없이 Vue의 디렉티브를 활용하여 화면 요소들 조작 가능

Vue 디렉티브(Directive)

디렉티브	설 명
v-if	Vue 데이터 값의 참, 거짓 여부에 따라 해당 HTML 태그를 화면에 표시하거나 표시하지 않음 평가값에 따른 DOM 요소 추가, 제거 해당 태그를 완전히 삭제
v-for	Vue 데이터의 개수 만큼 해당 HTML 태그를 반복 출력
v-show	v-if와 유사하게 데이터의 진위 여부에 따라 해당 HTML 태그를 화면에 표시하거나 표시하지 않음 v-show는 css효과만 display:none; 으로 주어 실제 태그는 남아 있고 화면 상으로만 보이지 않음
v-bind	HTML 태그의 기본 속성과 Vue 데이터 속성을 연결
v-on	화면의 이벤트를 감지하여 처리할 때 사용 가령, v-on:click 은 해당 태그의 클릭 이벤트를 감지하여 특정 메서드를 실행
v-model	폼(form)에서 주로 사용되는 속성 폼에 입력한 값을 Vue 인스턴스의 데이터와 즉시 동기화 화면에 입력된 값을 저장하여 서버에 보내거나 watch와 같은 고급 속성을 이용하여 추가 로직을 수행할 수 있음 <input>, <select>, <textarea> 태그에만 사용

Vue 디렉티브(Directive)

- 조건에 따른 렌더링 비교
 - 일반적으로 스타일을 수정하는 쪽보다 DOM을 수정하는 쪽이 렌더링 비용이 더 큼
 - 따라서 평가 값이 빈번하게 바뀌는 경우에는 v-show 사용

```
<p v-if="인자">  
    //참이면 화면에 표시, 거짓이면 표시하지 않음  
    //DOM 요소 추가 및 제거  
</p>
```

```
<p v-show="인자">  
    //스타일에 display 프로퍼티 값 변경하는 방식으로 동작  
</p>
```

이벤트 처리

- Vue는 데이터에 대한 반응성을 지원하기 때문에 바인딩 된 데이터가 변화가 있으면 즉각 화면에 갱신
- Vue에서는 데이터의 변화에 따른 기능을 수행할 수 있는 속성 보유

고급 템플릿 기법

- 실제 개발 시 유용한 속성을 사용하는 기법
- Vue는 데이터에 대한 반응성을 지원하기 때문에 바인딩 된 데이터가 변화가 있으면 즉각 화면에 갱신
- 데이터의 변화에 따른 기능을 수행할 수 있는 속성 제공
- 데이터 바인딩, 디렉티브 문법과 함께 사용
- 연관 속성

속성	설 명
computed	data 속성값의 변화에 따라 자동으로 정의된 function 을 다시 수행 v-on:change 나 @change를 이용하여 element의 값이 바뀐 것에 대한 기능을 수행할 수 있음 computed는 function을 여러 곳에 사용하게 되면 캐싱 기능이 있어 한번 수행한 결과를 다른 곳에도 출력만 함 따라서 복잡한 연산에는 computed가 적합
watch	computed 속성과 유사하게 특정 data 변화에 따라 수행 단, 데이터 호출과 같이 시간이 상대적으로 많이 소요되는 비동기 처리에 적합함

고급 템플릿 기법

- computed 속성 : methods 속성의 차이점

computed

데이터 연산들을 정의하는 영역

[장점]

- data 값의 변화에 따라 자동으로 다시 연산 수행
- 데이터가 변경되지 않는 한 이전의 계산 값 보유(캐싱)하고 있다가 필요할 때 바로 반환 해 줌(동일한 연산을 반복해서 하지 않기 위함)

복잡한 연산을 반복 수행해서 화면에 나타내야 한다면 computed 속성 이용 권장

methods

호출할 때만 해당 로직이 수행
수행할 때마다 연산을 하기 때문에 캐싱 사용 안함

고급 템플릿 기법

- watch 속성
 - computed 속성과 유사하게 특정 data 변화에 따라 수행
 - computed 속성과 다르게 data를 변경하기 전까지 function이 수행되지 않음
 - 주로 input의 값을 변경하게 되면 지정된 function을 수행하여 데이터 값 변경
 - watch 속성은 감시할 데이터를 지정하고 그 데이터가 바뀌면 이런 함수를 실행하라는 방식의 소프트웨어 공학에서 이야기하는 '명령형 Programming' 방식

| Vue 프로젝트 구성

Vue 프로젝트 구성

1	싱글 파일 컴포넌트 체계(Single File Components)
2	Vue CLI
3	WebPack 이해하기
4	Vue 프로젝트 생성 및 실행하기

싱글 파일 컴포넌트 체계(Single File Components)

- 싱글 파일 컴포넌트 체계(Single File Components)란?
 - .vue 파일로 프로젝트 구조를 구성하는 방식 의미
 - .vue 파일 1개는 "Vue 애플리케이션"을 구성하는 한 개의 컴포넌트와 동일
- 구조
 - <template> 태그 안에는 HTML 태그와 Vue 데이터 바인딩 값들을 넣고, <script> 태그에는 Vue 컴포넌트에서 사용할 속성들을 정의
 - <style>태그에는 HTML 태그의 스타일 속성들을 정의

싱글 파일 컴포넌트 체계(Single File Components)

