SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

Vue.js

SAMSUNG SW ACADEMY FOR YOUTH

FrontEnd Framework

Vue 개요 & Instance

- JavaScript 정리
- Vue 소개
- Vue Instance

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

JavaScript 정리

- JavaScript
  - 프로토타입 기반 객체 생성을 지원하는 동적 스크립트 언어
  - 웹 브라우저에서 주로 사용, Node.js를 이용하여 콘솔 환경에서 사용
  - 웹 브라우저의 UI를 제어하기 위해 만들어진 프로그래밍 언어
  - 자바와 기본 구문이 비슷하다. (C언어의 기본 구문을 바탕)

HTML 자바스크립트 사용

- <script> </script> 태그를 사용
- 문서 내의 위치의 제약이 없다.

```
<html>
<head>
    <meta charset="UTF-8">
    <title>JavaScript</title>
    <script>console.log('head');</script>
</head>
<body>
    <script>console.log('body');</script>
</body>
</html>
```

외부스크립트 참조하기

- .js 확장자를 가진 파일을 생성
- html 문서에서 <script src="외부파일의 위치"></script>

outer.js

```
console.log('hello');
```

html

```
<body>
<h1>외부 스크립트 파일 참조</h1>
<script src="outer.js">console.log("실행되지 않는 부분임");</script>
<script>
    console.log("파일 호출과 별도의 태그를 만들어서 실행");
</script>
</body>
```

변수 (Variable)

- 자바스크립트의 변수 타입은 가리키는 값에 대한 타입을 나타낸다.
- var, let, const 키워드를 이용해서 변수를 선언
- var를 이용한 변수의 선언일 경우 중복 선언이 가능
- undefined 는 변수에 아무 값도 없어서 타입을 알 수 없는 경우를 말한다.
- 동적 타입 : 대입되는 값에 따라서 용도가 변경되는 방식
- 문자, $, _ 로 시작, 대소문자 구분, 예약어 사용 x
- typeof 메서드를 활용하여 데이터 타입을 얻을 수 있다.

https://www.w3schools.com/js/js_reserved.asp

- 숫자형 (Number)
  - 정수와 실수로 나누어 구분하지 않음. (부동소수점 형식)
  - 일반적인 숫자 외 특수 숫자 포함 (Infinity, NaN …)
  - e 를 활용하여 거듭제곱 표현 가능

https://developer.mozilla.org/ko/docs/Web/JavaScript/Reference/Global_Objects/NaN

문자열 (String)

- “ ” 로 감싼다.
- ‘ ’ 로 감싼다.
- `` (backtick) 으로 감싼다. → Template Literal (ES6)
  여러 줄 입력이 가능 - 공백, 줄 넘김 유지
  문자열 내 ${변수명}을 이용하여 변수와 문자열을 결합

```
let msg = "자바스크립트 문자열";
msg = '자바스크립트 문자열';
msg = `자바스크립트 문자열`;

let name = "홍길동";
msg = `나의 이름은 "${name}"입니다.`;

let msg2 = `저의
이름은
홍길동 입니다.`;
```

- 자바스크립트 false
  - 아래의 5가지 값은 false로 인식, 나머지 값은 true로 인식
  - null
  - undefined
  - 0
  - '' (빈문자열)
  - NaN

일치 연산자

- 값과 타입이 일치하는지 체크
- === , !==

```
var i = 100;
var j = "100";

console.log("i == j", i == j);  // true
console.log("i === j", i === j);  // false
```

배열 (Array)

- 배열의 생성 : [] 또는 Array() 활용
- 배열의 크기는 동적으로 변경된다.
- 크기가 지정되어 있지 않은 경우에도 데이터의 입력 가능
- 배열은 여러가지의 데이터 타입을 하나의 배열에 입력할 수 있다.
- push 함수를 이용하여 데이터 추가 가능

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

객체 (Object)

- 객체는 문자열로 이름을 붙인 값들의 집합체이다. (Key : Value)
- 객체에 저장하는 값을 프로퍼티(Property) 라고 한다.
- 객체는 prototype 이라는 특별한 프로퍼티를 가지고 있다.

