

09월 07일 (수)

1장. Vue.js 개요

1-1. VUE.js 특징

SSR(Server Side Rendering)이란?? [*시험]

- 전통적인 렌더링 방식
- HTML 문서를 서버에서 생성해 클라이언트로 전송 → 첫 화면 생성이 빠르다
- 검색 엔진 최적화(SEO) 쉬움
- 화면 변경사항 발생 시
 - 서버로 다시 렌더링 요청 → 서버 부담 증가
 - 화면 전환 시 새로 고침 발생

CSR(client Side Rendering)이란??

- 최근 등장한 렌더링 방식
- **HTML** 문서를 클라이언트에서 생성 → 첫 화면 생성 느림
- 검색 엔진 최적화(SEO) 어려움
- 화면 변경사항 발생 시
 - 서버에 요청하지 않고 변화가 존재하는 부분만 변경 → 서버 부담 감소
 - 새로 고침 발생 안함(부드러운 화면 전환)
- **CSR** 은 **Vue.js** 방식

Vue.js : 순수 **Javascript** 를 좀 더 편하고 직관적으로 바꾼 것

Vue , **React** 등을 사용하게 되면 Virtual DOM 방식으로 DOM

DOM이란??

HTML의 요소

Virtual DOM[*시험]

모든 DOM 제어 이후, 최종적으로 한 번 렌더링을 수행하는 방법

Virtual DOM 특징??

- 모든 DOM 제어가 끝나면 최종적으로 단 한번만 렌더링 → **성능 향상**
- Virtual DOM 채택 시 , Real DOM에 직접 접근 금지

Virtual DOM의 필요성??

성능때문에 씀

일반적인 DOM 변경 발생 시 Render Tree 부터 재 시작이 이루어짐

- Render Tree 과정 : 모든 DOM에 style 정보를 다시 입히는 과정
- DOM 제어를 할 때마다 Client Browser 의 속도 저하를 발생함
- 10개의 DOM을 순차적으로 변경할 경우, 총 10회 Render Tree 를 추가 생성함

Virtual DOM 동작 방식??

속도 개선을 Virtual DOM Tree 동작 과정

1. 진짜 DOM Tree 기반으로 Virtual DOM Tree 를 생성
2. DOM 제어를 Virtual DOM Tree에 모두 적용
3. 모든 DOM 제어를 끝났을 때, Virtual DOM Tree를 진짜 DOM Tree에 적용
4. 렌더링을 시작함

[이자룡 강사]

서버에서 만들 때 장점은
프로그래밍 언어의 영역이므로

반복문, 변수 저장 등에 자유로워 짐(HTML에서는 안됨)
서버입장에서는 힘들지만 사용자 입장에선 더 빠른 방법임

SSR은 HTML에서 코드를 쓸 수 있음
하지만 웹에서는 알아들을 수 없음
웹은 js, html, css 만 알아들음

java와 spring 하는 사람들은 js 모름 ㅋㅋ

검색 엔진 최적화(SEO) 쉬움 이란 말은 만약 웹크롤링을 할 때 자바스크립트는 안뜨는데...불라블라

사용자 컴퓨터가 좋아졌고 버틸 수 있는 능력이 있기 때문에 서버를 힘들게 안해도 됨
또 다른 이유는 모바일 사용자가 많아져서

1-2. MVVM 패턴

디자인 패턴이란??

MVVM pattern??

Vue.js 특징??

컴포넌트 기반의 조립식 웹 UI 개발

- 원래는 HTML 에서 다 했던 것을 head 파일 따로 메인 파일 따로 만드는 방법

쉬운 DOM 제어

- 순수 Javascript 에 비해 조건,반복,데이터 저장, 이벤트 처리가 쉬움

Virtual DOM 방식 사용


- 웹페이지 성능 향상

1-3 Vue.js 설치

수업에선 2버전 3(버전은 과도기)

vue CDN by jsDelivr - A CDN for npm and GitHub

Supports npm, GitHub, WordPress, Deno, and more. Largest network and best performance among all CDNs. Serving more than 80 billion requests per month. Built for production use.

 <https://www.jsdelivr.com/package/npm/vue?version=2.7.10>

JSDEL

Hello Vue.js

new Vue

`el` : #app 으로 지정된 div를 Vue 형식으로 만듦.