- .vue 기본 구조

```
<template>
  <!--
    HTML tag 내용으로 화면에 표시할 요소들을 정의하는 영역
    HTML과 뷰 데이터 바인딩 값들 표현
  -->
</template>

<script>
  export default {
    //자바스크립트 내용으로 뷰 컴포넌트의 내용을 정의하는 영역
  }
</script>

<style>
  /* CSS 스타일 내용으로 템플릿에 추가한 HTML 태그의 CSS 스타일로 정의하는 영역 */
</style>
```

export default{}
ES6 의 자바스크립트 모듈화와 관련된 문법

Vue CLI

- 싱글 파일 컴포넌트 체계를 사용하기 위해서는 .vue 파일을 웹 브라우저가 인식할 수 있는 형태의 파일로 변환해 주기 위한 도구 필요

도구	설 명
웹팩(Webpack)	웹 앱의 자원(HTML, CSS, 이미지등)들을 자바스크립트 모듈로 변환해 하나로 묶어 웹 성능을 향상시켜주는 자바스크립트 모듈 번들
브라우저파일(Browserify)	웹팩의 기능과 흡사하나 웹자원 압축이나 빌드 자동화등의 기능은 없음

- Vue 개발팀의 노력
 - 별도의 도구등에 대한 학습 시간을 단축 시켜주고자 Vue 코어 팀에서 CLI(Command Line Interface) 도구 제공

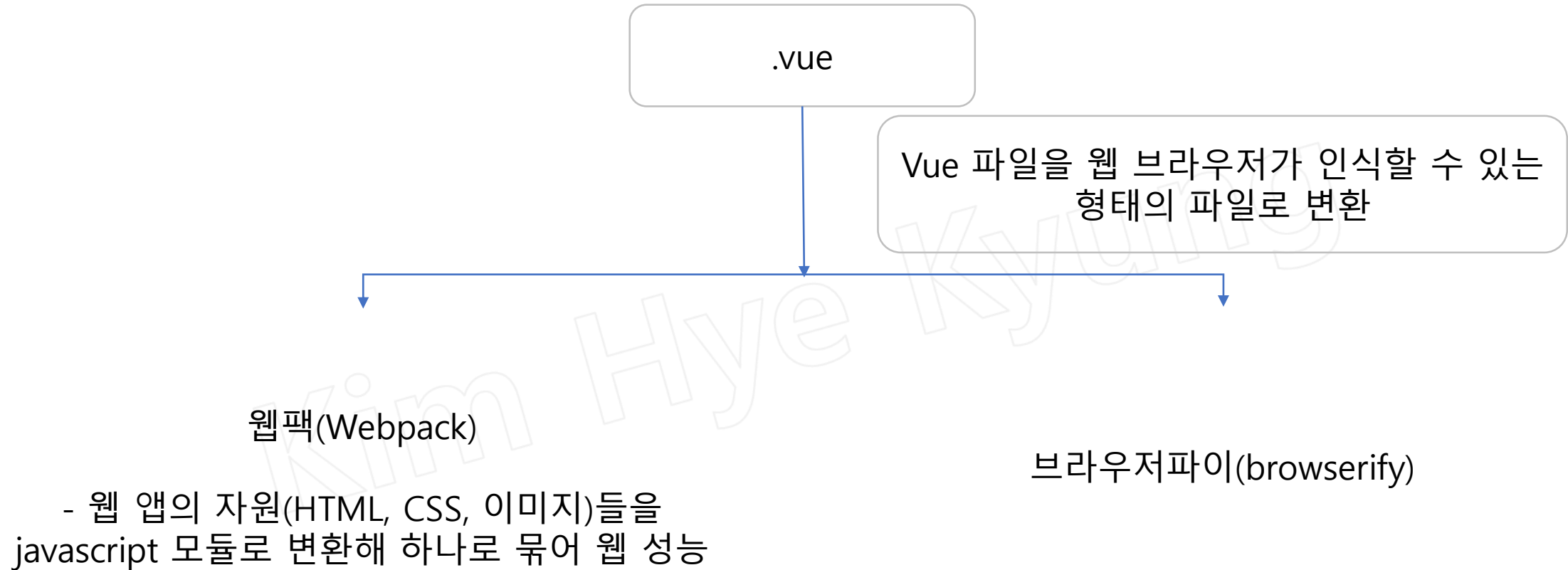
Vue CLI

- Command line Interface
- Vue 개발자들이 편하게 프로젝트를 구성할 수 있도록 Vue core 팀에서 제공하는 도구
- 커맨드 창에서 명령어로 특정 동작을 수행 할 수 있는 도구
- 설치 조건
 - Node.js가 설치되어 있어야 함
- 설치 명령어 및 확인 명령어

```
>npm install vue-cli -global
```

```
>vue
```

Vue CLI



Vue CLI 설치

```
C:\Users\WKimhyekyung>npm install vue-cli -global
npm WARN deprecated coffee-script@1.12.7: CoffeeScript on NPM has moved to "coffeescript" (no hyphen)
C:\Users\WKimhyekyung\AppData\Roaming\npm\vue -> C:\Users\WKimhyekyung\AppData\Roaming\npm\node_modules\vue-cli\bin\vue
C:\Users\WKimhyekyung\AppData\Roaming\npm\vue-list -> C:\Users\WKimhyekyung\AppData\Roaming\npm\node_modules\vue-cli\bin\vue-list
C:\Users\WKimhyekyung\AppData\Roaming\npm\vue-init -> C:\Users\WKimhyekyung\AppData\Roaming\npm\node_modules\vue-cli\bin\vue-init
+ vue-cli@2.9.6
added 239 packages from 206 contributors in 20.861s

C:\Users\WKimhyekyung>vue
Usage: vue <command> [options]

Options:
  -V, --version  output the version number
  -h, --help     output usage information

Commands:
  init          generate a new project from a template
  list          list available official templates
  build         prototype a new project
  create        (for v3 warning only)
  help [cmd]    display help for [cmd]
```

```
C:\Users\WKimhyekyung>vue -V
2.9.6
```