https://developer.mozilla.org/ko/docs/Web/JavaScript/Reference/Global_Objects/Object

함수

- 자바스크립트에서 함수는 객체 타입으로 값처럼 사용이 가능하다.
- 함수를 변수에 대입하거나 매개변수로 넘길 수 있다.
- 배열의 요소에 넣거나 객체의 프로퍼티로 설정이 가능하다.
- 매개변수의 개수가 일치하지 않아도 호출이 가능하다.
- JavaScript의 함수는 일급 객체(First-class citizen)에 해당
  - 변수에 할당 가능
  - 함수의 매개변수로 전달 가능
  - 함수의 반환 값으로 사용가능

https://developer.mozilla.org/ko/docs/Glossary/First-class_Function

함수 선언식 (function declaration)

- 함수의 이름과 함께 정의하는 방식
- 함수의 이름
- 매개 변수
- 내용
- 호이스팅

```
function func( ) {
    console.log('선언식');
}
func( );
```

함수 표현식 (function expression)

- 익명함수로 정의가능
- 매개 변수
- 내용

```
let func = function ( ) {
    console.log('표현식');
};
func( );
```

함수의 리턴

- 함수의 실행 결과로 함수를 반환할 수 있다.
- 함수가 특별한 값을 리턴 하지 않은 경우 undefined가 반환된다.

```
function func( ) {
    return function (num1, num2) {
        return num1 + num2;
    }
}
function func2( ) { }
```

함수의 호출

- 정의된 함수를 호출 시 함수를 값으로 넘길 수 있다.

```
function func( callFn ) {

    callFn('hello');

}

function fn( msg ) {

    console.log(msg);

}
```

함수 매개변수

- 함수는 호출 시 매개변수의 영향을 받지 않는다.
- arguments 라는 함수 내부의 프로퍼티를 이용하여 매개변수의 처리가 가능하다.
- 자바스크립트의 함수는 오버로딩 개념을 지원하지 않는다.
- 기본 인자 (default arguments)를 사용할 수 있다.

화살표 함수 (Arrow Function)

- ES6에서 추가된 개념
- 함수를 심플하게 정의할 수 있도록 해준다.
- 형태 : (매개변수) => { 명령어 }
- 작성순서
  1. function 키워드 삭제
  2. ( ) 안에 함수가 사용할 파라미터 이름 작성
  3. 화살표 (=>) 를 붙인다.
  4. { } 를 작성하고 블록 안에 함수가 실행할 코드 작성

- Web Storage
  - Web Storage API
  - 키 / 값 쌍으로 값을 저장
  - sessionStorage
  - localStorage
    - setItem(key, value)
    - getItem(key)
    - removeItem(key)
    - clear()
    - key(index)
    - length
    - 값은 반드시 문자열로 저장

https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API

- spread
  - 배열로 묶여 있는 값들을 각각의 개별 값으로 풀어준다.
  - 문자열은 각각의 문자로 나눈다.
- rest
  - 나머지 값들을 모아서 배열로 만듦

```
var params = [ "hello", true, 7 ]
var other = [ 1, 2, ...params ] // [ 1, 2, "hello", true, 7 ]

function f (x, y, ...a) {
    return (x + y) * a.length
}
f(1, 2, ...params) === 9

var str = "foo"
var chars = [ ...str ] // [ "f", "o", "o" ]
```

for/of 문법

```
for (variable of iterable) {
  // code block to be executed
}
```

```
const cars = ["BMW", "Volvo", "Mini"];
let text = "";

for (let x of cars) {
  text += x + " ";
}
```

- class: 자바스크립트 객체의 틀

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```

Property Shorthand

```
const id = ‘ssafy',
name = ' 싸피 ' ,
age = 3;
const member = {
  id: id,
  name: name,
  age: age
};
```

```
const id = ’ssafy',
name = ' 싸피 ' ,
age = 3;
const member = {
  id,
  name,
  age
};
```

Concise method

```
…
const member = {
  id: id,
  name: name,
  age: age,
 info: function () {
    console.log('info');
 }
};
```

```
…
const member = {
  id,
  name,
  age,
 info() {
    console.log('info');
 }
};
```