`data(){}` : Vue instance 에서 변수를 생성할 때 사용하는 객체 , 함수안에 변수를 정의함

데이터 바인딩 : `{{message}}` 처럼 `{{}}` 안에 넣는 것

```
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">{{message}}</div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
    <script>
      const app = new Vue({
        el: "#app",
        data() {
          return {
            message: "Hello Vue",
          };
        },
      });
    </script>
  </body>
</html>
```

`<!--{{message}}` 라고 나오다가 밑으로 가면서 `message : "Hello Vue"를 만나`
`{{message}} 를 "Hello Vue" 로 바꿈 -->`

2장 . Vue.js 지시자

V-On

이벤트 핸들러 지정할 때 사용

`v-on:click = "bbq"` : 클릭했을 때 bbq 메서드를 실행

`@click = "bbq"` : 축약형으로 사용 가능

클릭 이벤트와 전역 변수

클릭 이벤트

`v-on:click` : bbq 콜백 함수 등록

`this.message` : data에 있는 변수들은 전역 변수 this를 사용하여 접근 가능

`methods` : Vue instance 에서 메서드를 생성할 때 사용되는 객체

message에 값을 넣으면 즉시 반영 됨

```
<!-- hello_vue.js.html -->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <h1>{{message}}
    </h1>
    <button v-on:click="bbq">hey</button>
    </div>

    <!--// vue.js 2.7.10 링크:
    //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
```

```

<script>
  const app = new Vue({
    el: "#app",
    data() {
      return {
        message: "Hello Vue",
        nextMessage: "클릭함 ㅎㅎ",
      };
    },
    methods: {
      bbq() {
        this.message = this.nextMessage;
      },
    },
  });
</script>
</body>
</html>

```

Hello Vue

hey

클릭함 ㅎㅎ

hey

순수 JavaScript 코드와 비교

```

<h1>클릭 전</h1>
<button>hey</button>

const h1 = document.querySelector("h1");
const btn = document.querySelector("button");
btn.addEventListener("click", () => {
  h1.textContent = "클릭함 ㅎㅎ";
});

```

단점

1. 이벤트와 DOM에 대한 전문성 필요

- 이 경우, `textContent` 속성을 변경해야 한다는 것을 정확히 알고있어야 함
- 만약 다른 속성을 변경해야 한다면 추가적인 학습 필요

2. Vue.js 코드에 비해 가독성 떨어짐

- 화면에서 변경 될 데이터, 클릭 시 작동할 이벤트 핸들러 한눈에 파악 어려움
- 변수와 함수가 많아질수록 생산성 떨어짐

3. 매번 DOM 제어가 이루어지기에 성능 하락

- Vue.js 는 Virtual DOM 사용하고, 모든 DOM 제어 이후 단 한번만 렌더링

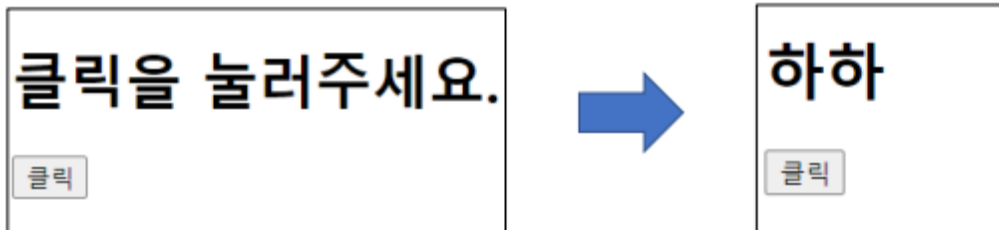
[도전] 버튼 클릭 시 글씨 변경

```

<!-- [도전]버튼 클릭 시 글씨 변경.html -->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <h1>{{message}}</h1>
      <button v-on:click="bbq">클릭</button>
    </div>

    <!--// vue.js 2.7.10 링크:
    //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
    <script>
      const app = new Vue({
        el: "#app",
        data() {
          return {
            message: "클릭을 눌러주세요",
            nextMessage: "하하",
          };
        },
        methods: {
          bbq() {
            this.message = this.nextMessage;
          },
        },
      });
    </script>
  
```

```
</body>
</html>
```



데이터 바인딩

데이터 바인딩이란??