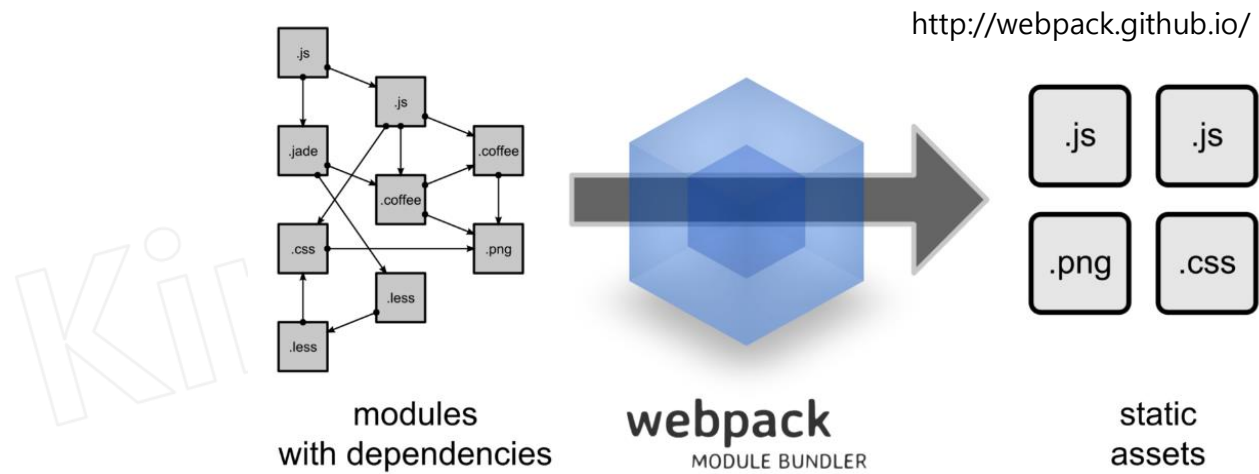

Vue CLI 명령어

- 프로젝트 초기 구성을 위한 명령어

템플릿 종류	설 명
vue init simple	최소 Vue 기능만 들어간 HTML 파일 1개 생성
vue init webpack	고급 웹팩 기능을 활용한 프로젝트 구성 방식 테스팅, 문법 검사 등을 지원
vue init webpack-simple	웹팩 최소 기능을 활용한 프로젝트 구성방식 빠른 화면 프로토타이핑용

Webpack 이해하기

- 웹팩이란?
 - 최신 프론트엔드 프레임워크들이 원하는 모듈 번들러
 - 모듈 번들러란?



서로 연관이 있는 모듈 간의 관계를 해석하여 정적인 자원으로 변환해 주는 변환 도구
- 파일 간의 연관 관계를 파악하여 하나의 자바스크립트 파일로 변환해 주는 변환 도구

Webpack 이해하기

- 주 용도
 - 애플리케이션 동작과 관련된 여러 개의 파일(HTML, CSS, JavaScript, 이미지등)들을 1개의 자바스크립트 파일 안에 다 넣어 버리고, 해당 자바스크립트 파일만 로딩해도 웹 앱이 구동되게 하는 취지
- 효과
 - web page의 로딩 속도 향상

Vue와 WebPack

- Vue CLI로 프로젝트 생성할 경우
 - WebPack 사용
 - WebPack dev server 사용
 - webpack 설정 파일의 변화를 감지하여 빠르게 webpack을 빌드할 수 있도록 지원하는 유틸리티이자 node.js의 서버
 - webpack 설정 파일의 내용이 변경되면 브라우저 화면을 자동으로 새로 고침 -> 바로 다시 webpack으로 빌드
 - 즉 webpack으로 화면을 빠르게 제작하고자 할 때 유용하게 사용

Vue와 WebPack

- Vue CLI 기반의 Project 생성 및 실행 명령어로 WebPack 이해하기

vue init webpack-simple

```
? Generate project in current directory? Yes
? Project name webpack_test
? Project description A Vue.js project
? Author ITKim <topickim@gmail.com>
? License MIT
? Use sass? No

vue-cli · Generated "ex03_webpackTest".

To get started:

  npm install
  npm run dev
```

npm install

```
npm WARN browserslist@1.7.7: Browserslist 2 could fail on reading Browsersli
st >3.0 config used in other tools.
> core-js@2.6.9 postinstall D:\01.lecture\13.Vue\02.vueCLI\ex03_webpackTest\node_module
s\core-js
> node scripts/postinstall || echo "ignore"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling Jav
aScript standard library!

The project needs your help! Please consider supporting of core-js on Open Collective o
r Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -
)

> uglifyjs-webpack-plugin@0.4.6 postinstall D:\01.lecture\13.Vue\02.vueCLI\ex03_webpack
Test\node_modules\uglifyjs-webpack-plugin
> node lib/post_install.js

npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\fsevents):
npm WARN SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9:
wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

added 782 packages from 567 contributors and audited 9787 packages in 28.81s
found 8 vulnerabilities (4 moderate, 4 high)
run 'npm audit fix' to fix them, or 'npm audit' for details
```

npm run dev

```
> webpack_test@1.0.0 dev D:\01.lecture\13.Vue\02.vueCLI\ex03_webpackTest
> cross-env NODE_ENV=development webpack-dev-server --open --hot

Project is running at http://localhost:8080/
webpack output is served from /dist/
404s will fallback to /index.html
{ parser: "babylon" } is deprecated: we now treat it as { parser: "babel" }.
-
```

- node_modules
- src
- .babelrc
- .editorconfig
- .gitignore
- index.html
- package.json
- package-lock.json
- README.md
- webpack.config.js

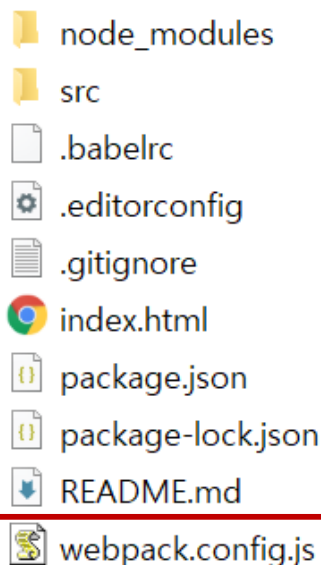
dist 폴더는 안 보임

명령어 실행창에 dist라는 이름의 폴더에 웹팩 결과물로 웹 앱을 로딩 한다는 의미

vue에선 "npm run dev"라는 명령어 사용
- dist라는 폴더는 존재하지 않음
즉 메모리 상에 존재(인 메모리 방식)
파일 시스템에 파일을 쓰고 읽는 시간보다 메모리에 저장하고 읽는 시간이 더 빠르기 때문