Destructuring

객체(배열, 객체)에 입력된 값을 개별적인 변수에 할당하는 간편 방식 제공

```
const member = {
    id: 'aaa',
    name: 'bbb',
    age: 22,
};
```

```
let id = member.id;
let name = member.name;
let age = member.age;


ES6
let {id, name, age} = member;
```

Destructuring

```
function getMember() {
      return {
          id: 'aaa',
          name: 'bbb',
          age: 22,
      };
}
```

```
let member = getMember();
let id = member.id;
let name = member.name;
let age = member.age;

ES6
let {id, name, age} =  getMember();
```

Destructuring

```
let member = {
    id: 'aaa',
    name: 'bbb',
    age: 22,
};
```

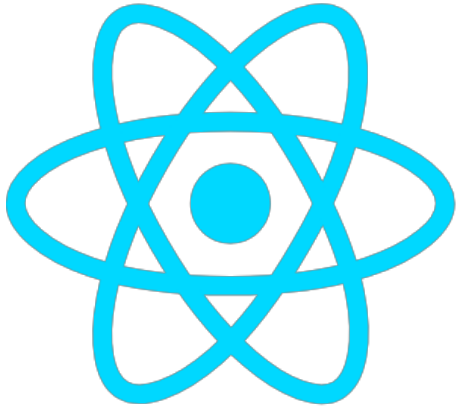
```
function showMember(member) {
  console.log("아이디 :", member.id);
  console.log("이름 :", member.name);
  console.log("나이 :", member.age);
}

function showMember({ id, name, age }) {
  console.log("아이디 :", id);
  console.log("이름 :", name);
  console.log("나이 :", age);
}
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Vue 소개

프론트 엔드 프레임워크 3대장



https://reactjs.org/

https://angular.io/

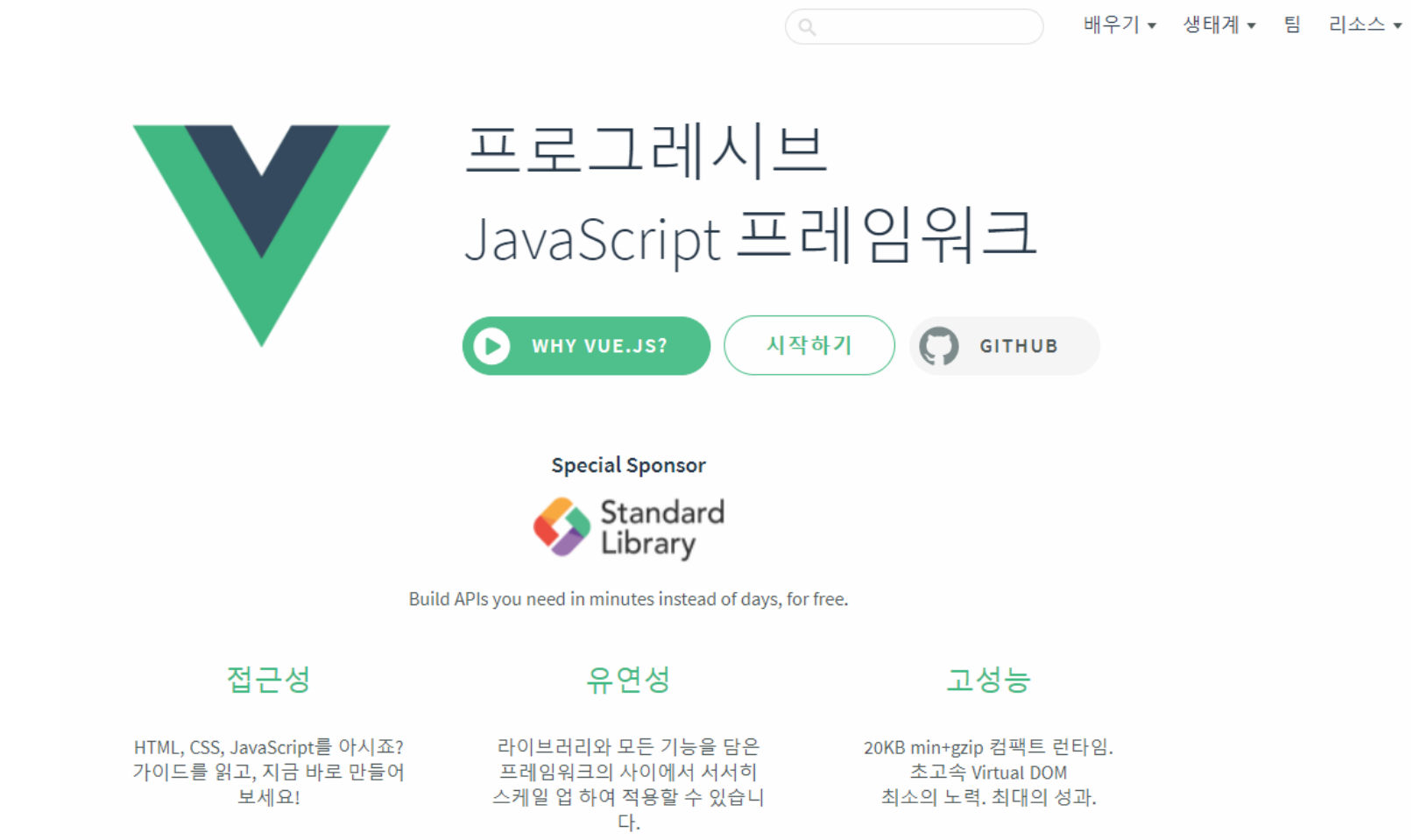
https://vuejs.org/

Vue 소개

Confidential

Vue.js

https://kr.vuejs.org/



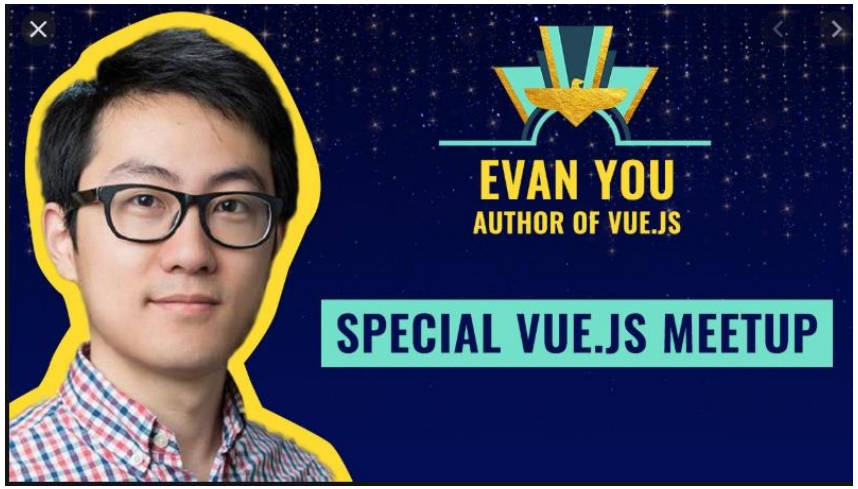
Vue 소개

Vue.js