data 객체 내의 변수와 화면의 데이터를 연동하는 것을 의미함

바인딩의 종류[*시험]

1. 단방향 바인딩 : `v-bind`, `{}`
2. 양방향 바인딩 : `v-model`

{{변수명}}

Vue instance 의 값을 수정하면 즉시 `{{message}}` 에 값이 반영됨

단방향 Binding : data 객체의 해당되는 변수와 연결됨

단방향이란, 데이터가 한쪽 방향으로만 묶여있다는 뜻임. data 객체 안에 message가 바뀔 수는 있지만 사용자가 화면에서 message 자체를 변경할 순 없다.

v-bind : 단방향 바인딩

기본형 : <a v-bind:href="url1"> 구글
축약형 : <a : href="url2"> 네이버

```
<!-- v-bind.html -->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <a v-bind:href="url1"> 구글</a>
      <a : href = "url2">네이버</a>
    </div>

    <!--// vue.js 2.7.10 링크:
    //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
    <script>
      const app = new Vue({
        el: "#app",
        data() {
          return {
            url1 : "https://google.com",
            url2 : "https://naver.com",
          };
        },
      },
    </script>
  </body>
</html>
```

[도전 class 속성 바꾸기]

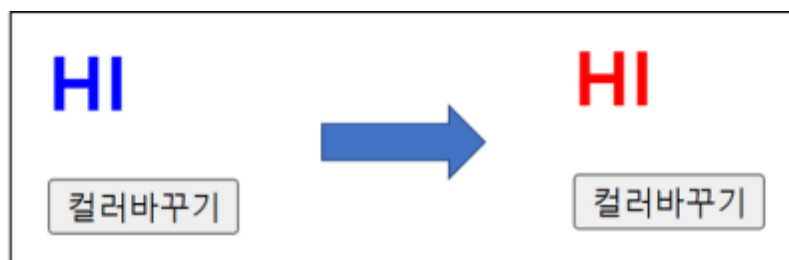
```
<!--도전 class 속성 바꾸기.html-->
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document</title>
```

```

<style>
  .colorRed {
    color: red;
  }
  .colorBlue {
    color: blue;
  }
</style>
</head>
<body>
  <div id="app">
    <h1 v-bind:class ="color">HI</h1>
    <button v-on:click="bbq">컬러바꾸기</button>
  </div>

  <!--// vue.js 2.7.10 링크:
  //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
  <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
  <script>
    const app = new Vue({
      el: "#app",
      data() {
        return {
          color:"colorBlue",
          nextcolor:"colorRed",
        };
      },
      methods: {
        bbq() {
          this.color = this.nextcolor;
        },
      },
    });
  </script>
</body>
</html>

```



순수 Javascript로 구현하면 다른점은??

```

<style>
  .colorBlue {
    color: blue;
  }
  .colorRed {
    color: red;
  }
</style>
<title>Document</title>
<head>
</head>
<body>
  <h1>HI</h1>
  <button>컬러바꾸기</button>
  <script>
    let colorOfHi = "colorBlue";
    const h1 = document.querySelector("h1");
    const btn = document.querySelector("button");
    h1.classList.add(colorOfHi);
    btn.addEventListener("click", () => {
      // 아래 코드는 작동하지 않는다.
      // colorOfHi = "colorRed";

      // 아래 코드는 작동한다.
      let tempColorClass = colorOfHi;
      colorOfHi = "colorRed";
      h1.classList.replace(tempColorClass, colorOfHi);
    });
  </script>

```

V-model

v-model : 양방향 바인딩

input에 쓰임(거의 다 input)

html form 의 값을 바꾸면 , Vue 변수에 값이 변경됨

Vue 변수의 값을 변경하면 html form의 값이 변경됨

<input> 태그에 사용됨 → 사용자의 입력이 data 객체의 속성을 변하게 함

```

<!--v-model.html-->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <input type="text" v-model="message"/>
      <div>{{message}} </div>
    </div>
  </body>
</html>

```

```

</div>

<!--// vue.js 2.7.10 링크:
//https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
<script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
<script>
  const app = new Vue({
    el: "#app",
    data() {
      return {
        message: "",
      }
    }
  });
</script>
</body>
</html>

