SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

Vue.js

<알림>

본 강의는 삼성 청년 SW아카데미의 컨텐츠로
보안서약서에 의거하여
강의 내용을 어떠한 사유로도 임의로 복사,
촬영, 녹음, 복제, 보관, 전송하거나
허가 받지 않은 저장매체를
이용한 보관, 제3자에게 누설, 공개,
또는 사용하는 등의 행위를 금합니다.

SAMSUNG SW ACADEMY FOR YOUTH

목차

SAMSUNG SW ACADEMY FOR YOUTH

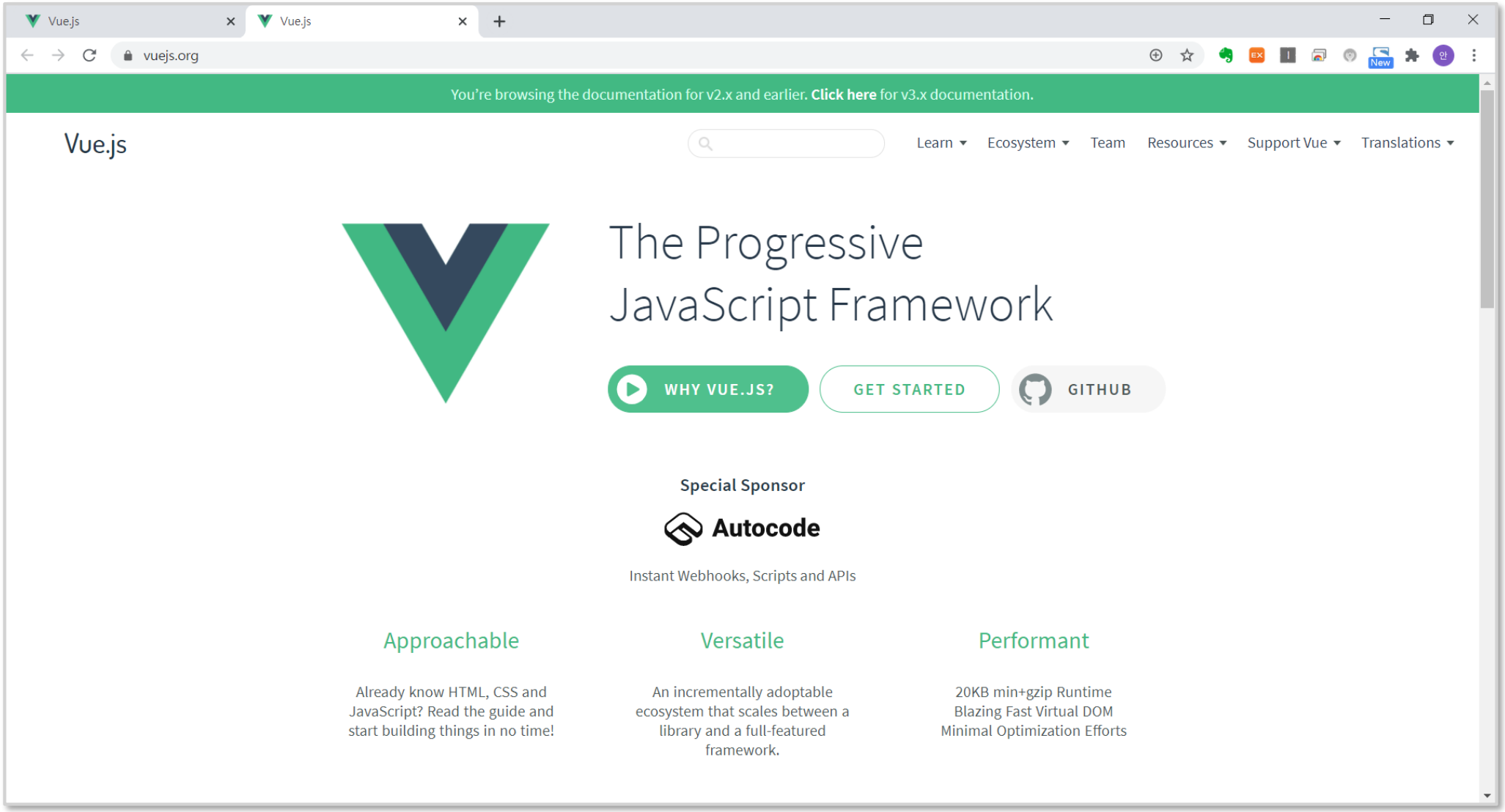
함께가요 미래로!
Enabling People

Vue.js

삼성 청년 SW 아카데미

Vue.js

✓ https://vuejs.org/.



✓ Vue.js

- Evan You에 의해서 만들어짐.
- Vue 탄생은 Google에서 Angular로 개발하다가 가벼운 걸 만들어 보고 싶은 생각으로 시작한 개인 프로젝트.
- 사용자 인터페이스를 만들기 위해 사용하는 오픈 소스 Progressive Framework.

✓ Vue.js 특징.

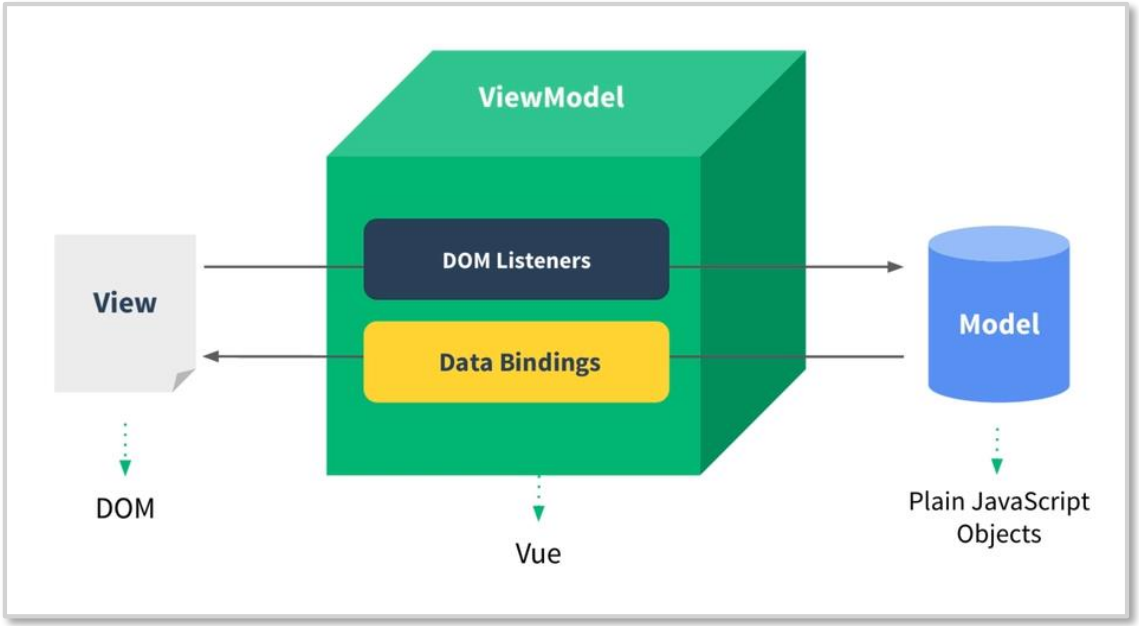
- Approachable (접근성).
- Versatile (유연성).
- Performant (고성능).

✓ MVVM Pattern.

- Model + View + ViewModel.
- Model : 순수 자바스크립트 객체.
- View : 웹페이지의 DOM.
- ViewModel : Vue의 역할.

기존에는 자바스크립트로 view에 해당하는 DOM에 접근하거나 수정하기 위해 jQuery와 같은 library 이용.

Vue는 view와 Model을 연결하고 자동으로 바인딩하므로 양방향 통신을 가능하게 함.



✓ MVC VS MVVM

ㅇㅋㅇㅋ!!

Controller

화면 업데이트해!!!

data 갱신 좀!!!

View

Model

ㅇㅋㅇㅋ!!

View

이거 바꿔달래!!

ViewModel

내가 바뀌는거 보고 있을게~~

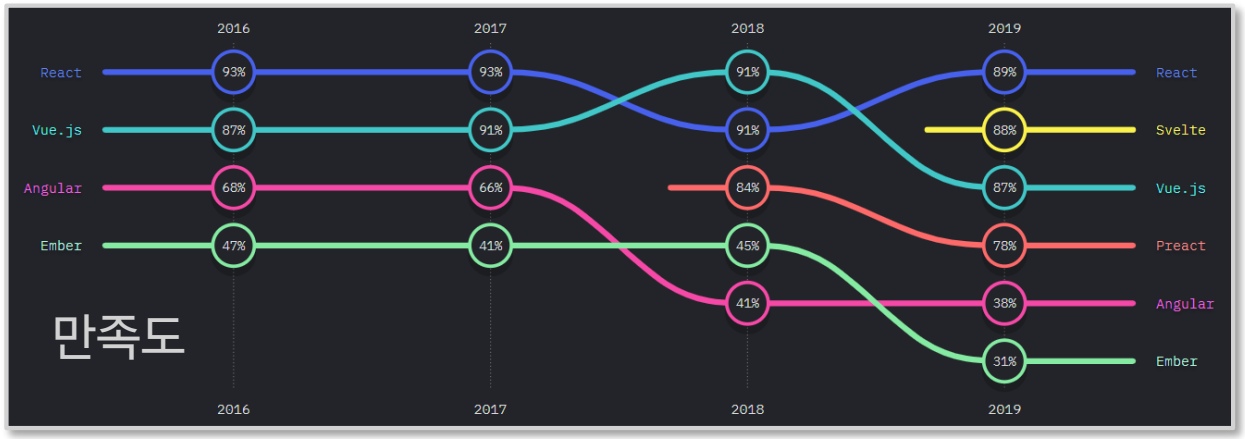
data 갱신 좀!!!

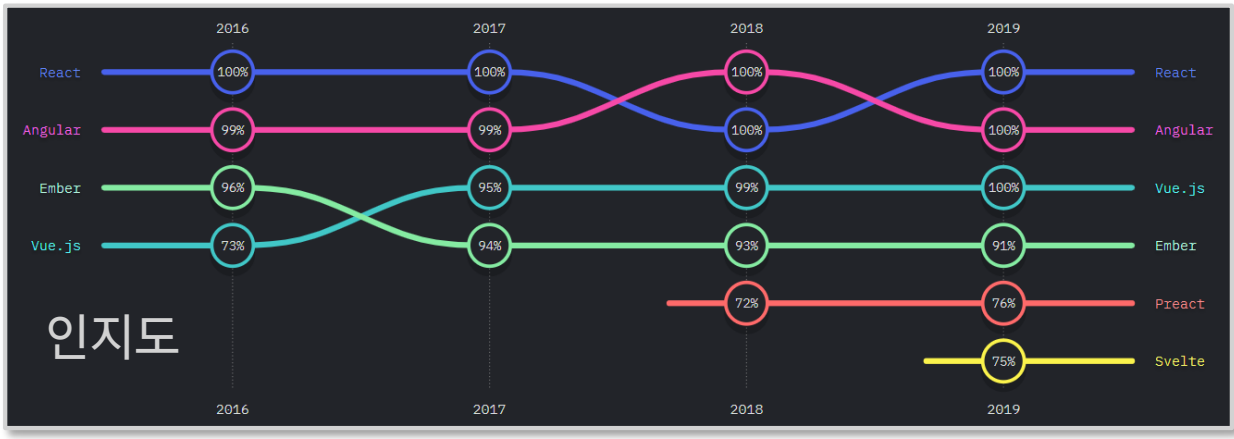
data 갱신 다 했어!!!

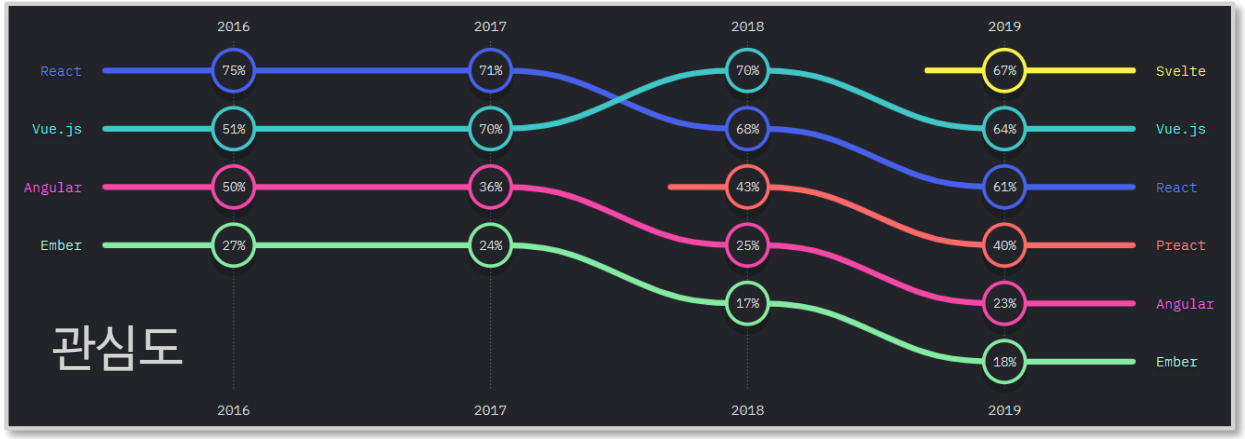
Model

✓ Front End Framework Trend.