Vue와 WebPack

- Webpack-simple Project의 webpack 설정 파일 분석



webpack.config.js에 정의된 설정에 따라 .vue 파일을 포함한 기타 파일들이 webpack으로 빌드

```
1 var path = require('path')
2 var webpack = require('webpack')
3
4 module.exports = {
5   entry: './src/main.js',
6   output: {
7     path: path.resolve(__dirname, './dist'),
8     publicPath: '/dist/',
9     filename: 'build.js'
10  },
11  module: {
12    rules: [
13      {
14        test: /\.css$/,
15        use: [
16          'vue-style-loader',
17          'css-loader'
18        ],
19      }, {
20        test: /\.vue$/,
21        loader: 'vue-loader',
22        options: {
23          loaders: {
24          }
25          // other vue-loader options go here
26        }
27      },
28      {
29        test: /\.js$/,
30        loader: 'babel-loader',
31        exclude: /node_modules/
32      },
33    ]
34  },
35  resolve: {
36    alias: {
37      'vue$': 'vue/dist/vue.esm.js'
38    },
39    extensions: ['*', '.js', '.vue', '.json']
40  },
41  devServer: {
42    historyApiFallback: true,
43    noInfo: true,
44    overlay: true
45  },
46  performance: {
47    hints: false
48  },
49  devtool: '#eval-source-map'
50 }
51
52 if (process.env.NODE_ENV === 'production') {
53   module.exports.devtool = '#source-map'
54   // http://vue-loader.vuejs.org/en/workflow/production.html
55   module.exports.plugins = (module.exports.plugins || []).concat([
56     new webpack.DefinePlugin({
57       'process.env': {
58         NODE_ENV: '"production"'
59       }
60     }),
61     new webpack.optimize.UglifyJsPlugin({
62       sourceMap: true,
63       compress: {
64         warnings: false
65       }
66     }),
67     new webpack.LoaderOptionsPlugin({
68       minimize: true
69     })
70   ])
71 }
```

```
33 {
34   test: /\. (png|jpg|gif|svg)$/,
35   loader: 'file-loader',
36   options: {
37     name: '[name].[ext]?[hash]'
38   }
39 }
40 ]
41 },
42 resolve: {
43   alias: {
44     'vue$': 'vue/dist/vue.esm.js'
45   },
46   extensions: ['*', '.js', '.vue', '.json']
47 },
48 devServer: {
49   historyApiFallback: true,
50   noInfo: true,
51   overlay: true
52 },
53 performance: {
54   hints: false
55 },
56 devtool: '#eval-source-map'
57 }
58
59 if (process.env.NODE_ENV === 'production') {
60   module.exports.devtool = '#source-map'
61   // http://vue-loader.vuejs.org/en/workflow/production.html
62   module.exports.plugins = (module.exports.plugins || []).concat([
63     new webpack.DefinePlugin({
64       'process.env': {
65         NODE_ENV: '"production"'
66       }
67     }),
68     new webpack.optimize.UglifyJsPlugin({
69       sourceMap: true,
70       compress: {
71         warnings: false
72       }
73     }),
74     new webpack.LoaderOptionsPlugin({
75       minimize: true
76     })
77   ])
78 }
```