```

v-bind 와 v-model 비교

```

<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <div><input type="text" v-model="message"/></div>
      <div><input type="text" v-bind:value="message"/></div>
      <input type="text" v-model="message"/>

      <div>{{message}} </div>

    </div>

    <!--// vue.js 2.7.10 링크:
    //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
    <script>
      const app = new Vue({
        el: "#app",
        data() {
          return {

```

```

        message: "글자를 지워보자",
      }
    }
  });
</script>
</body>
</html>

```

v-if

DOM의 출력 여부 결정

`v-if` = "Boolean 값"

`true` 인 경우 출력 o

`false` 인 경우 출력 x

v-else 와 v-else-if 와 함께 사용 가능

v-for

화면에 DOM 요소를 반복적으로 출력할 때 사용

배열 각각의 요소, 인덱스도 받을 수 있음

키를 지정하지 않으면 warning 발생

```

<!-- v-for(todo).html-->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <li v-for="(todo,index) in todos" v-bind:key = "index">

```

```

    {{todo}}
  </li>

</div>

<!--// vue.js 2.7.10 링크:
//https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
<script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
<script>
  const app = new Vue({
    el: "#app",
    data() {
      return {
        todos : ["밥먹기", "잠자기", "코딩하기"]
      }
    }
  });
</script>
</body>
</html>

```

```

<!--[도전]bucket list 만들기.html-->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <li v-for="(bucket,index) in buckets" v-bind:key = "bucket.id">
        {{bucket.text}}
      </li>

    </div>

    <!--// vue.js 2.7.10 링크:
    //https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
    <script>
      const app = new Vue({
        el: "#app",
        data() {

```

```

        return {
          buckets : [
            {id:1,text: "치킨먹기", done: true},
            {id:2,text: "마라탕먹기", done: true},
            {id:3,text: "놀러가기", done: false},
            {id:4,text: "코딩하기", done: false},
            {id:5,text: "티비보기", done: true},
          ]
        }
      }
    });
  </script>
</body>
</html>

```

3장. 라이프 사이클

라이플 사이클[*시험]

`created()` :

`mounted()` :

`created`

[실습] create.html

```

<!-- [실습] create.html-->
<!DOCTYPE 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" />
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <div>{{todo.id}}</div>
      <div>{{todo.title}}</div>
      <div>{{todo.completed}}</div>
    </div>
  </body>
</html>

```

```

<!--// vue.js 2.7.10 링크:
//https://www.jsdelivr.com/package/npm/vue?version=2.7.10-->
<script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/axios@0.27.2/dist/axios.min.js"></script>

<script>
  const app = new Vue({
    el : "#app",
    data()
    {
      return {
        todo:{}
      }
    },
    methods:{

    },
    async created(){
      const result = await axios.get("https://jsonplaceholder.typicode.com/todos/1");
      console.log(result);
      this.todo = result.data;
    }
  })
</script>
</body>
</html>

```

```

<body>
  <div id="app">
    {{todo}}
    <div>{{todo.id}}</div>
    <div>{{todo.title}}</div>
    <div>{{todo.completed}}</div>
  </div>

  <script src="https://cdn.jsdelivr.net/npm/vue@2.7.8/dist/vue.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
  <script>
    const app = new Vue({
      el: "#app",
      data() {
        return {
          todo: {}
        }
      },
      methods: {

      },
      async created() {
        const result = await axios.get("https://jsonplaceholder.typicode.com/todos/1");
        console.log(result);
        this.todo = result.data;
      }
    });
  </script>
</body>

```



```

{ "userId": 1, "id": 1, "title": "delectus aut autem", "completed": false }
1
delectus aut autem
false

```

mounted

4장. Vue CLI

Vue CLI 개요

Vue CLI??[*시험]

- Vue.js 개발환경을 설정해주는 도구
- 기본적인 프로젝트 세팅 제공
- 각각의 컴포넌트를 .vue 파일로 분리해 여러 개의 컴포넌트 상종 가능
- 클라이언트에서 node.js 테스트 서버 사용

Vue CLI 설치[*시험]

\$ npm i @vue/cli -g

i는 install 의 약자 , init이랑은 다름

설치 후 npm list - g로 설치 됐는 지 확인

```
C:\Users\multicampus>npm list -g
C:\Users\multicampus\AppData\Roaming\npm
+-- @vue/cli@5.0.8
`-- nodemon@2.0.19
```

```
vue create main
```

Babel??[*시험]

Linter??[*시험]

Router, Vuex??

```
#테스트 서버 작동 코드
npm run serve
```

SPA(single Page Application)??