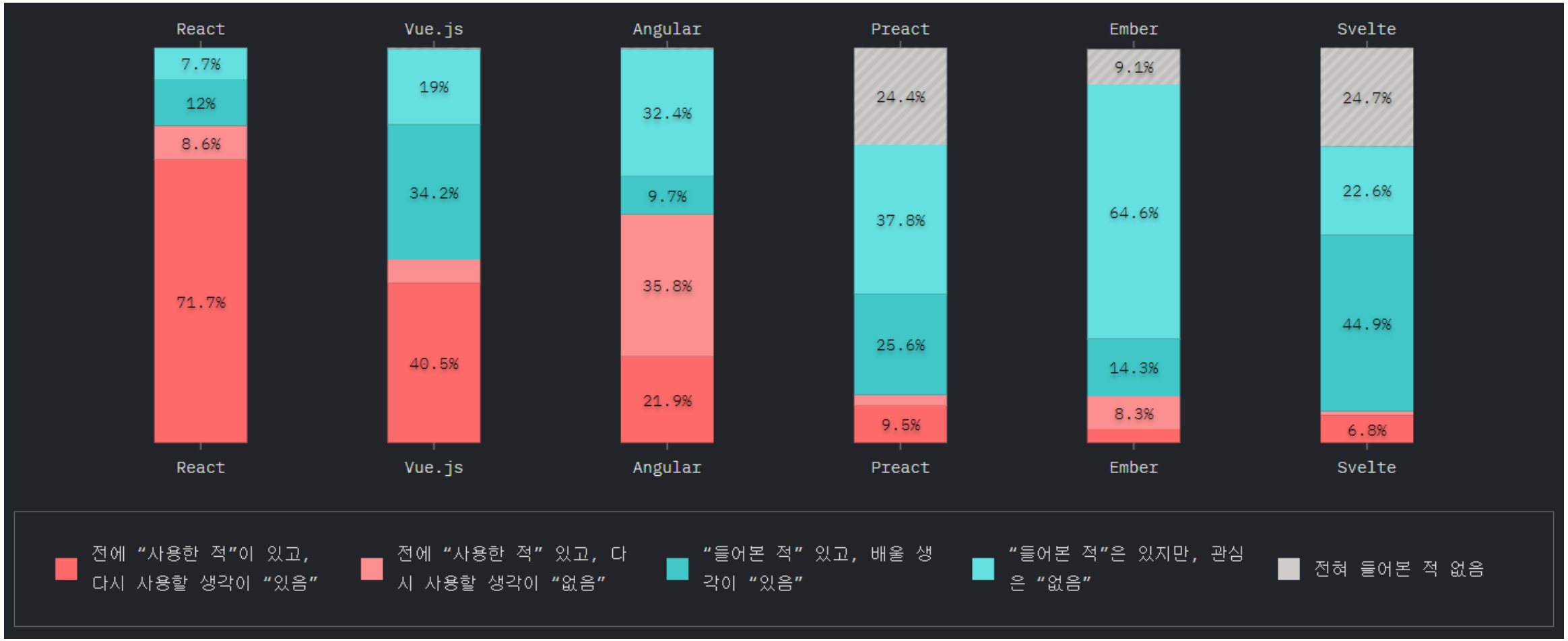
- https://2020.stateofjs.com/ko-KR/technologies/front-end-frameworks/



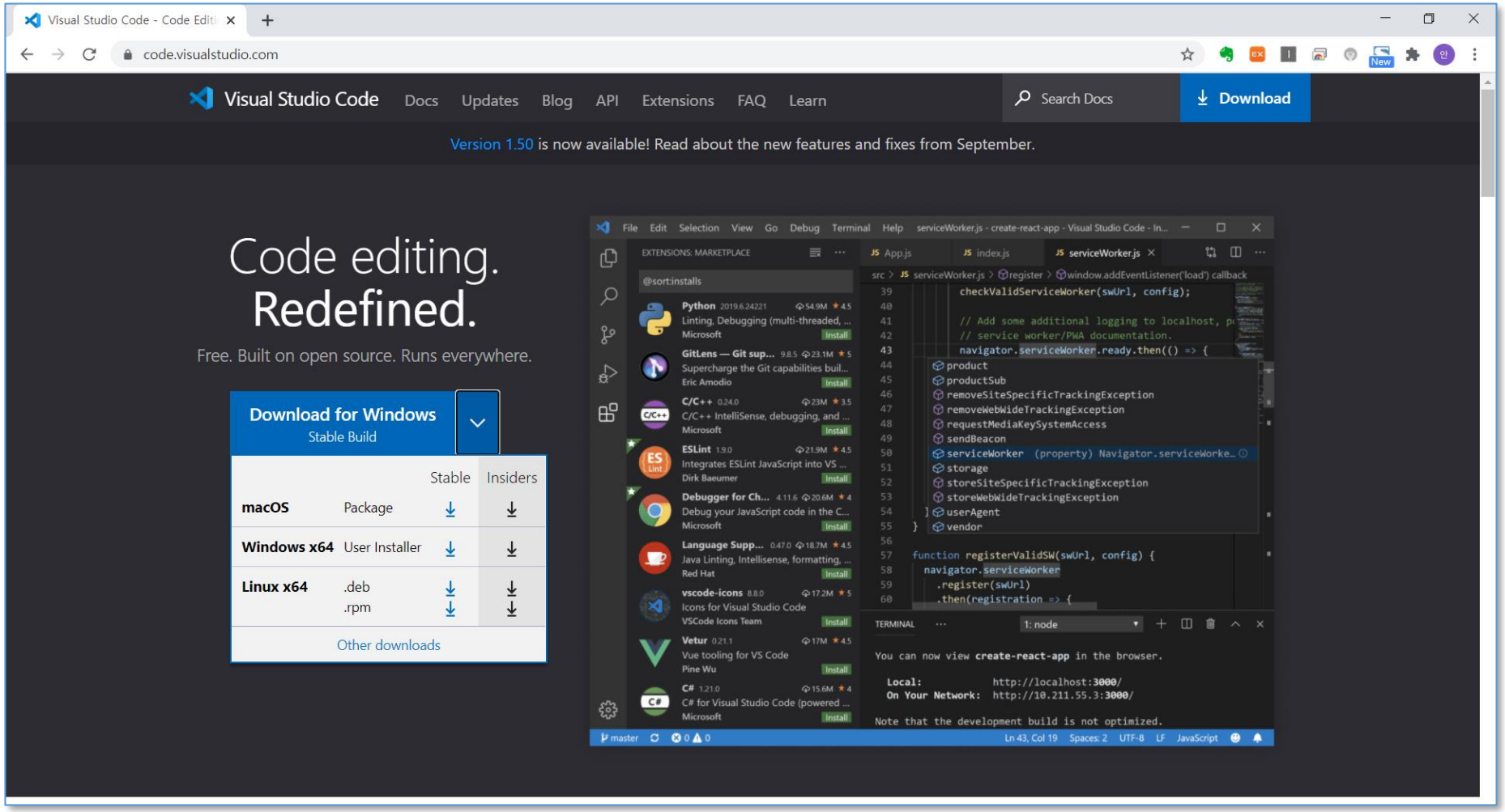




✓ Front End Framework Trend.

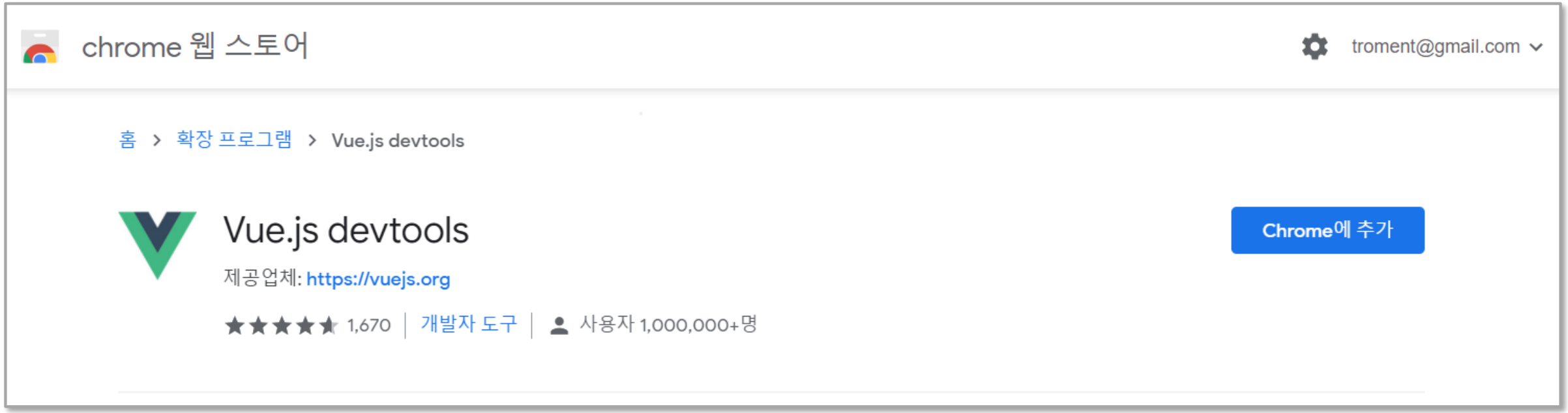


- ✓ https://code.visualstudio.com
  - 운영체제에 맞는 Stable Download.



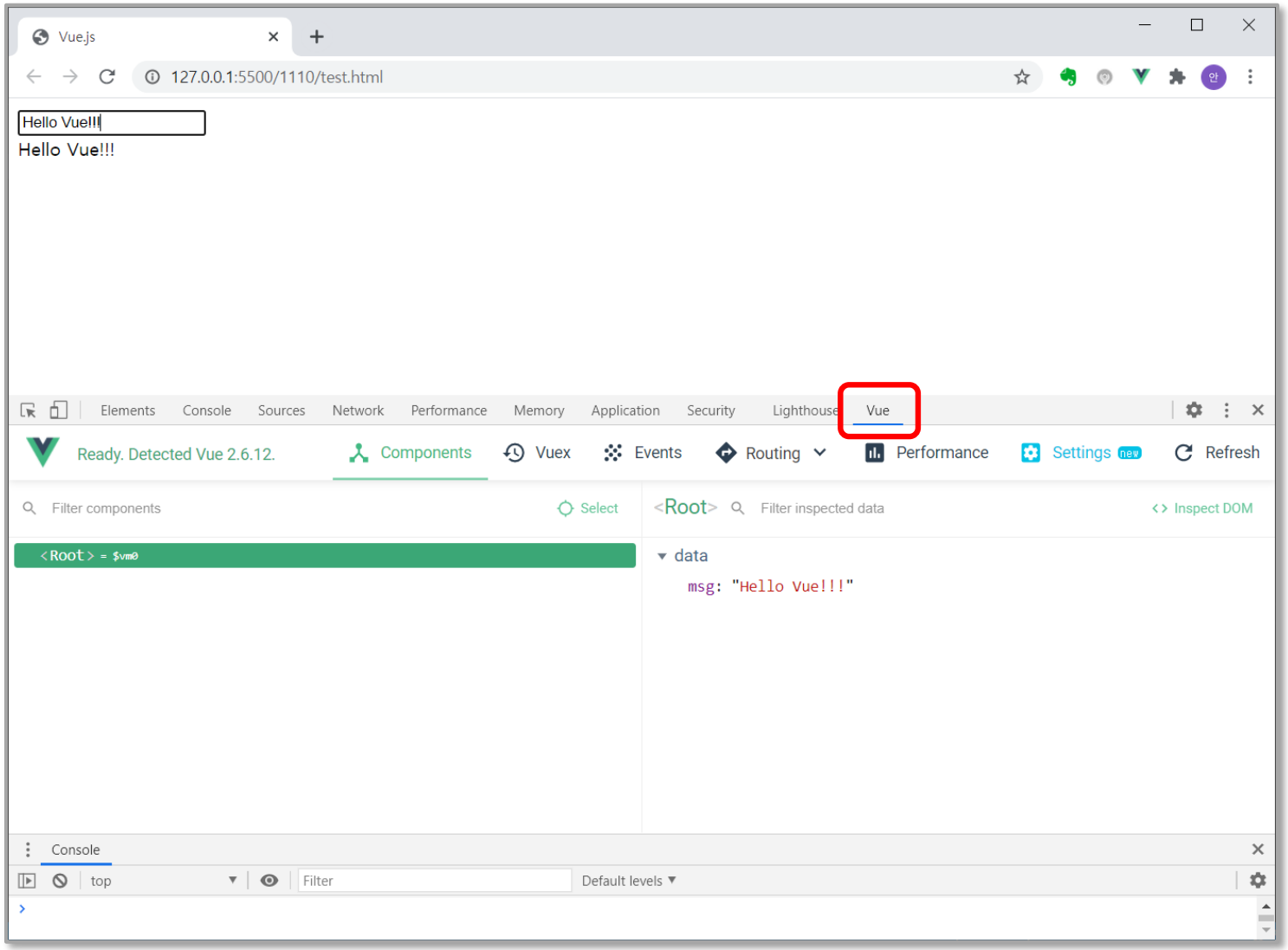
✓ Vue.js devtools

- https://kr.vuejs.org/v2/guide/installation.html
- https://chrome.google.com/webstore/detail/vuejs-devtools/nhdogjmejiglipccpnnnanhbledajbpd?hl=ko&



✓ Vue.js devtools

- 개발자도구



✓ Vue.js Installation.