- 사용자 인터페이스를 만들기 위한 progressive framework 이다.
- 현대적 도구 및 지원하는 라이브러리와 함께 사용하면 정교한 SPA(Single Page Application)을 완벽하게 지원한다.
- Evan You에 의해서 만들어짐.
- Vue 탄생은 Google에서 Angular로 개발하다가 가벼운 걸 만들어 보고 싶은 생각으로 시작한 개인 프로젝트.

Vue.js 특징

- Approachable (접근성)
- Versatile (유연성)
- Performant (고성능)

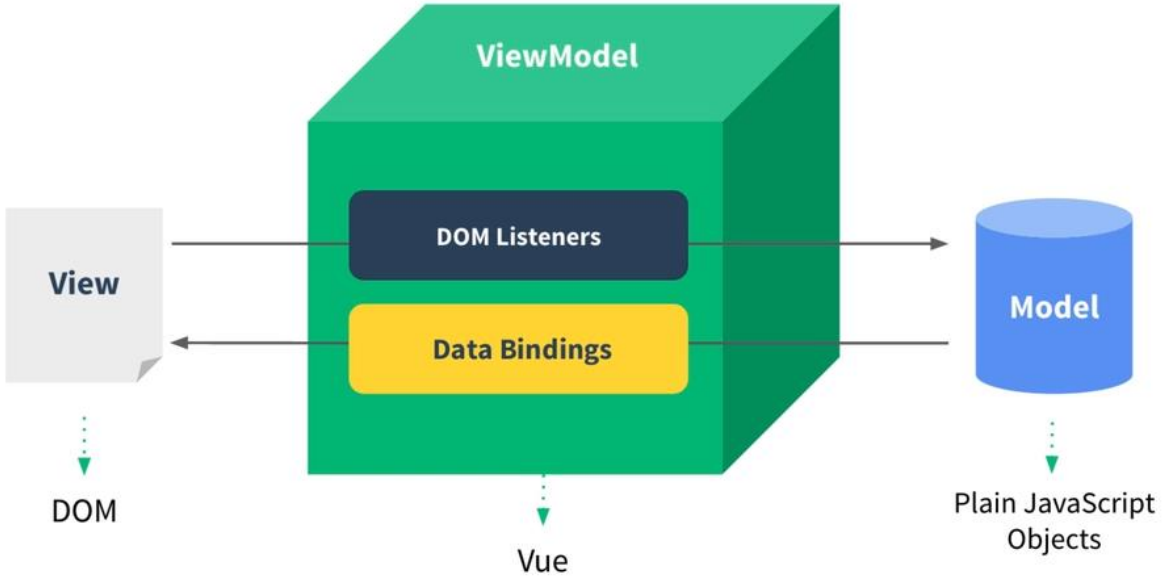


https://evanyou.me/

https://en.wikipedia.org/wiki/Single-page_application

MVVM Pattern

- Model + View + ViewModel
- 애플리케이션 로직을 UI로 부터 분리하기 위해 설계한 디자인 패턴



- **Model**
  - 순수 자바스크립트 객체 (데이터를 담기 위해 사용)
- **View**
  - 웹페이지의 DOM(HTML)
- **ViewModel**
  - Vue 역할, View와 Model 사이에서 Data와 DOM에 관련된 일 처리