이자룡 강사님
nodemodule 같은 크기 큰 것들을 쓰는 것만 html, css, javascript 로 변환하여 disk에 넣음??
만약 용량이 115MB 였던 것들이 2~3 MB로 줄어듦

package.json 의 dependencies 와 devdependencies 의 차이
개발할때만 쓰임??

localhost:8080 쓰고 있으면 8081쓰게 됨
만약 저번에 했던 node.js 8080 썼는데 이걸 다시 build 한다면 8081을 쓰게 됨

public의 favicon이 뭘까??웹페이지 icon임 f12눌러서 맨날 오류 뜨는게 이 icon 없어서 뜨는 거임

VScode extension 에서 Vetur 설치해야함
하면 .vue 파일 잘나옴

gitignore 에 node_modules 하면 안올라감

npm run serve 하면 main.js 부터 돌아감

Ape.vue

```
App.vue U X
Learning-Web-Course > 2022년 9월 07일 > main > src > App.vue > style > #app
1  <template>
2    <div id="app">
3      <nav>
4        <router-link to="/">Home</router-link> |
5        <router-link to="/about">About</router-link>
6      </nav>
7      <router-view/>
8    </div>
9  </template>
10
11  <style>
12  #app {
13    font-family: Avenir, Helvetica, Arial, sans-serif;
14    -webkit-font-smoothing: antialiased;
15    -moz-osx-font-smoothing: grayscale;
16    text-align: center;
17    color: #2c3e50;
18  }
19
20  nav {
21    padding: 30px;
22  }
23
24  nav a {
25    font-weight: bold;
26    color: #2c3e50;
27  }
28
29  nav a.router-link-exact-active {
30    color: #42b983;
31  }
32  </style>
33
```

template??[*시험]

HTML 이랑 같음

script??

Javascript 랑 같음

style??

css 랑 같음

```
<template>
  <div id="app">
    <h1>Hello</h1>
  </div>
</template>

<script>
export default {

};
</script>

<style>
</style>
```

컴포넌트

다음과 같이 나눈다고 가정

Header

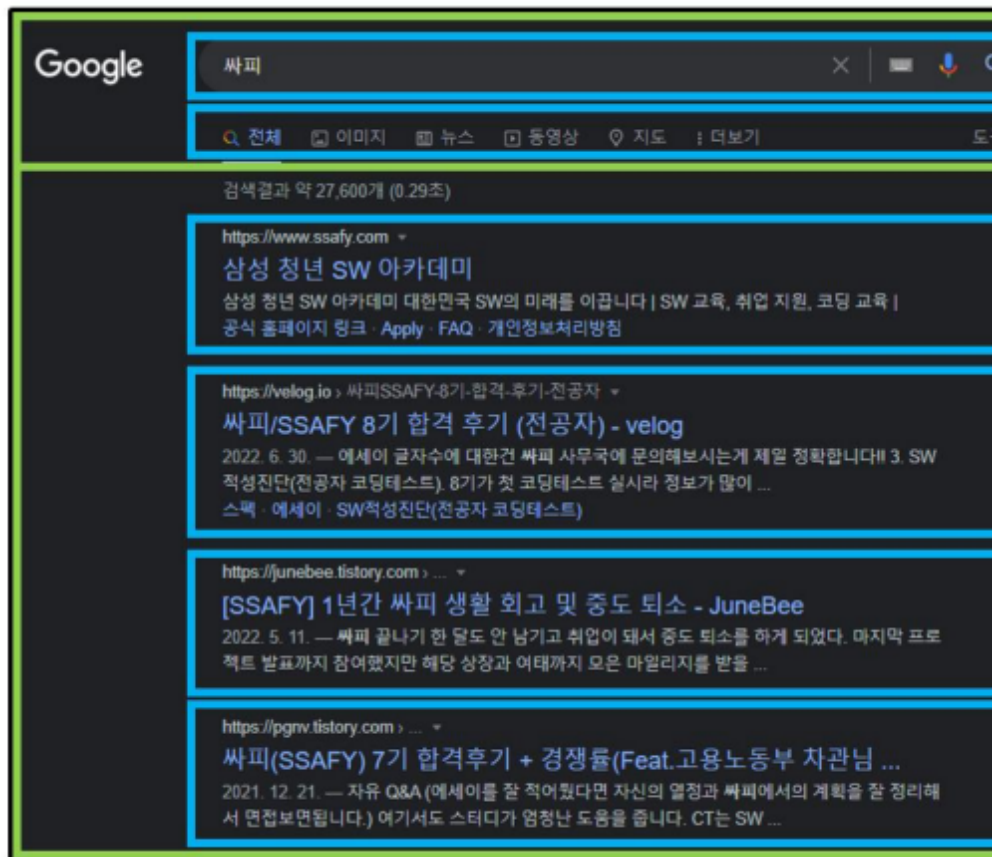
- Search (검색창)
- Navigation (메뉴 바)