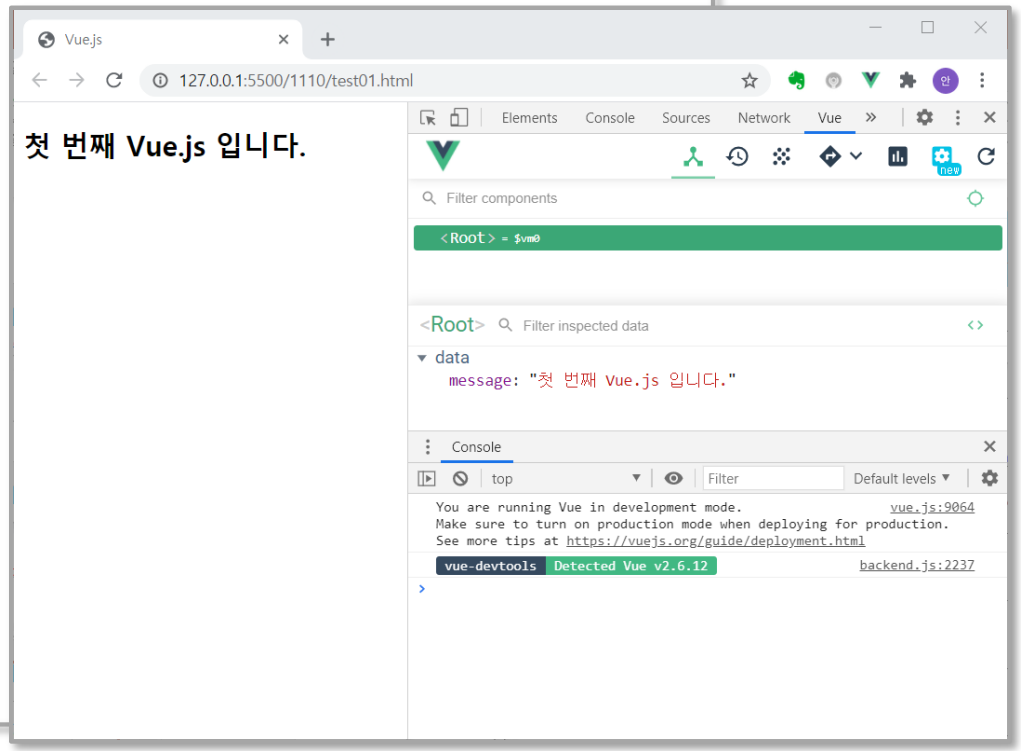
- Direct <script> include.
  - download
  - CDN : `<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>`
- NPM
- CLI

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

✓ Hello Vue.js

test01.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Vue.js</title>
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
</head>
<body>
    <div id="app">
        <h2>{{message}}</h2>
    </div>
    <script>
        new Vue({
            el: '#app',
            data: {
                message: '첫 번째 Vue.js 앱입니다.'
            }
        });
    </script>
</body>
</html>
```



SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Vue Instance

삼성 청년 SW 아카데미

✓ Vue Instance 생성.

```
<script>
    new Vue({
        el: '#app',
        data: {
            message: '첫 번째 Vue.js 앱입니다.'
        }
    });
</script>
```

인스턴스

el 속성

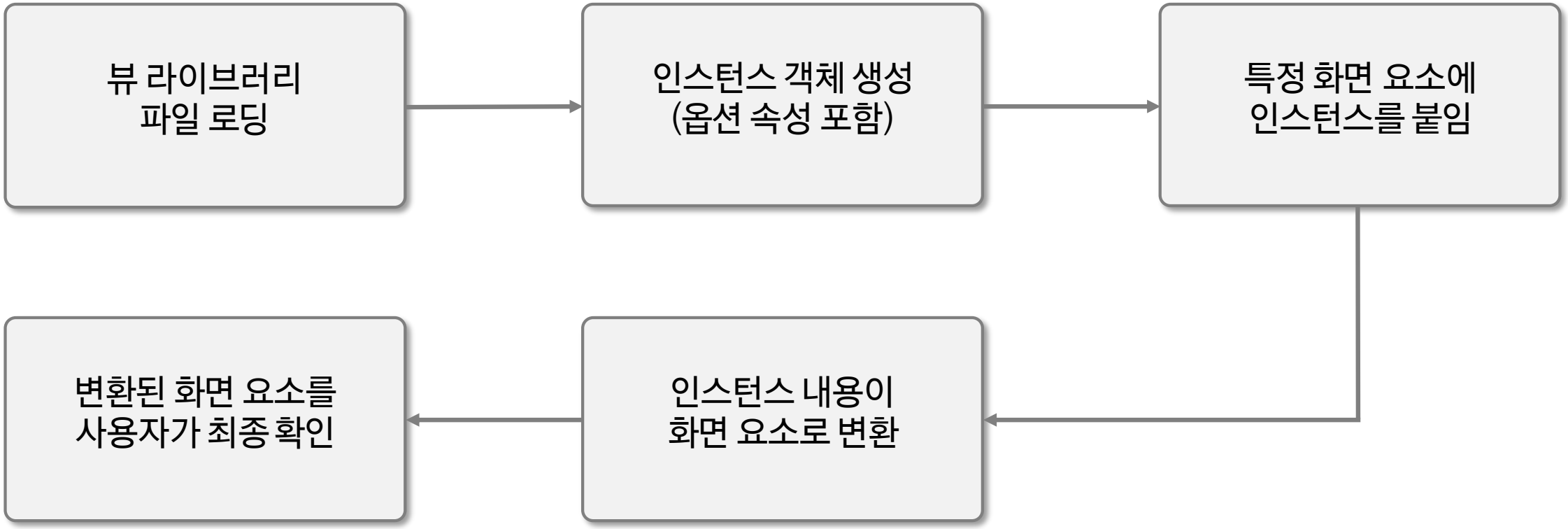
data 속성

```
data() {
    return {
        message: 'test.'
    }
}
```

속성	설명
el	Vue가 적용될 요소 지정. CSS Selector or HTML Element
data	Vue에서 사용되는 정보 저장. 객체 또는 함수의 형태
template	화면에 표시할 HTML, CSS등의 마크업 요소를 정의하는 속성. 뷰의 데이터 및 기타 속성들도 함께 화면에 그릴 수 있다.
methods	화면 로직 제어와 관계된 method를 정의하는 속성. 마우스 클릭 이벤트 처리와 같이 화면의 전반적인 이벤트와 화면 동작과 관련된 로직을 추가.
created	뷰 인스턴스가 생성되자 마자 실행할 로직을 정의.

✓ Vue Instance의 유효범위.

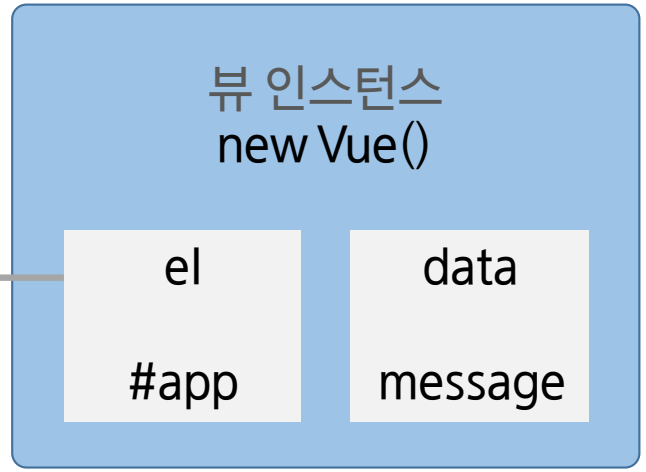
- Vue Instance를 생성하면 HTML의 특정 범위 안에서만 옵션 속성들이 적용.
- el 속성과 밀접한 관계가 있다.



인스턴스가 화면에 적용되는 과정.

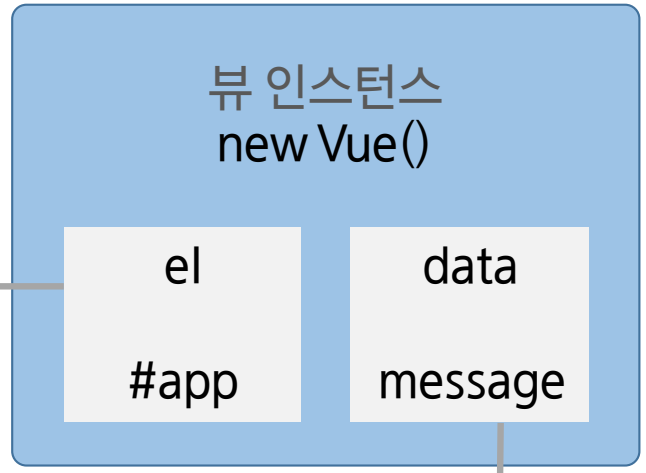
✓ Vue Instance의 유효범위.

- Vue()로 인스턴스가 생성됨.
- el 속성에 지정한 화면 요소(돔)에 인스턴스가 부착됨.




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

- el 속성에 인스턴스가 부착된 후 data 속성이 el 속성에 지정한 화면 요소와 그 이하 레벨의 화면 요소에 적용되어 값이 치환.



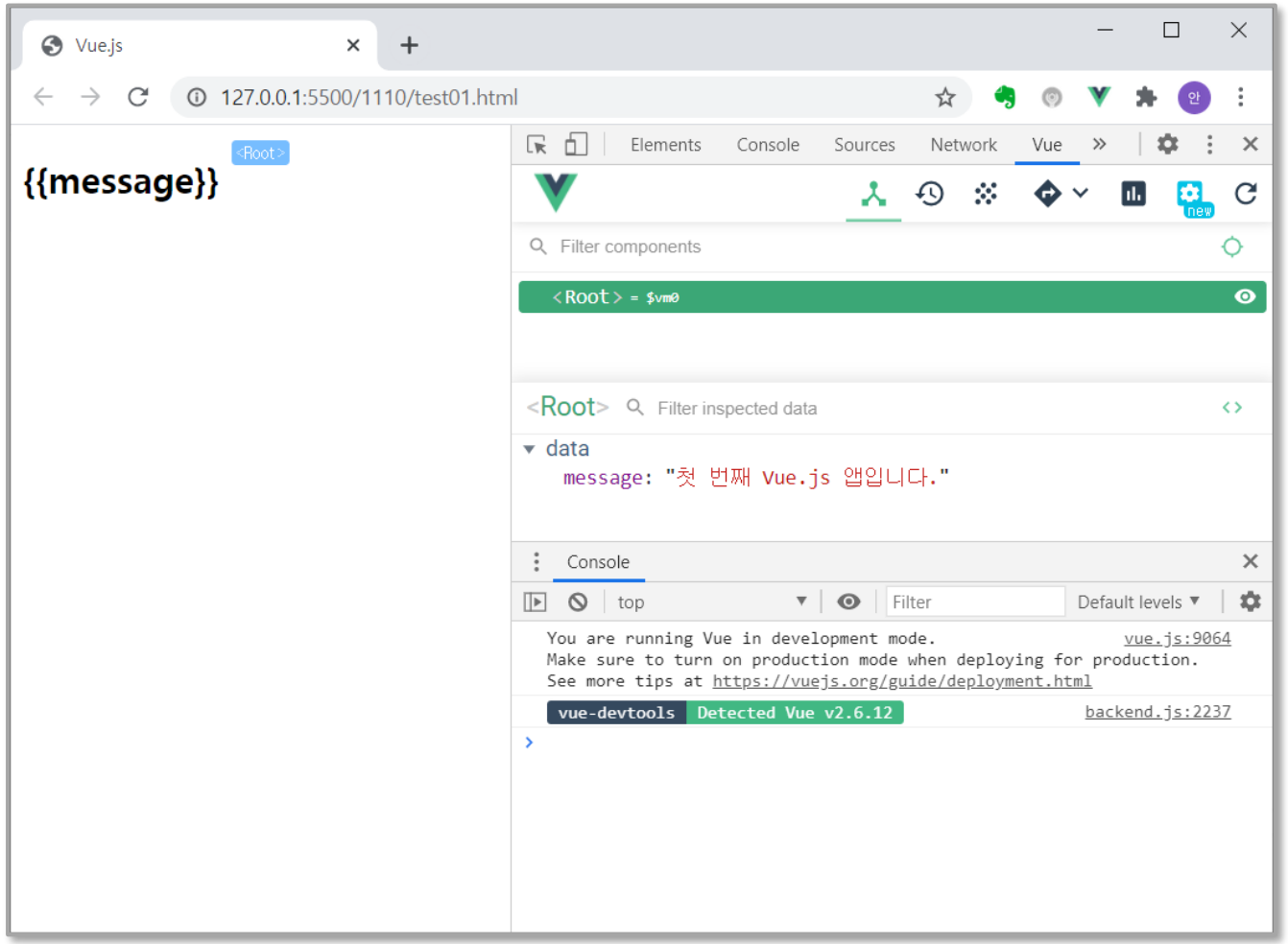

```
<div id="app">
    <h2>{{message}}</h2>
</div>
```

✓ Vue Instance의 유효범위.

- 유효범위를 벗어난 경우.