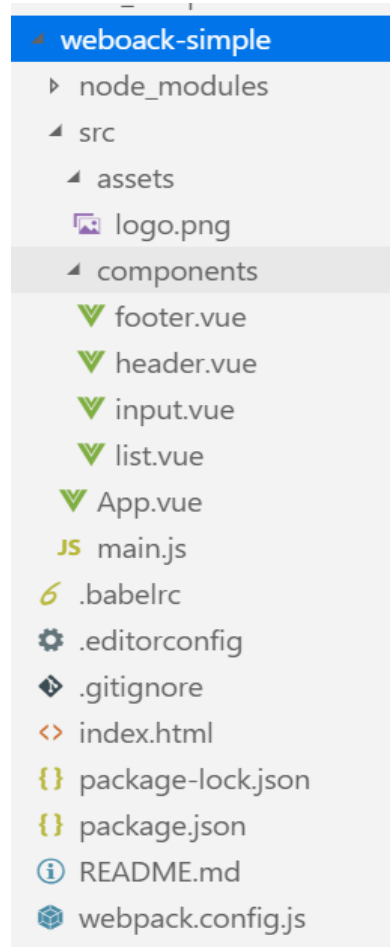
Vue와 WebPack

- WebPack의 주요 속성 5가지

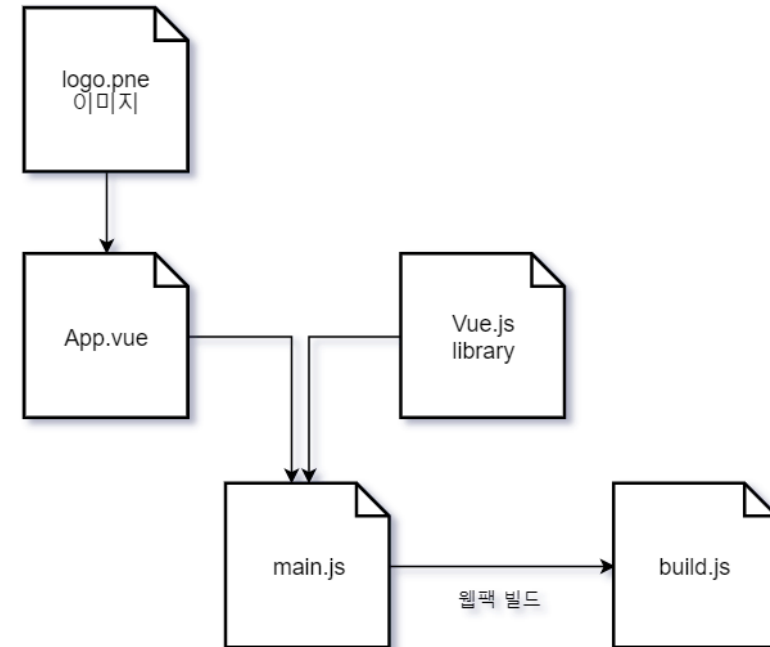
속성	설 명
entry	웹팩으로 빌드(변환)할 대상 파일을 지정하는 속성 entry로 지정한 파일의 내용에는 전체 애플리케이션 로직과 필요 library를 로딩하는 로직 설정
output	웹팩으로 빌드한 결과물의 위치와 파일 이름등 세부 옵션을 설정하는 속성
loader	웹팩으로 빌드할 때 HTML, CSS, 이미지등의 파일들을 자바스크립트로 변환하기 위해 필요한 설정을 정의하는 속성
plugin	웹팩으로 빌드하고 나온 결과물에 대해 추가 기능을 제공하는 속성 가령 결과물의 사이즈를 줄이거나 결과물(기본적으로 자바스크립트)을 기타 CSS, HTML 파일로 분리하는 기능등 설정
resolve	웹팩으로 빌드할 때 해당 파일이 어떻게 해석되는지 정의하는 속성 가령, 특정 라이브러리를 로딩할 때 버전은 어떤 것으로 할지, 파일 경로는 어디로 지정할 지 등을 정의

Vue와 WebPack

- webpack으로 빌드할 때 파일 간의 관계도



npm run build
or
npm run dev



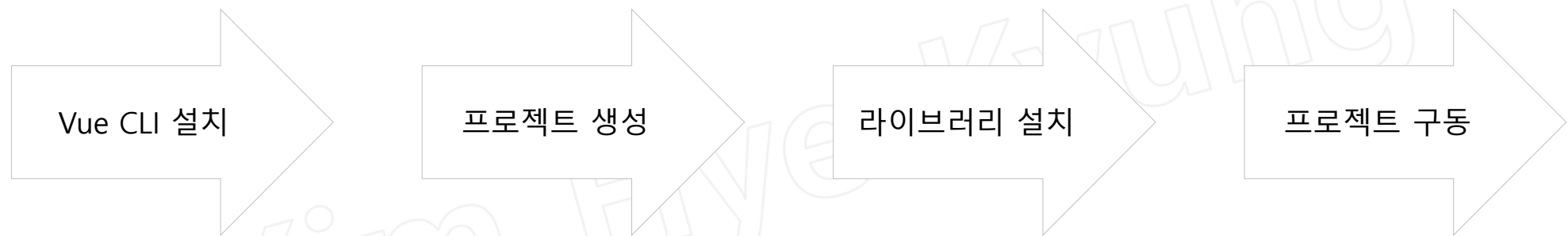
main.js 파일에서 App.vue 파일과 Vue.js library를 불러와서 애플리케이션을 동작시키고, App.vue에서 logo.png를 이용하여 웹 page를 구성

빌드시 파일간의 관계에 따라 build.js 파일 생성

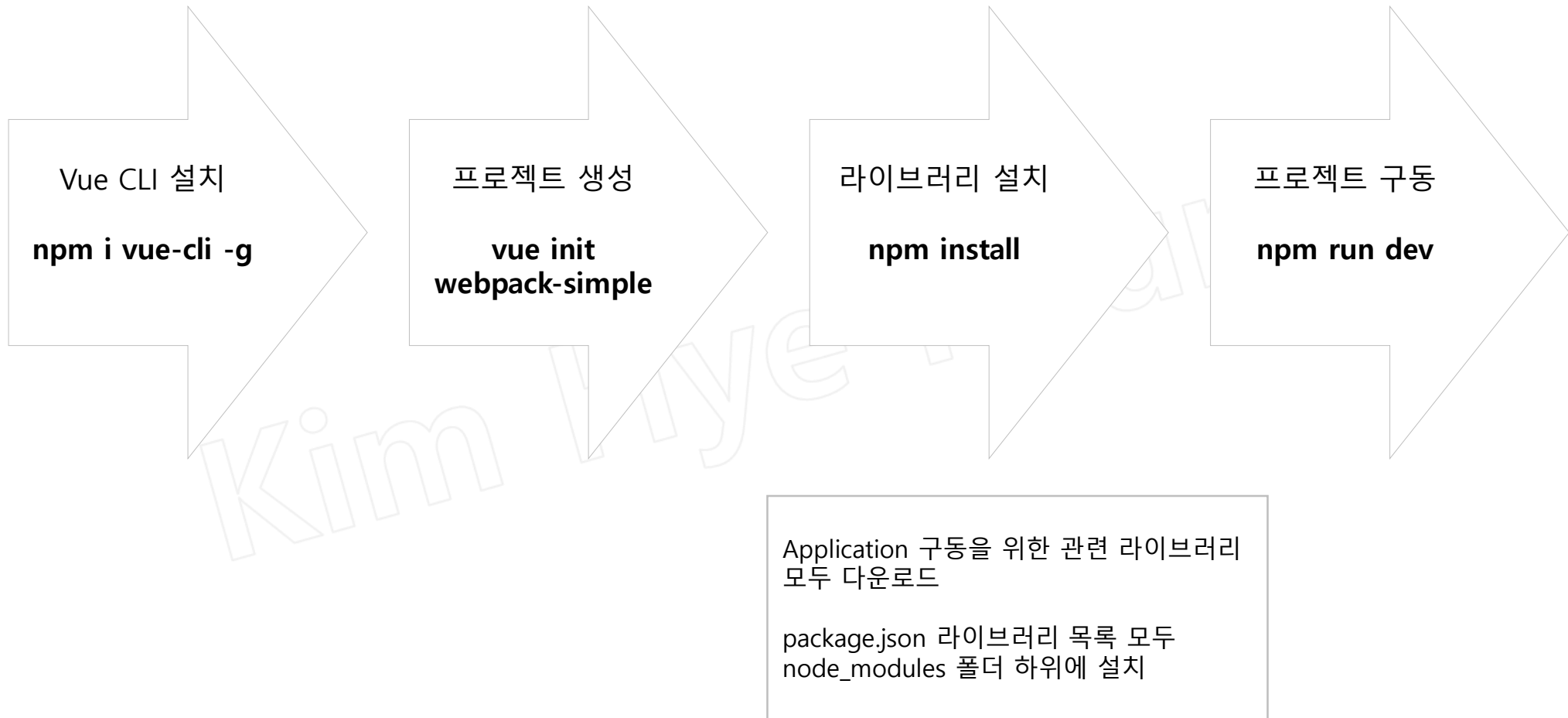
build.js는 애플리케이션 구조에 맞게 파일(모듈)간의 순서에 맞게 설정