Main

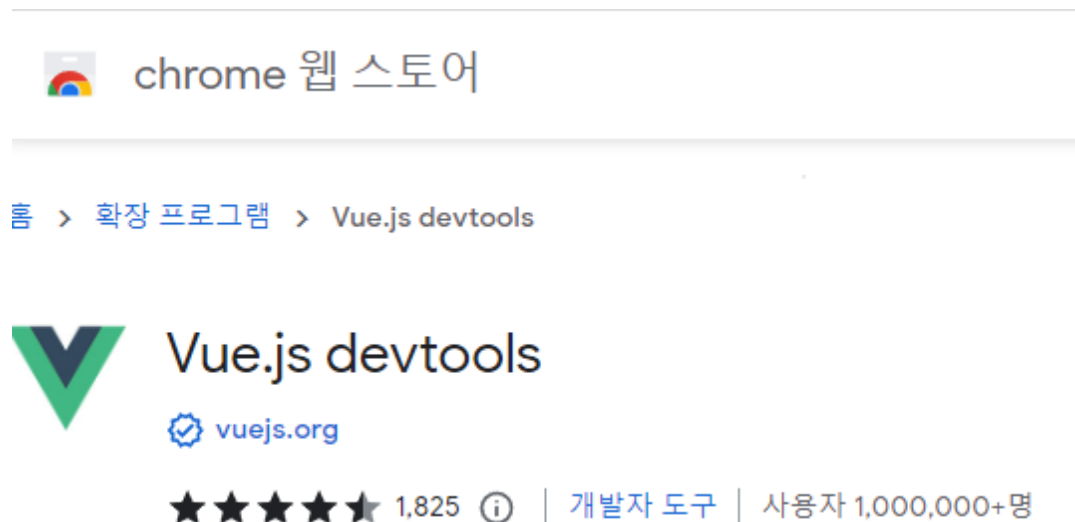
- EachResult (검색 결과)

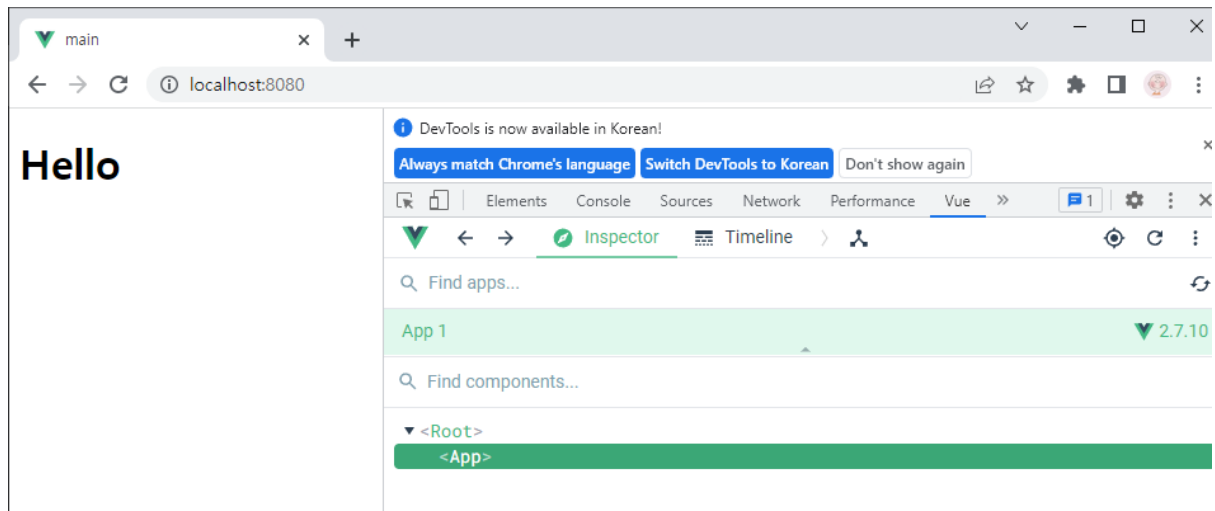
컴포넌트 안에 컴포넌트가 있을 경우, 부모 컴포넌트 , 자식 컴포넌트로 구분