```
<div id="app">

</div>
<h2>{{message}}</h2>
```

유효범위



SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

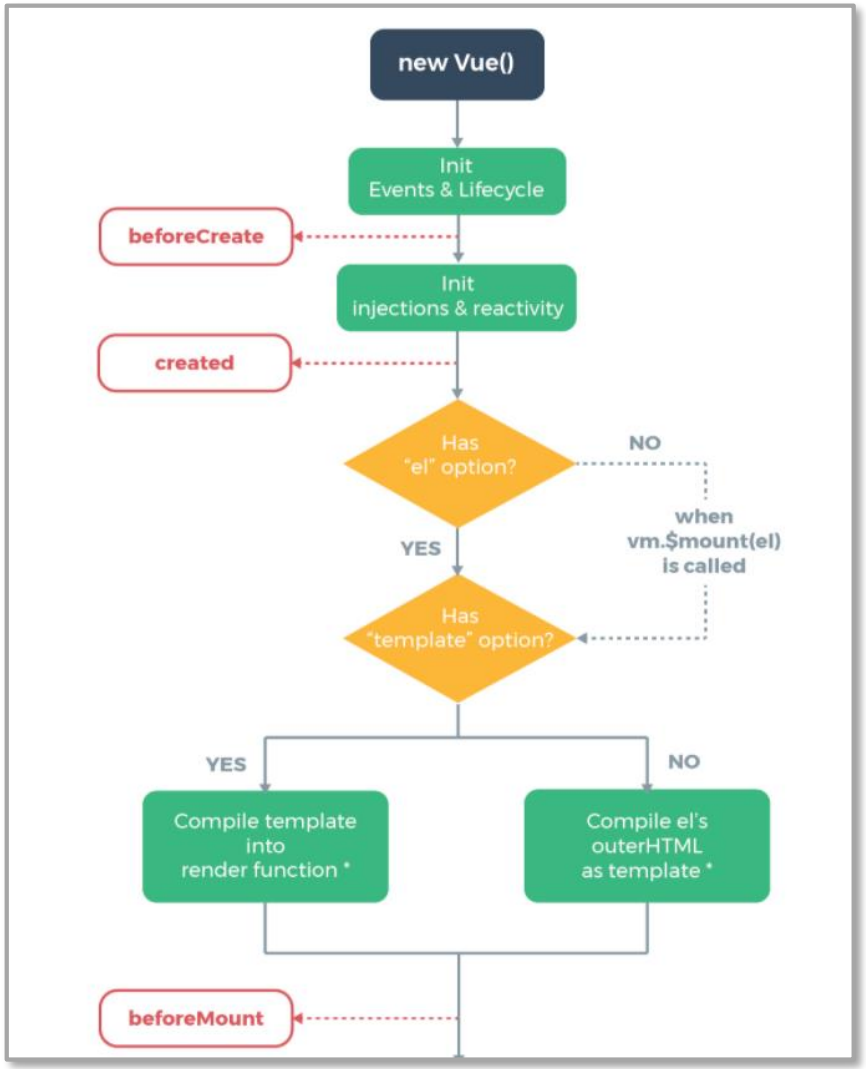
SAMSUNG SW ACADEMY FOR YOUTH

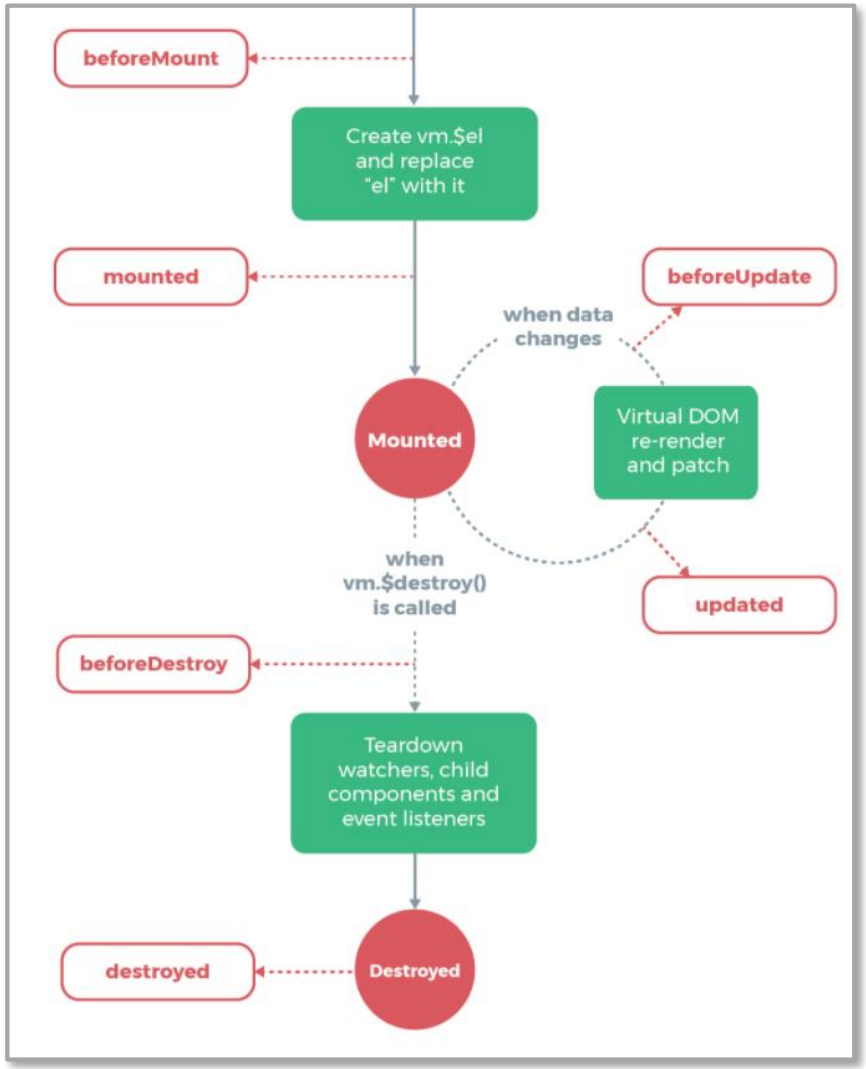
함께가요 미래로! Enabling People

Vue Instance Life-Cycle

삼성 청년 SW 아카데미

✓ Vue Instance Life Cycle.





https://vuejs.org/v2/guide/instance.html#Lifecycle-Diagram

✓ Vue Instance Life Cycle.

- Life Cycle은 크게 나누면 Instance의 **생성**, 생성된 Instance를 화면에 **부착**, 화면에 부착된 Instance의 내용이 **갱신**, Instance가 제거되는 **소멸**의 4단계로 나뉜다.

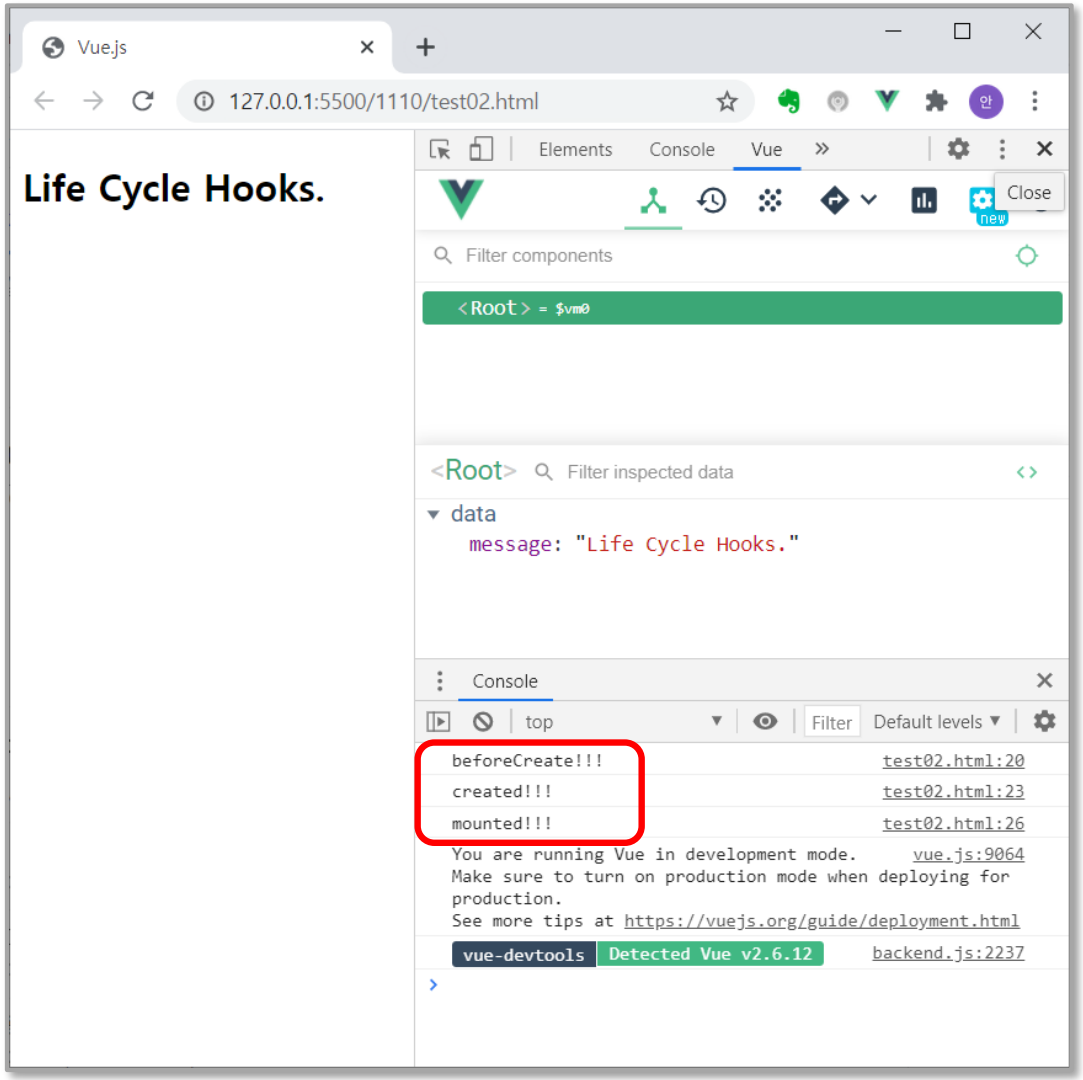
[Life Cycle Hooks]

life cycle 속성	설명
beforeCreate	Vue Instance가 생성되고 각 정보의 설정 전에 호출. DOM과 같은 화면요소에 접근 불가.
created	Vue Instance가 생성된 후 데이터들의 설정이 완료된 후 호출. Instance가 화면에 부착하기 전이기때문에 template 속성에 정의된 DOM 요소는 접근 불가. 서버에 데이터를 요청(http 통신)하여 받아오는 로직을 수행하기 좋다.
beforeMount	마운트가 시작되기 전에 호출.
mounted	지정된 element에 Vue Instance 데이터가 마운트 된 후에 호출. template 속성에 정의한 화면 요소에 접근할 수 있어 화면 요소를 제어하는 로직 수행.
beforeUpdate	데이터가 변경될 때 virtual DOM이 랜더링, 패치 되기 전에 호출.
updated	Vue에서 관리되는 데이터가 변경되어 DOM이 업데이트 된 상태. 데이터 변경 후 화면 요소 제어와 관련된 로직을 추가.
beforeDestroy	Vue Instance가 제거되기 전에 호출.
destroyed	Vue Instance가 제거된 후에 호출.

✓ Life Cycle Hooks Ex.

test02.html

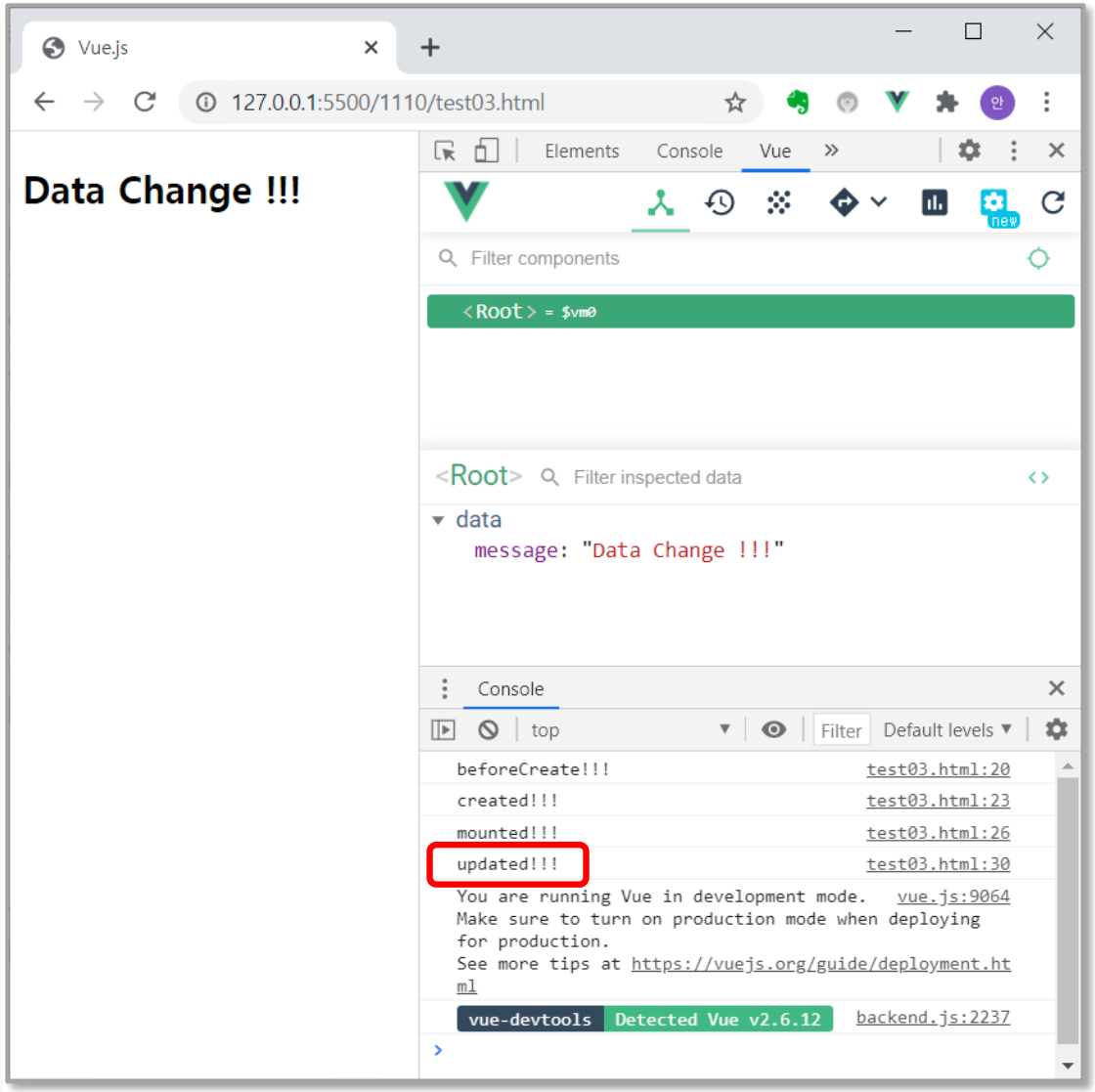
```
<div id="app">
    <h2>{{message}}</h2>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            message: 'Life Cycle Hooks.'
        },
        beforeCreate() {
            console.log('beforeCreate!!!');
        },
        created() {
            console.log('created!!!');
        },
        mounted() {
            console.log('mounted!!!');
        },
        updated() {
            console.log('updated!!!');
        },
    });
</script>
```



✓ Life Cycle Hooks Ex.

test03.html