결론 : index.html 파일에서 webpack으로 빌드한 build.js 파일만 로딩하면 애플리케이션 로직을 구성하는 vue 파일, png 파일, 자바스크립트 library를 로딩한 것과 동일한 방식으로 동작

Vue CLI로 Project를 구성하는 과정



Vue CLI로 Project를 구성하는 과정



Vue CLI로 Project를 생성 및 실행하기

1단계 : Vue CLI로 프로젝트 생성

2단계 : npm 명령어를 통한 dependency 추가 및 package.json 확인

3단계 : 실행

4단계 : 브라우저에서 실행 확인

Vue CLI로 프로젝트 생성

- 기본 명령어

vue init webpack-simple

:project명에 대문자 사용 불가

npm install

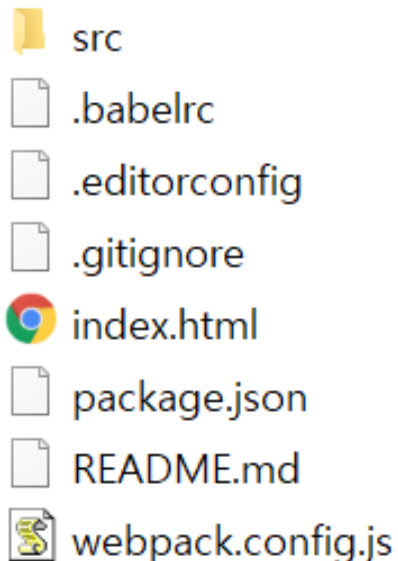
npm run dev

이미 서버가 실행중인 경우 port 번호가
1이 증가된 상태로 새로운 서버 실행

Vue CLI로 Project를 생성 및 실행하기

프로젝트 생성

```
>vue init webpack-simple
```



목록	설 명
1. Project name	프로젝트 명, 프로젝트 생성과 함께 생성되는 package.json 파일의 name 필드값
2. Project description	프로젝트 설명 package.json 파일의 description 필드값
3. Author	애플리케이션 작성자, package.json 파일의 author 필드값, 기본값은 gitconfig의 이름 및 이메일 주소
4. Vue build	Vue Build에 대한 standalone 여부, 전체 빌드 또는 런타임만 사용할 것인지 선택
5. Install vue-router?	vue-router 사용여부. 기본값은 Y
6. User ESLint to lint your code?	ESLint 사용여부 (ESLint란 정적 코드 검증 솔루션) 기본값은 Y
7. Set up unit tests	단위 테스트 모듈 설치 여부. 기본값은 Y
8. Setup e2e tests with Nightwatch?	E2E(End-to-End) 테스트는 전체 시스템이 제대로 작동하는지 확인 하기 위한 테스트
9. Should we run 'npm install' for you after pr oject has been created? (recommanded)	프로젝트 생성 후 npm install 즉 의존 모듈 설치 진행 여부

라이브러리 설치

라이브러리 다운로드

```
>npm install
```

- build
- config
- node_modules
- src
- static
- test
- .babelrc
- .editorconfig
- .gitignore
- .postcssrc.js
- index.html
- package.json
- package-lock.json
- README.md

Application 구동을 위한 관련 라이브러리 모두 다운로드











Axios 다운로드

```
>npm install axios --save
```

Axios는 HTTP 통신을 하기 위한 라이브러리

--save 옵션 : 웹팩으로 프로젝트를 변환했을 때 설치한 다른 환경에서도 재 설치 하지 않고 사용 가능하게 내장되도록 해 주는 옵션

Vue CLI로 프로젝트 구조

 node_modules	npm install 명령어로 다운받은 library들 존재
 src	.vue 파일을 비롯하여 애플리케이션 동작을 위해 필요한 로직이 있는 위치
 .babelrc	vue로 만든 웹 앱의 시작점, npm run dev 실행 시 로딩되는 파일 공통 css, js 적용
 .editorconfig	
 .gitignore	
 index.html	
 package.json	npm 설정 파일, 프로젝트 정보, vue 애플리케이션이 동작하는 데 필요한 library들 정의
 package-lock.json	
 README.md	웹팩 설정 파일. 웹팩 빌드를 위해 필요한 로직들 정의
 webpack.config.js	

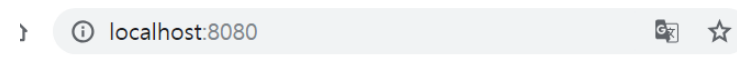
애플리케이션 실행

- 웹팩에서 로컬 서버 실행

```
>npm run dev
```

```
> cli-template@1.0.0 dev D:\W01.lecture\W13.Vue\W01.cliTest
> cross-env NODE_ENV=development webpack-dev-server --open --hot

Project is running at http://localhost:8080/
webpack output is served from /dist/
404s will fallback to /index.html
{ parser: "babylon" } is deprecated; we now treat it as { parser: "babel" }
```