EachResult 는 v-for 를 사용하여 반복함



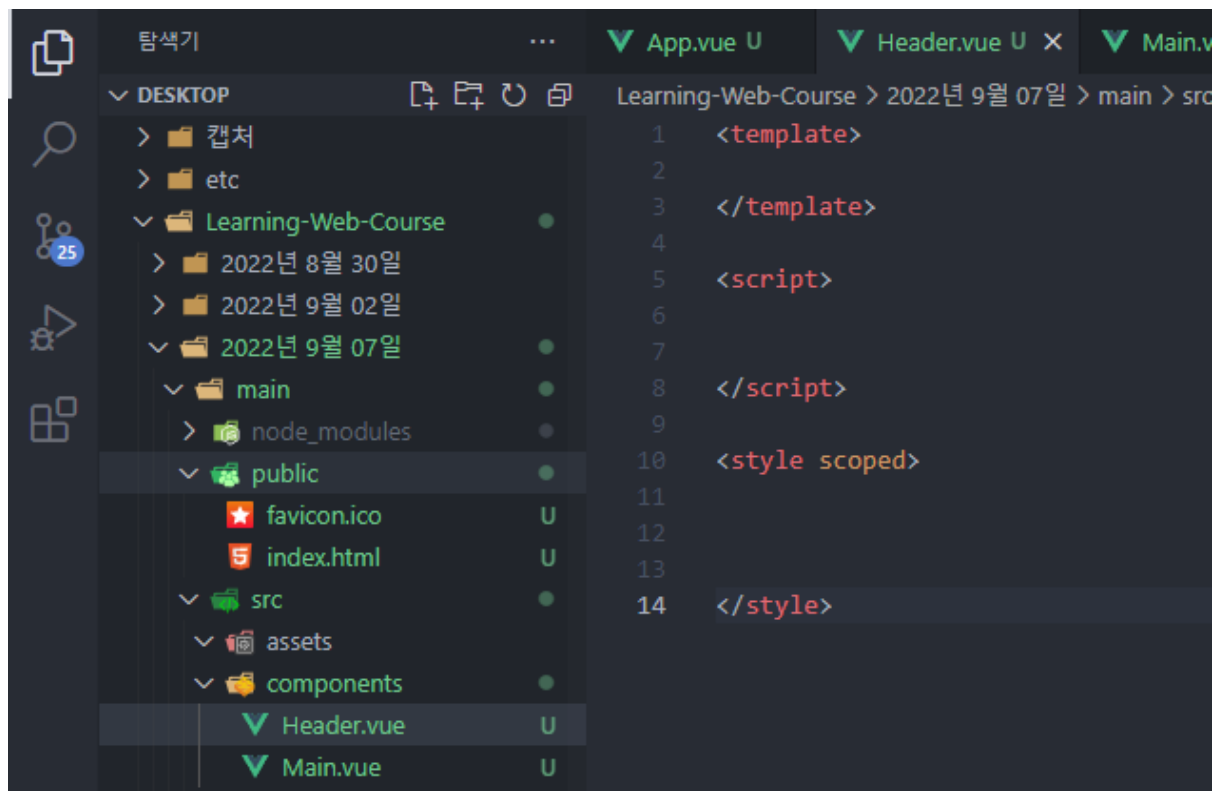
Vue.js devtools 설치





[Page 89]

컴포넌트 구현



```

<template>
  <div id ="app">
    <h1>Hello~~~</h1>
  </div>
</template>

```

```
<script>
export default {};
</script>

<style scoped></style>
```

이자룡 강사님
component 만드세요 하면 위예거 처럼 만들면 됨