```
<div id="app">
    <h2>{{message}}</h2>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            message: 'Life Cycle Hooks.'
        },
        beforeCreate() {
            console.log('beforeCreate!!!');
        },
        created() {
            console.log('created!!!');
        },
        mounted() {
            console.log('mounted!!!');
            this.message = 'Data Change !!!'
        },
        updated() {
            console.log('updated!!!');
        },
    });
</script>
```



SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

template - 보간법 interpolation

삼성 청년 SW 아카데미

✓ 보간법(Interpolation)

✓ 문자열.

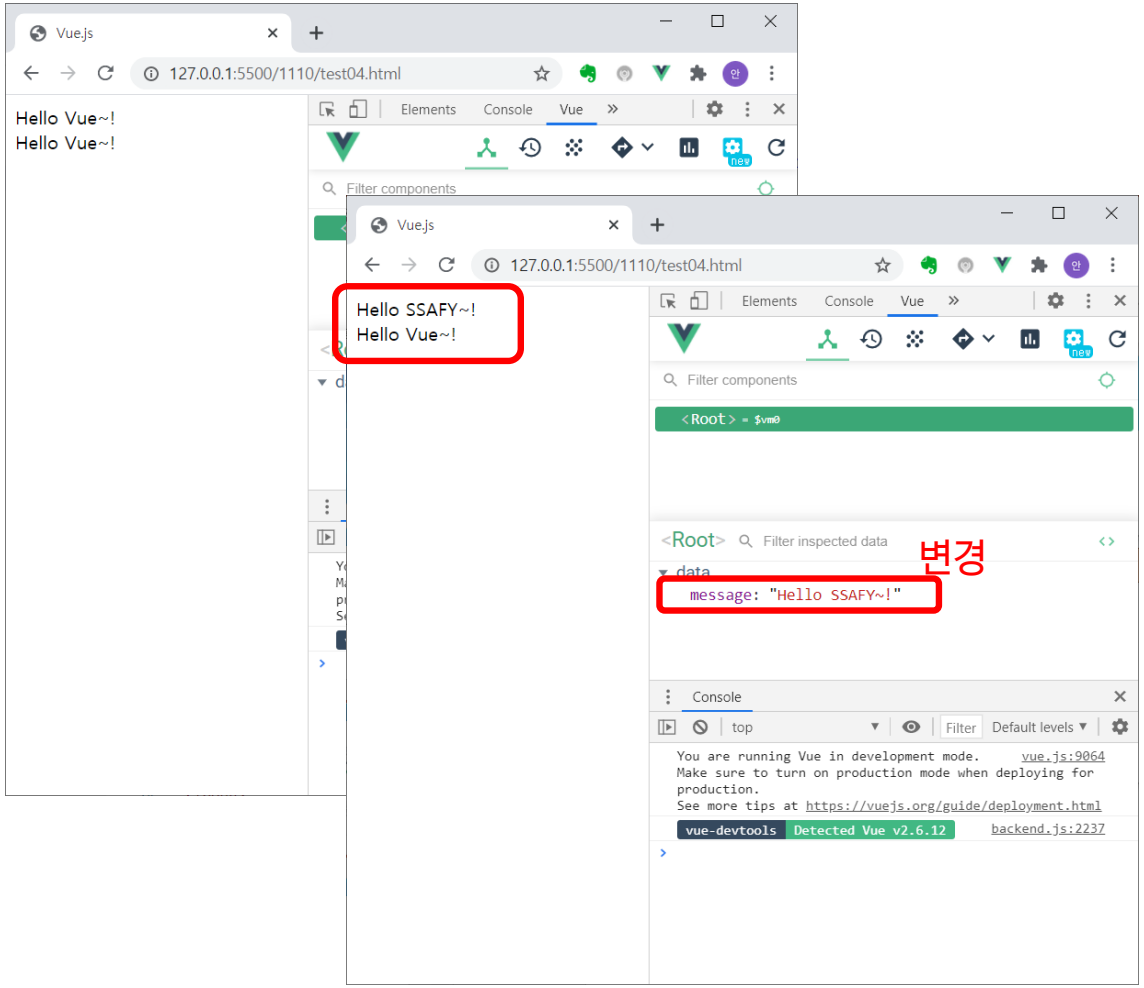
- 데이터 바인딩의 가장 기본 형태는 “Mustache” 구문(이중 중괄호)을 사용한 텍스트 보간.
  - {{ 속성명 }}
- v-once 디렉티브를 사용하여 데이터 변경 시 업데이트 되지 않는 일회성 보간을 수행
  - v-once

```
<span>메시지: {{ msg }}</span>

<span v-once>다시는 변경하지 않습니다: {{ msg }}</span>
```

✓ 문자열.

test04.html

```
<div id="app">
    <div>{{message}}</div>
    <div v-once>{{message}}</div>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            message: 'Hello Vue~!'
        }
    });
</script>
```



✓ 원시 HTML.

- 이중 중괄호(mustaches)는 HTML이 아닌 일반 텍스트로 데이터를 해석
- 실제 HTML을 출력하려면 v-html 디렉티브를 사용

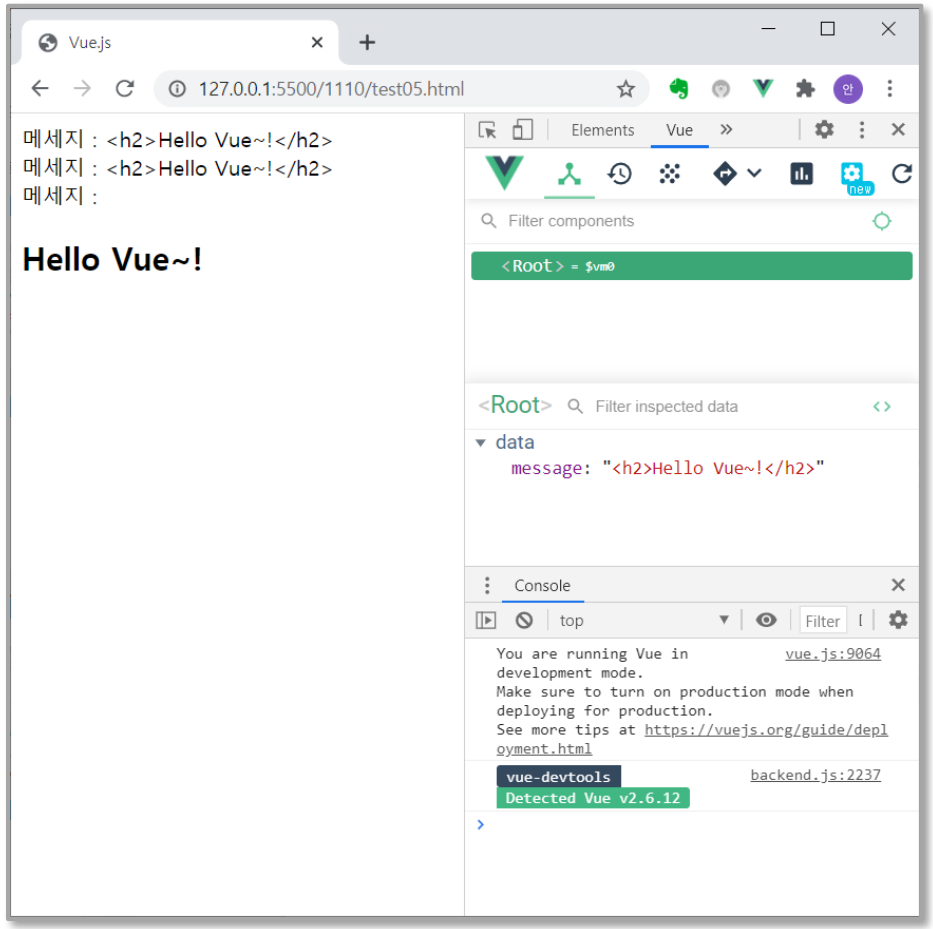
```
<p>Using mustaches: {{ rawHtml }}</p>

<p>Using v-html directive: <span v-html="rawHtml"></span></p>
```

Using mustaches: This should be red.
Using v-html directive: This should be red.

✓ 원시 HTML.

```
<div id="app">
    <div>메세지 : {{message}}</div>
    <div v-text="'메세지 : ' + message">무시된다</div>
    <div v-html="'메세지 : ' + message">무시된다</div>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            message: '<h2>Hello Vue~!</h2>'
        }
    });
</script>
```

test05.html



✓ JavaScript 표현식 사용.

- Vue.js는 모든 데이터 바인딩 내에서 JavaScript 표현식의 모든 기능을 지원.