Welcome to Your Vue.js App

Essential Links

[Core Docs](#) [Forum](#) [Community Chat](#) [Twitter](#)

Ecosystem

[vue-router](#) [vuex](#) [vue-loader](#) [awesome-vue](#)

Vue CLI로 프로젝트 생성 예제

```
>vue init webpack vue_cliProject01
```

```
? Project name vue_cli_project01
? Project description A Vue.js project
? Author ITKim <topickim@gmail.com>
? Vue build standalone
? Install vue-router? Yes
? Use ESLint to lint your code? No
? Set up unit tests Yes
? Pick a test runner noTest
? Setup e2e tests with Nightwatch? No
? Should we run `npm install` for you after the project has been created? (recomm

vue-cli · Generated "vue_cli_project01".

# Installing project dependencies ...
# =====

npm WARN deprecated extract-text-webpack-plugin@3.0.2: Deprecated. Please use htt
npm WARN deprecated browserslist@2.11.3: Browserslist 2 could fail on reading Bro
npm WARN deprecated bfj-node4@5.3.1: Switch to the `bfj` package for fixes and ne
npm WARN deprecated browserslist@1.7.7: Browserslist 2 could fail on reading Brow

> core-js@2.6.9 postinstall D:\W01.lecture\13.Vue\02.vueCLI\vue_cli_project01\node
> node scripts/postinstall || echo "ignore"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilli
The project needs your help! Please consider supporting of core-js on Open Collec
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock

Also, the author of core-js ( https://github.com/zloirock ) is looking for a good
```

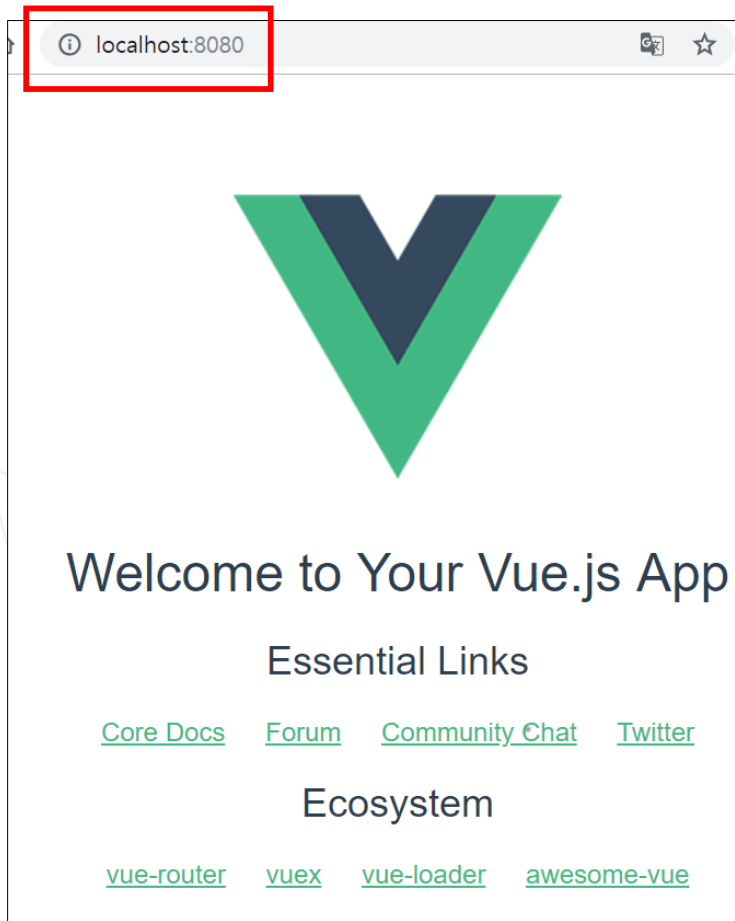
Vue Loader

- 웹팩에서 지원하는 라이브러리
- 싱글 파일 컴포넌트 체계에서 사용하는 .vue 파일의 내용을 브라우저에서 실행 가능한 웹 페이지의 형태로 변환해 줌