Front End Framework Trend

https://redmonk.com/

https://2021.stateofjs.com/ko-KR/

https://gitstar-ranking.com/

설치 방법

- Direct <script> include
  - download
  - CDN : `<script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>`
- NPM
- CLI

https://v2.vuejs.org/v2/guide/installation.html

Vue.js devtools 설치

홈 > 확장 프로그램 > Vue.js devtools

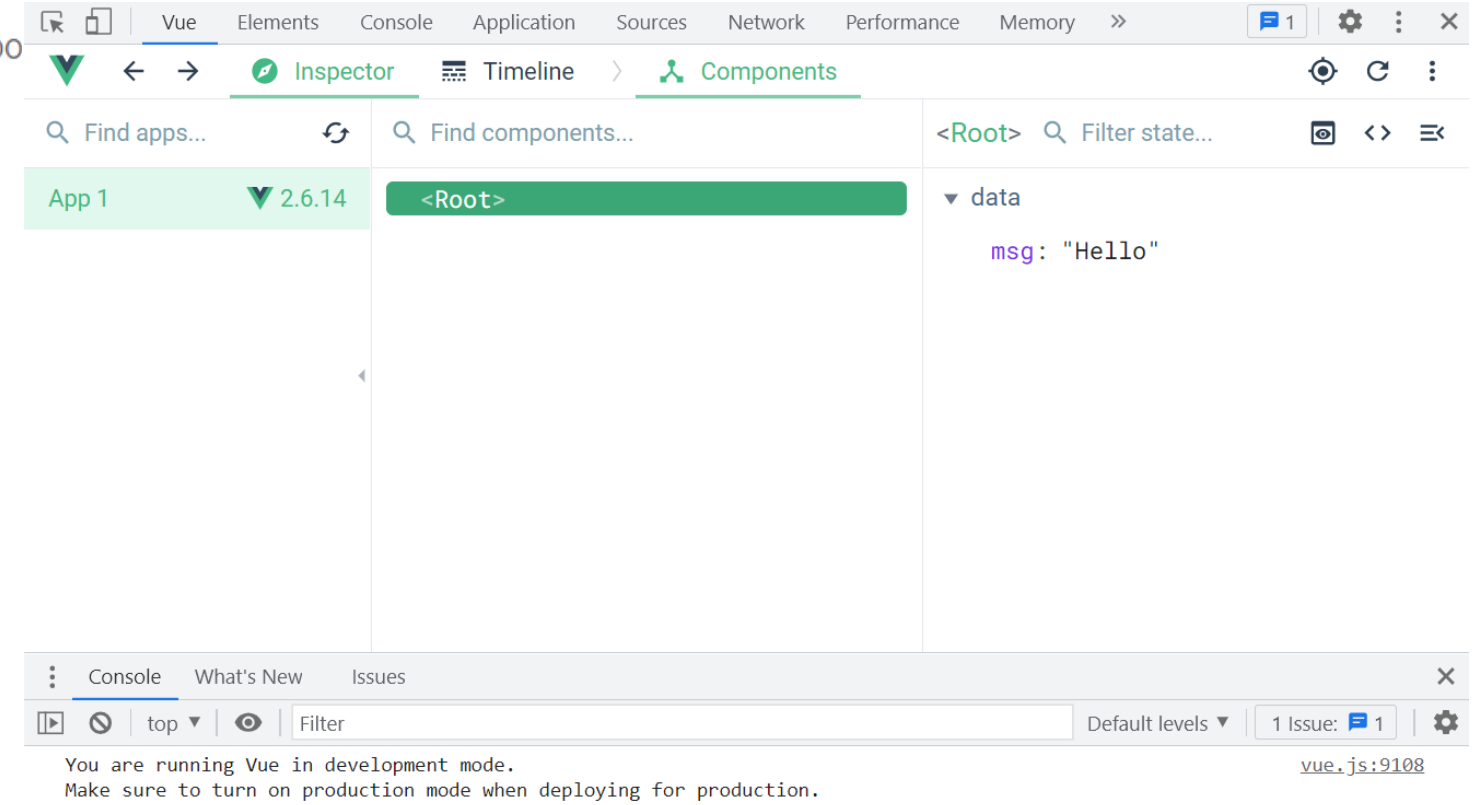
Vue.js devtools

https://vuejs.org

★★★★★ 1,798 | 개발자 도구 | 사용자 1,000,000

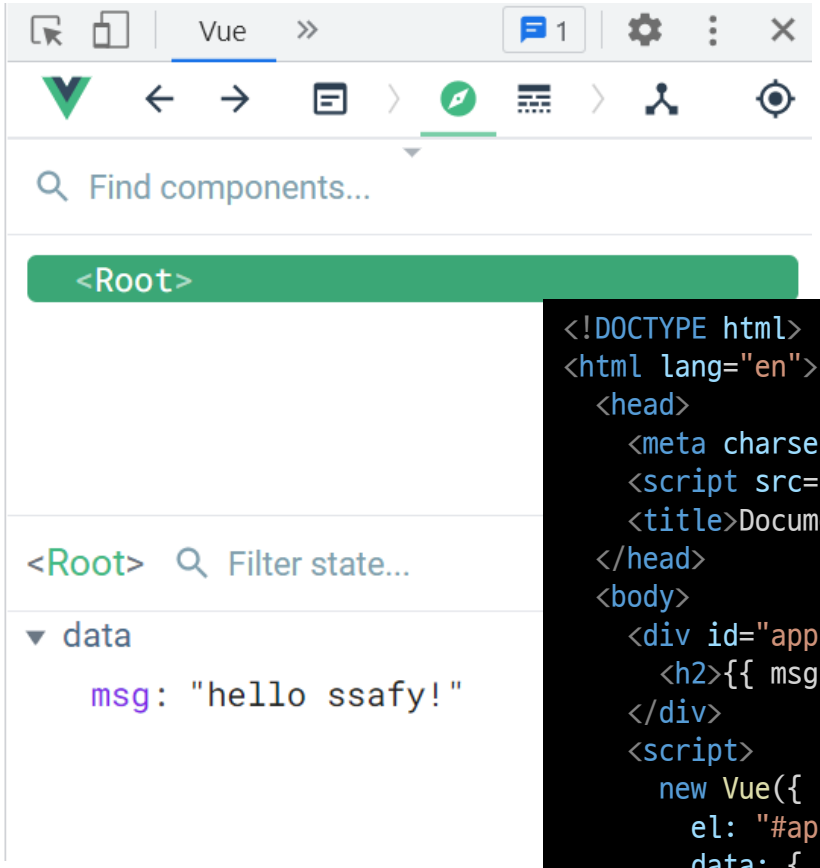
Chrome에서 삭제

https://github.com/vuejs/devtools#vue-devtools



Hello Vue.js

hello ssafy!