```
{{ number + 1 }}

{{ ok ? 'YES' : 'NO' }}

{{ message.split('').reverse().join('') }}

<div v-bind:id="'list-' + id"></div>
```

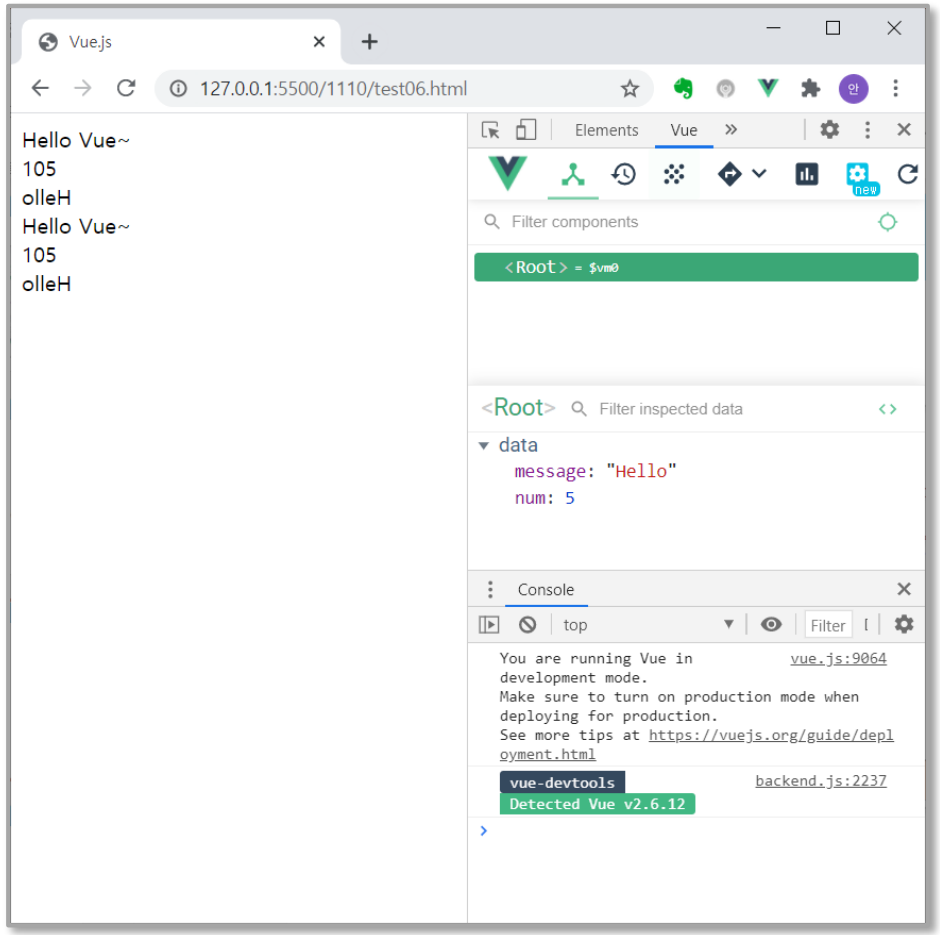
- 한가지 제한사항은 각 바인딩에 하나의 단일 표현식 만 포함될 수 있으므로 아래처럼 작성하면 안된다.

```
<!-- 아래는 구문입니다, 표현식이 아닙니다. -->
{{ var a = 1 }}

<!-- 조건문은 작동하지 않습니다. 삼항 연산자를 사용해야 합니다. -->
{{ if (ok) { return message } }}
```

✓ JavaScript 표현식 사용.

```
<div id="app">
    <div>{{message + ' Vue~'}}</div>
    <div>{{num > 10 ? num * num : num + 100}}</div>
    <div>{{message.split("").reverse().join("")}}</div>
    <div v-text="message + ' Vue~'"></div>
    <div v-text="num > 10 ? num * num : num + 100"></div>
    <div v-text="message.split('').reverse().join('')"></div>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            num: 5,
            message: 'Hello'
        }
    });
</script>
```

test06.html



SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로! Enabling People

template - Directive

삼성 청년 SW 아카데미

✓ 디렉티브 (Directives).

- 디렉티브는 v- 접두사가 있는 특수 속성.
- 디렉티브 속성 값은 단일 JavaScript 표현식이 된다. (v-for는 예외)
- 디렉티브의 역할은 표현식의 값이 변경될 때 사이드 이펙트를 반응적으로 DOM에 적용.

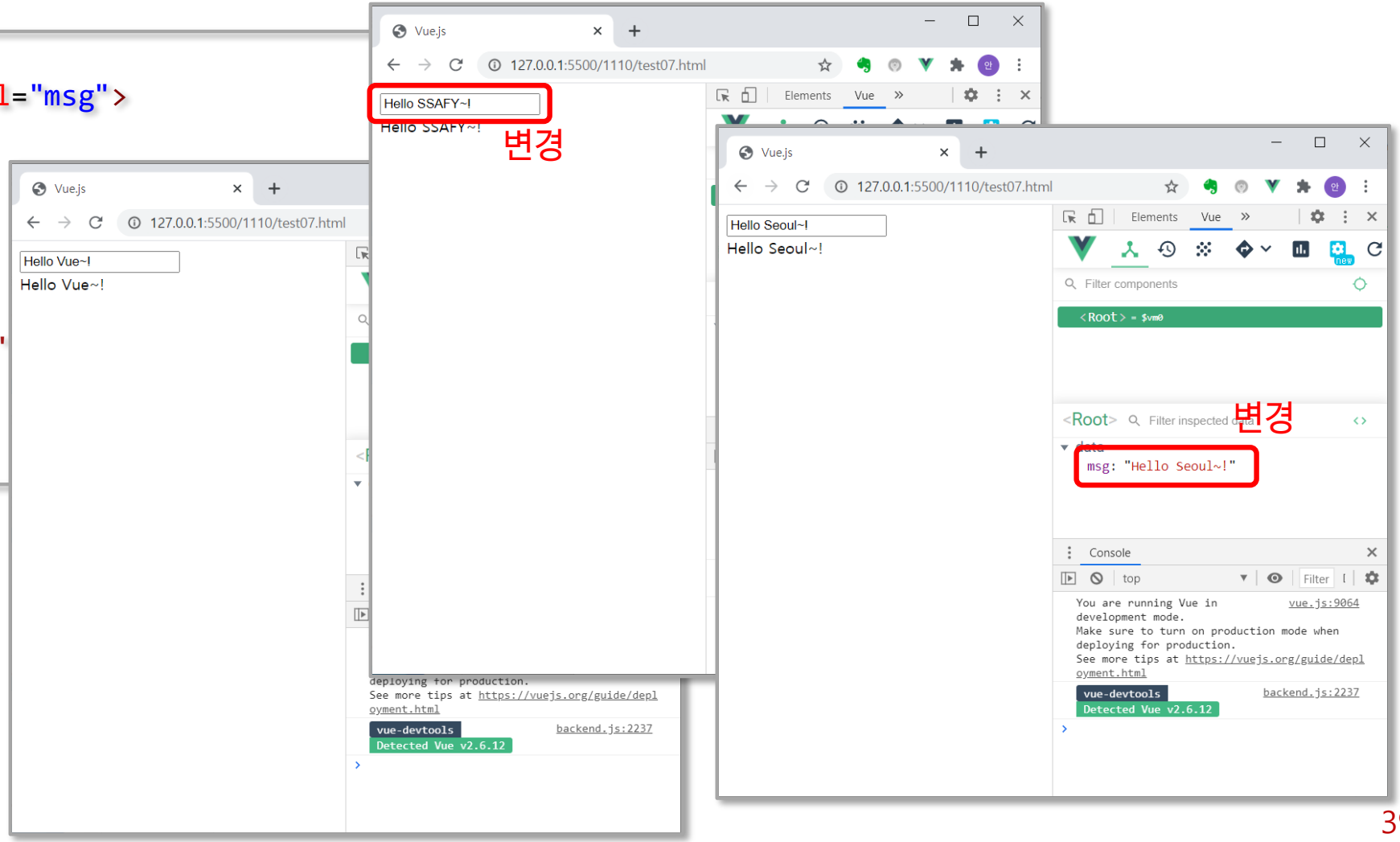
• v-text	• v-bind	• v-else
• v-html	• v-show	• v-for
• v-once	• v-if	• v-cloak
• v-model	• v-else-if	• v-on

✓ v-model.

- 양방향 바인딩 처리를 위해서 사용 (form의 input, textarea).

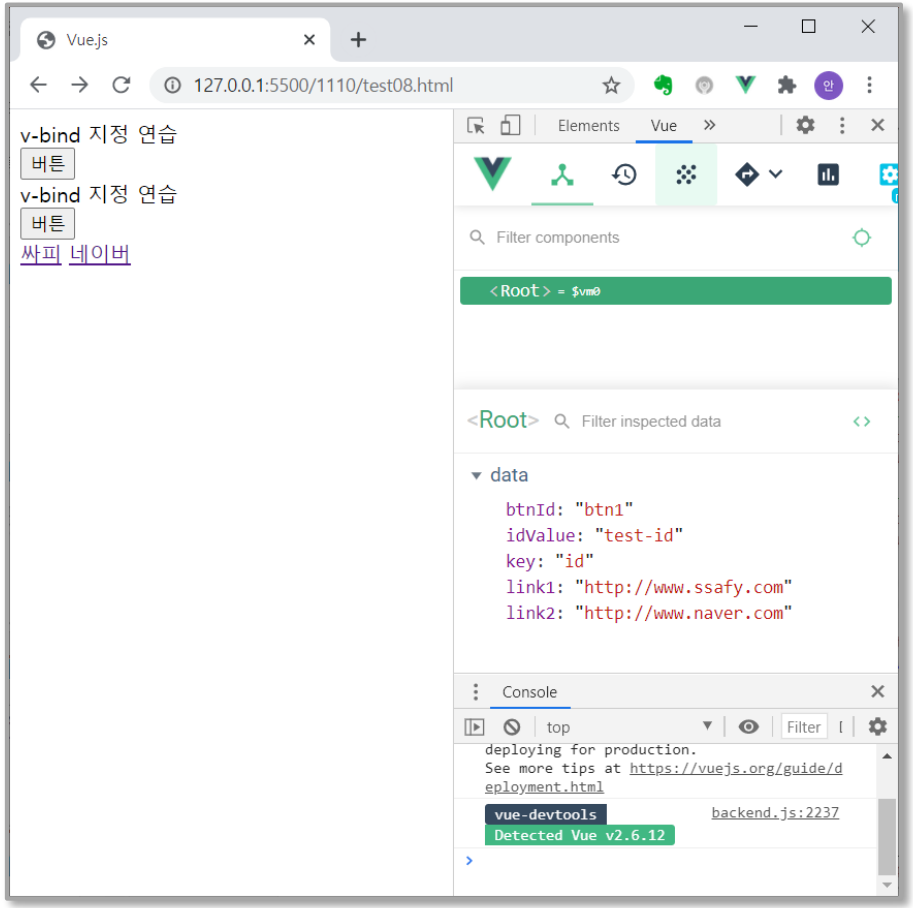
```
<div id="app">
    <input type="text" v-model="msg">
    <div>{{msg}}</div>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            msg: 'Hello Vue~!'
        }
    });
</script>
```

test07.html



✓ v-bind.

- 엘리먼트의 속성과 바인딩 처리를 위해서 사용.
- v-bind는 약어로 “:”로 사용 가능.



test08.html

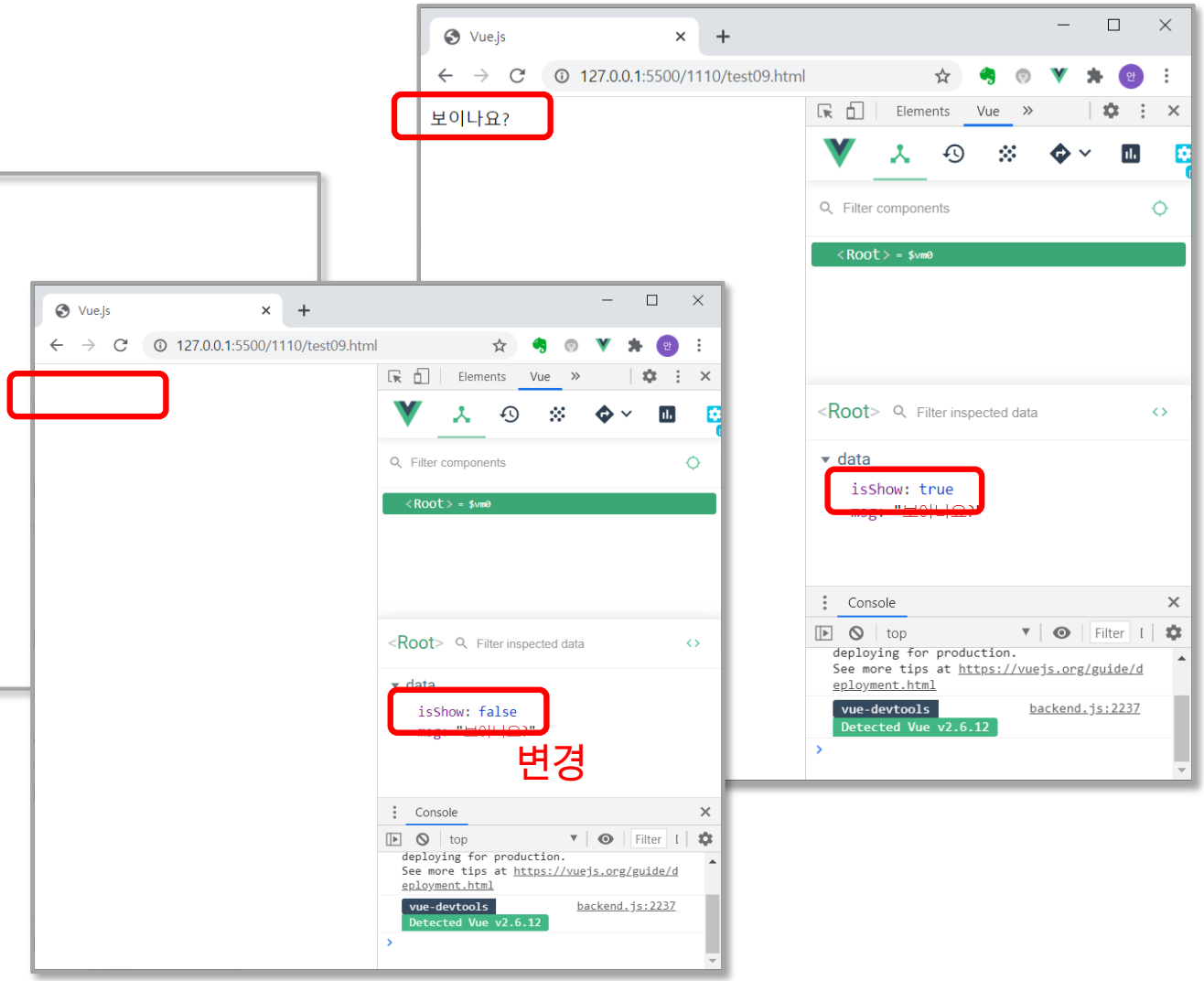
```
<div id="app">
    <!-- 속성을 바인딩 합니다. -->
    <div v-bind:id="idValue">v-bind 지정 연습</div>
    <button v-bind:[key]="btnId">버튼</button>

    <!-- 약어를 이용한 바인딩. -->
    <div :id="idValue">v-bind 지정 연습</div>
    <button :[key]="btnId">버튼</button><br>

    <a v-bind:href="link1">싸피</a>
    <a :href="link2">네이버</a>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            idValue: 'test-id',
            key: 'id',
            btnId: 'btn1',
            link1: 'http://www.ssafy.com',
            link2: 'http://www.naver.com'
        }
    });
</script>
```

✓ v-show.

- 조건에 따라 엘리먼트를 화면에 렌더링.
- style의 display를 변경.