Vue Loader

- npm run dev 실행 후
확인하기
- App.vue 파일 내용



```
<template>
  <div id="app">
    
    <router-view/>
  </div>
</template>

<script>
export default {
  name: 'App'
}
</script>

<style>
#app {
  font-family: 'Avenir', Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

Vue Loader

Project 생성 및 실행

웹팩의 Vue Loader가 변환

App.vue 실행

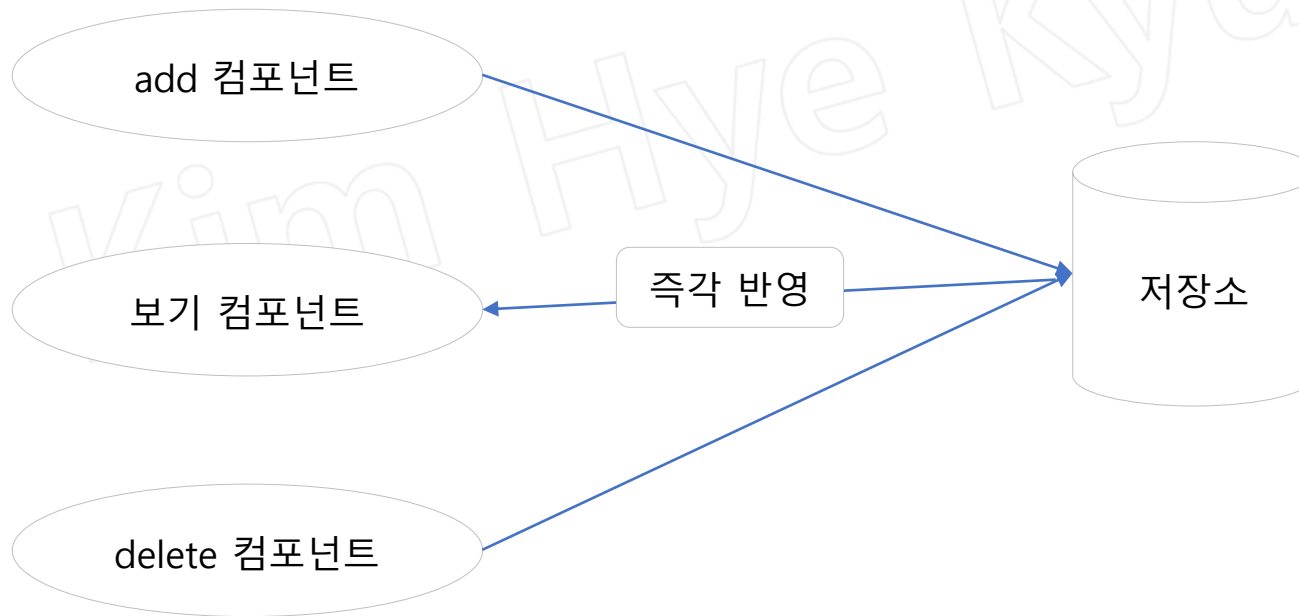
- 웹 페이지에 표시
- `<template>`, `<script>`, `<style>` 의 내용이 각 html, javascript, css 코드로 변환

참고 : 웹팩은 javascript 모듈만 인식

따라서 Vue Loader는 .vue 파일을 java script 모듈로 변환

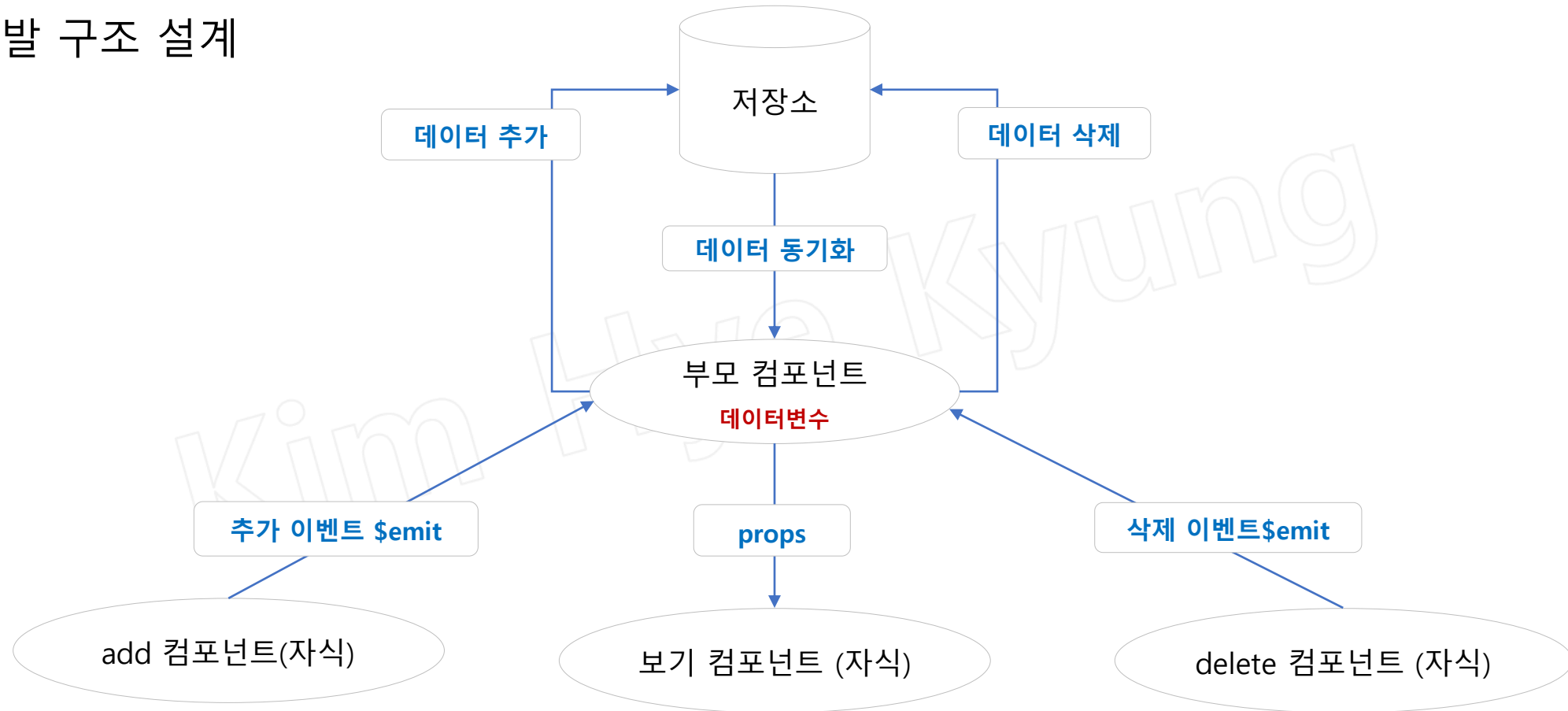
Vue Project 실습

- 조건
 - 컴포넌트를 기능별로 분리해서 개발해 본다
 - 컴포넌트간 데이터가 공유되어야 하며 동기화 되어야 한다



Vue Project 실습

- 개발 구조 설계



API 소개

No	함수명	설명
	\$mount()	El 속성과 동일하게 인스턴스를 화면에 붙이는 역할 인스턴스 생성시에 el 속성없이도 생성 후에 \$mount() 이용하면 강제로 인스턴스를 화면에 붙일 수 있음 Vue 라우터의 공식 문서에서도 주로 사용 하는 방식