```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script>
    <title>Document</title>
  </head>
  <body>
    <div id="app">
      <h2>{{ msg }}</h2>
    </div>
    <script>
      new Vue({
        el: "#app",
        data: {
          msg: "hello ssafy!",
        },
      });
    </script>
  </body>
</html>
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Vue Instance

Vue Instance

Vue Instance 생성

- 모든 Vue 앱은 Vue 함수로 새 Vue 인스턴스를 만드는 것 부터 시작.

```
<script>
    const app = new Vue({
        // 옵션
    })
</script>
```

- Vue 인스턴스를 생성할 때는 Options 객체를 전달해야 함.
- 사용가능한 전체 옵션은 https://kr.vuejs.org/v2/api/#propsData 에서 확인

https://kr.vuejs.org/v2/guide/instance.html

Vue Instance

Vue Instance - Options : el

- Vue 인스턴스에 마운트(연결)할 기존 DOM 엘리먼트 지정
- CSS selector or HTML Element 작성
- new 를 이용한 인스턴스 생성때만 사용한다.

```
<script>
    const app = new Vue({
        el: '#app'
    })
</script>
```

```
<div id="app">

</div>
```

```
<script>
    const div = document.querySelector("#app");
    const app = new Vue({
        el: div
    })
</script>
```

https://kr.vuejs.org/v2/api/#el

Vue Instance - Options : data

- Vue 인스턴스의 데이터 객체
- 객체 또는 함수의 형태로 작성 가능
- 컴포넌트를 정의할 때 data는 데이터를 반환하는 함수로 선언해야 함.
  (일반 객체 사용시 생성된 모든 인스턴스에서 동일 객체 참조 공유, 함수로 생성시 새 복사본을 반환)
- 화살표 함수 사용 X
  화살표 함수가 부모 컨텍스트를 바인딩하기 때문에 'this'는 예상과 달리 Vue 인스턴스가 아님.

```
<script>
    const app = new Vue({
        el: '#app',
        data: {
            msg: 'Hello'
        }
    })
</script>
```

```
data() {
    return {
        msg: 'Hello'
    }
}
```

```
<div id="app">
    <h2> {{ msg }} </h2>
</div>
```

https://kr.vuejs.org/v2/api/#data

Vue Instance

Vue Instance - Options : methods

- Vue 인스턴스에 추가할 메소드
- VM 인스턴스를 통해 직접 접근 or Directive 표현식에서 사용가능.
- 모든 메소드는 자동으로 this 컨텍스트를 Vue 인스턴스에 바인딩.
- 화살표 함수를 메소드 정의하는데 사용하면 안됨.

```
<script>
    const app = new Vue({
        el: '#app',
        data: { count: 1 },
        methods: {
            plus: function() {
                this.count++;
            }
        }
    })
</script>
```

```
<div id="app">
    <h2> {{ count }} </h2>
</div>
```

https://kr.vuejs.org/v2/api/#methods

Vue Instance - Options : template

- Vue 인스턴스의 마크업으로 사용할 문자열 템플릿.
- template은 마운트(연결)된 엘리먼트를 대체함.
- Vue 옵션에 render 함수가 있으면 템플릿 속성은 무시

https://kr.vuejs.org/v2/api/#template

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

다음 방송에서 만나요!

삼성 청년 SW 아카데미