```
<div id="app">
    <div v-show="isShow">{{msg}}</div>
</div>
<script>
    new Vue({
        el: '#app',
        data: {
            isShow: true,
            msg: '보이나요?'
        }
    });
</script>
```

test09.html

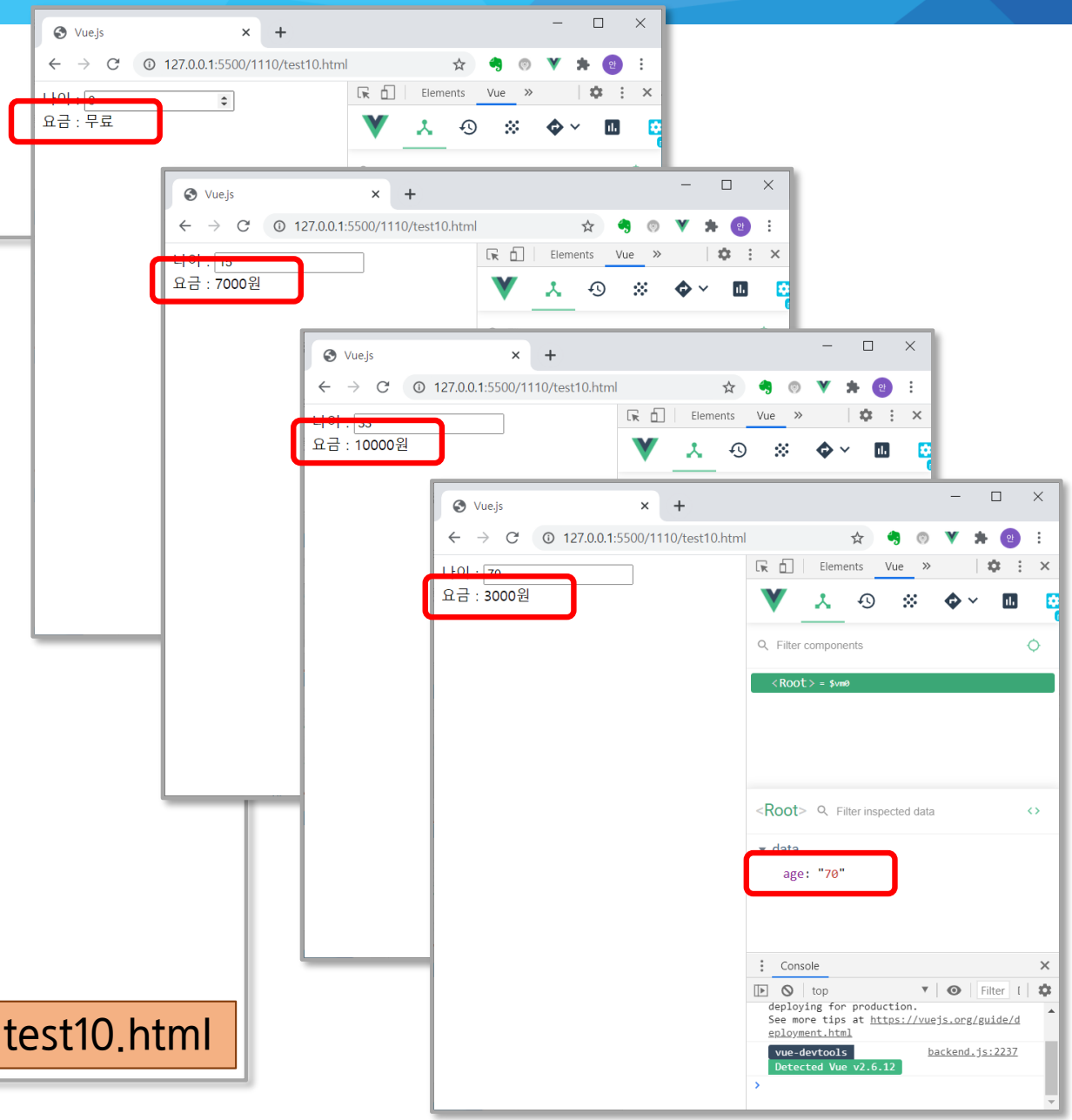


✓ v-if, v-else-if, v-else.

- 조건에 따라 엘리먼트를 화면에 렌더링.

```
<div id="app">
    <div>
        <span>나이 : </span>
        <input type="number" v-model="age">
    </div>
    <div>
        요금 :
        <span v-if="age < 10">무료</span>
        <span v-else-if="age < 20">7000원</span>
        <span v-else-if="age < 65">10000원</span>
        <span v-else>3000원</span>
    </div>
</div>
<script>
    const vm = new Vue({
        el: "#app",
        data: {
            age: '0'
        }
    });
</script>
```

test10.html



✓ v-show VS v-if의 차이점.

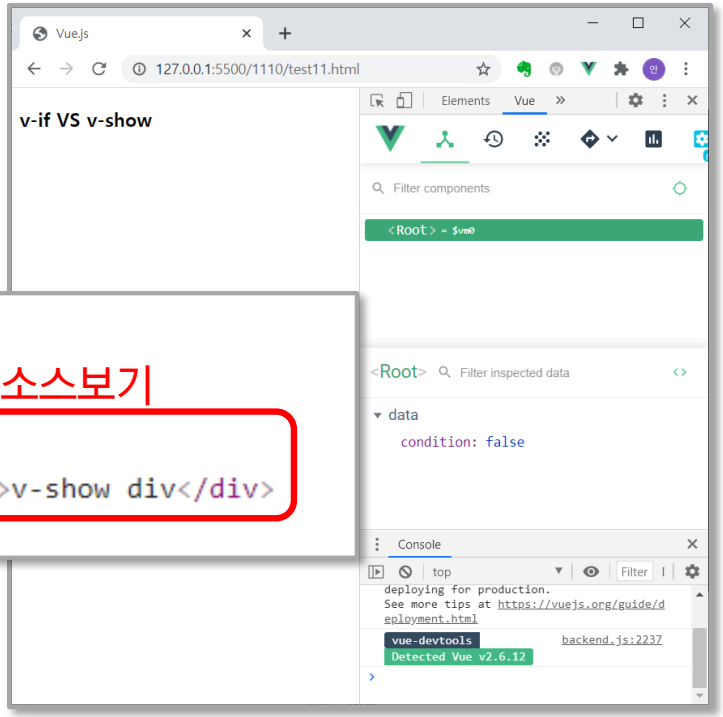
	v-if	v-show
렌더링	false일 경우 X	항상 O
false일 경우	엘리먼트 삭제	display:none 적용
template 지원	O	X
v-else 지원	O	X

```
<div id="app">
    <div>
        <h3>v-if VS v-show</h3>
        <div v-if="condition">v-if div</div>
        <div v-show="condition">v-show div</div>
    </div>
</div>
<script>
    const vm = new Vue({
        el: "#app",
        data: {
            condition: false
        }
    });
</script>
```

test11.html

```
▼<div id="app">
  ▼<div>
      <h3>v-if VS v-show</h3>
      <!---->
      <div style="display: none;">v-show div</div>
    </div>
```

소스보기



✓ v-for.

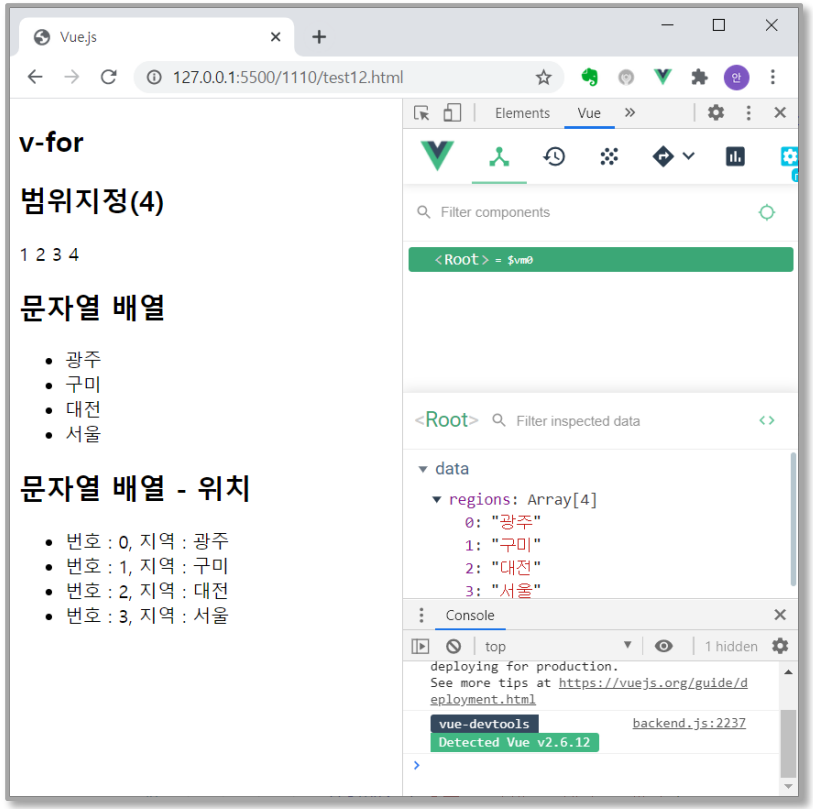
- 배열이나 객체의 반복에 사용.
- v-for="요소변수이름 in 배열" v-for="(요소변수이름, 인덱스) in 배열 "

```
<h2>v-for</h2>
<div id="app">
    <h2>문자열 배열</h2>
    <ul>
        <li v-for="name in regions">
            {{name}}
        </li>
    </ul>
</div>
<script>
    new Vue({
        el: "#app",
        data: {
            regions: ["광주", "구미", "대전", "서울"]
        }
    })
</script>
```

test12.html

test13.html

test14.html

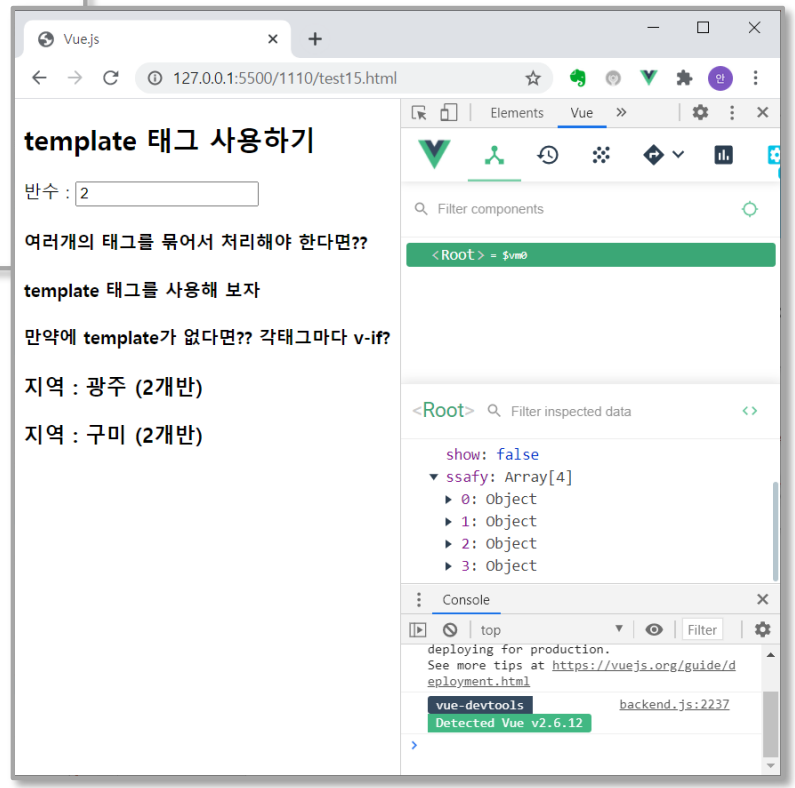


✓ template.

- 여러 개의 태그들을 묶어서 처리해야 할 경우 template를 사용.
- v-if, v-for, component등과 함께 많이 사용.

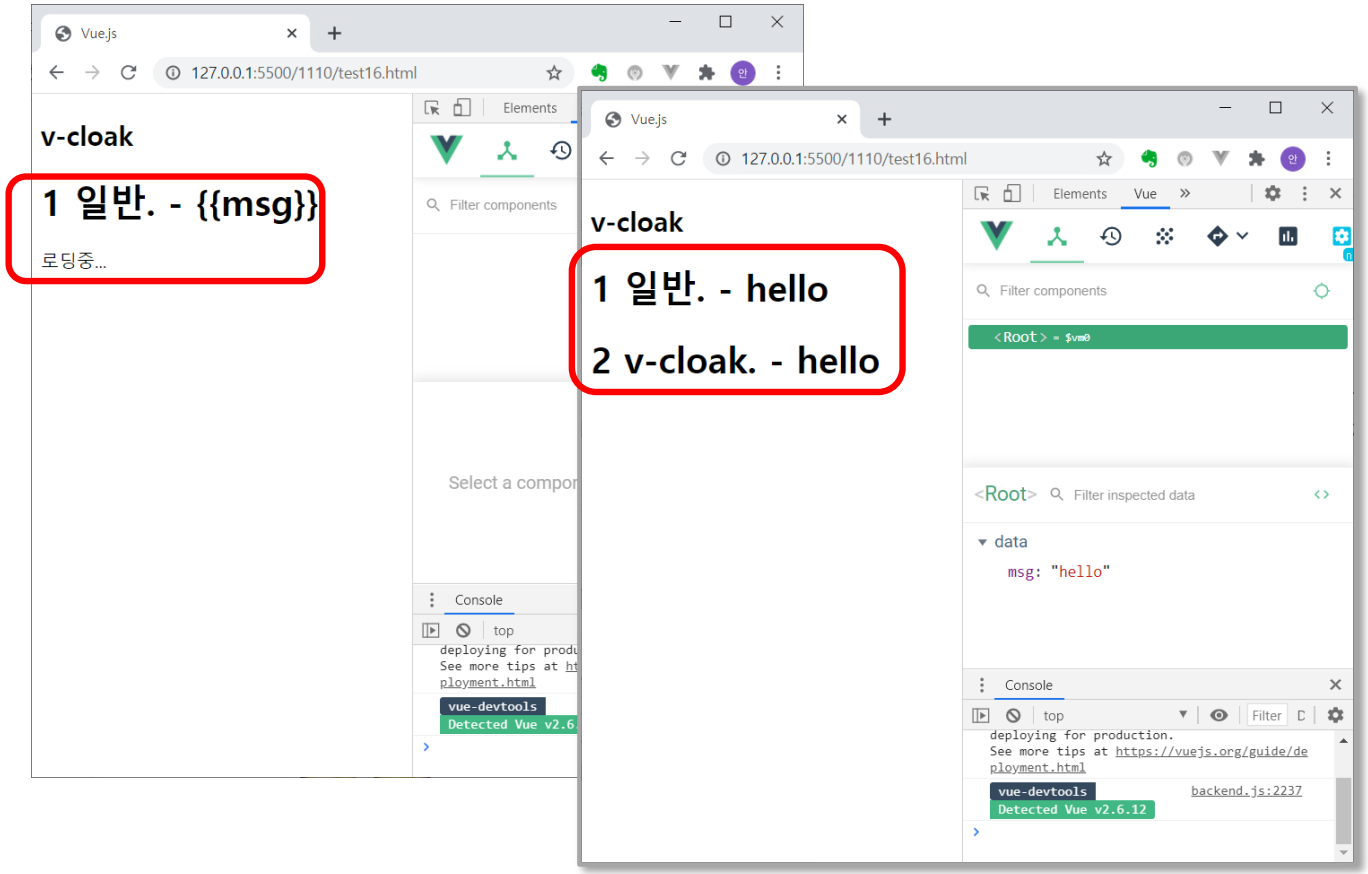
test15.html

```
<template v-if="count % 2 == 0">
    <h4>여러개의 태그를 묶어서 처리해야 한다면??</h4>
    <h4>template 태그를 사용해 보자</h4>
    <h4>만약에 template가 없다면?? 각태그마다 v-if?</h4>
</template>
<template v-for="(region, index) in ssafy" v-if="region.count === count">
    <h3>지역 : {{region.name}} ({{region.count}}개반)</h3>
</template>
```



✓ v-cloak.

- Vue Instance가 준비될 때까지 mustache 바인딩을 숨기는데 사용.
- [v-cloak] { display: none }과 같은 CSS 규칙과 함께 사용.
- Vue Instance가 준비되면 v-cloak는 제거됨.



test16.html

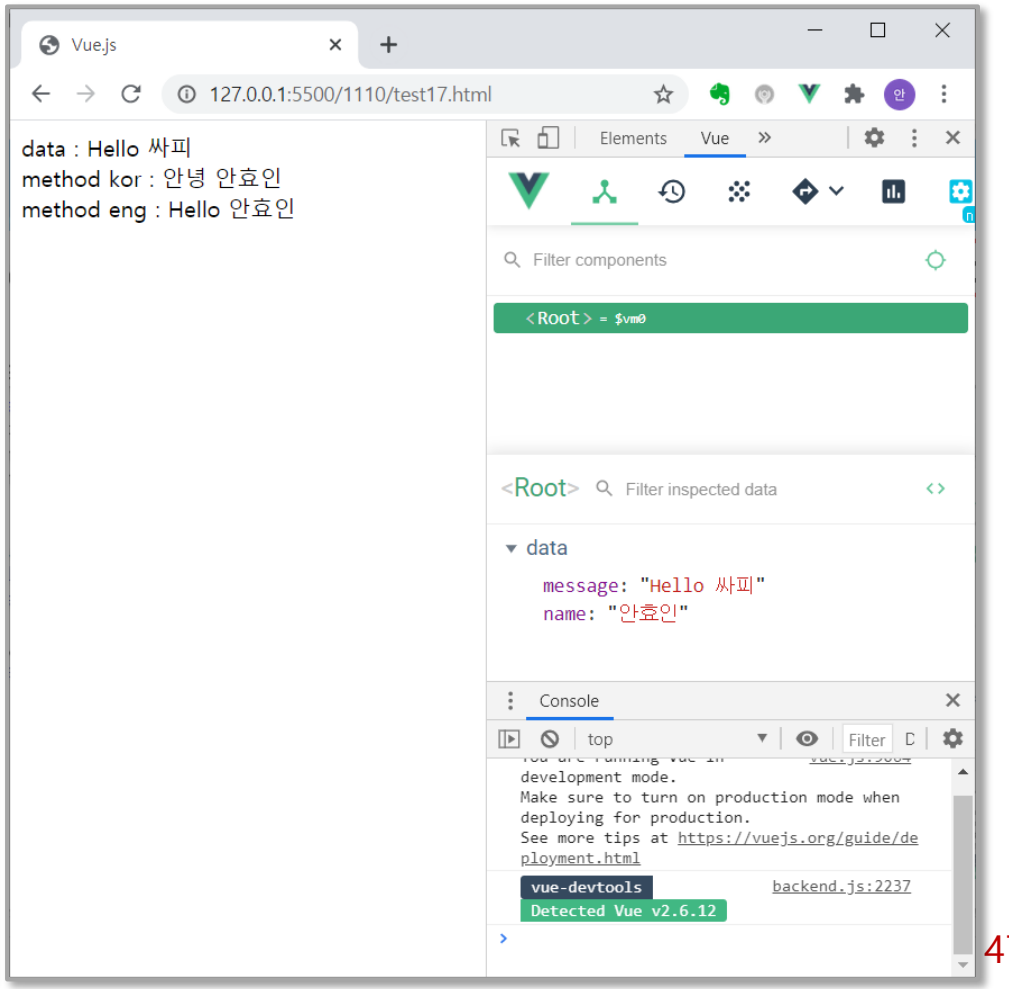
```
<style>
    [v-cloak]::before {
        content: '로딩중...'
    }
    [v-cloak] > * {
        display: none;
    }
</style>

<h2>v-cloak</h2>
<div id="app">
    <h1>1 일반. - {{msg}}</h1>
    <div v-cloak>
        <h1>2 v-cloak. - {{msg}}</h1>
    </div>
</div>
<script>
    setTimeout(function () {
        new Vue({
            el: "#app",
            data: {
                msg: "hello"
            }
        })
    }, 3000);
</script>
```

✓ Vue method.

- Vue Instance는 생성과 관련된 data 및 method의 정의 가능.
- method안에서 data를 "this.데이터이름" 으로 접근 가능.

```
<div id="app">
    <div>data : {{message}}</div>
    <div>method kor : {{helloKor()}}</div>
    <div>method eng : {{helloEng()}}</div>
</div>
<script>
    new Vue({
        el: "#app",
        data: {
            message: "Hello 싸피", name: "안효인"
        },
        methods: {
            helloEng() {
                return "Hello " + this.name
            },
            helloKor() {
                return "안녕 " + this.name
            }
        }
    })
</script>
```

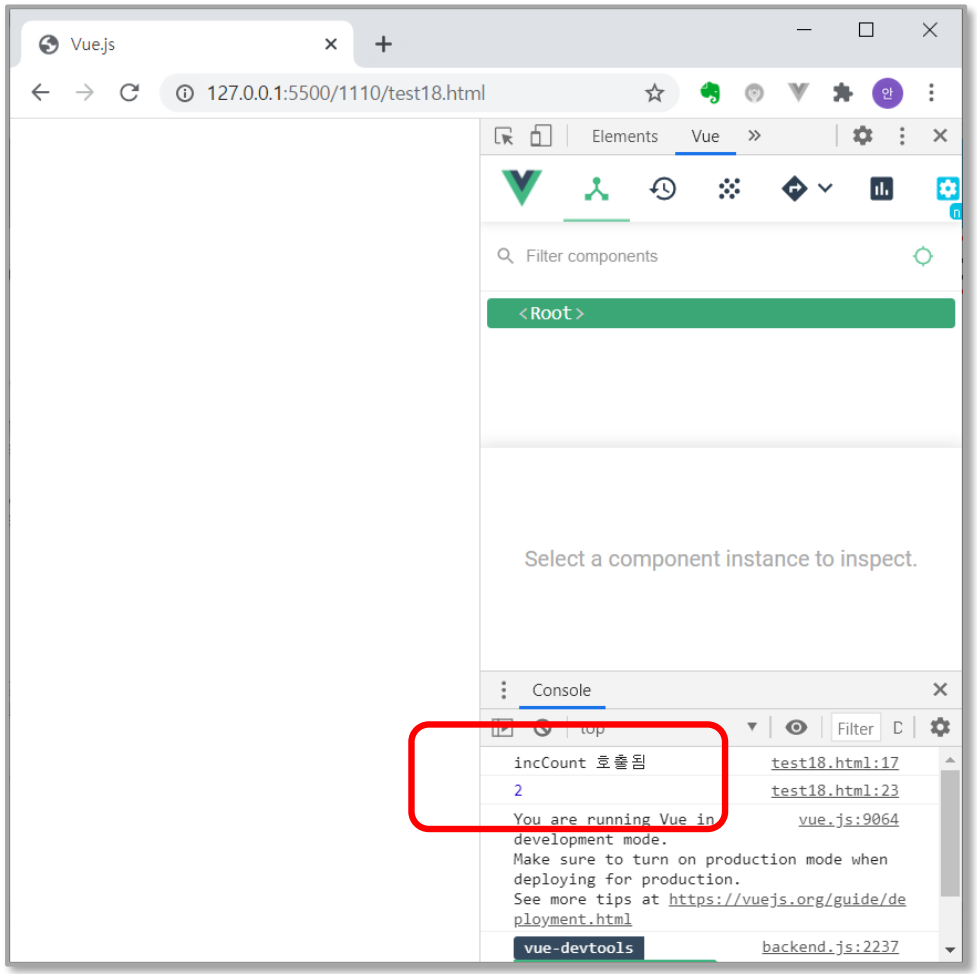
test17.html



✓ Vue method.

```
<script>
    let vm = new Vue({
        data: {
            count: 1
        },
        methods: {
            incCount() {
                console.log("incCount 호출됨");
                this.count++
            }
        }
    })
    vm.incCount();
    console.log(vm.count); // 2
</script>
```

test18.html



SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

내일 방송에서 만나요!

삼성 청년 SW 아카데미