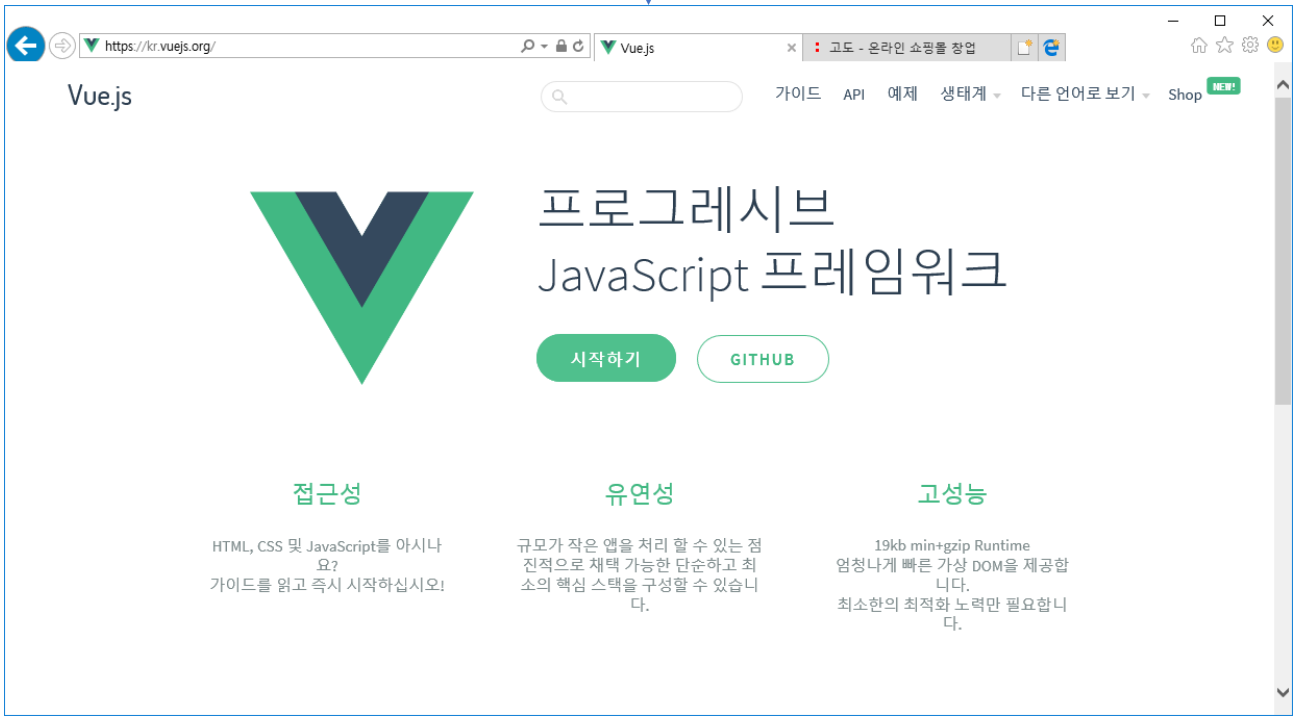
지니의 vueJS 스터디 시작편

2017. 07. 05 이호진 infohojin@naver.com

vueJS 공식 홈페이지 : https://vuejs.org/

한국어 사이트 : https://kr.vuejs.org/



vueJS 스크립트

Vue.js가 무엇인가요?

Vue(/vju:/ 로 발음, **view** 와 발음이 같습니다.)는 사용자 인터페이스를 만들기 위한 **진보적인 프레임워크** 입니다. 다른 단일형 프레임워크와 달리 Vue는 점진적으로 채택할 수 있도록 설계하였습니다. 핵심 라이브러리는 뷰 레이어만 초점을 맞추어 다른 라이브러리나 기존 프로젝트와의 통합이 매우 쉽습니다. 그리고 Vue는 **현대적 도구** 및 **지원하는 라이브러리**와 함께 사용한다면 정교한 단일 페이지 응용프로그램을 완벽하게 지원할 수 있습니다.

vueJS 스크립트

vueJS 는 간단하게 웹사이트에 <script> 스크립트 테그를 추가함으로서 간단하게 시작 및 사용을 할 수 있습니다.

```HTML
<script src="https://unpkg.com/vue"></script>
```

Vue.js의 핵심은 간단한 템플릿 구문을 사용해 선언적으로 DOM에 데이터를 렌더링하는 것입니다.

호환성 정보

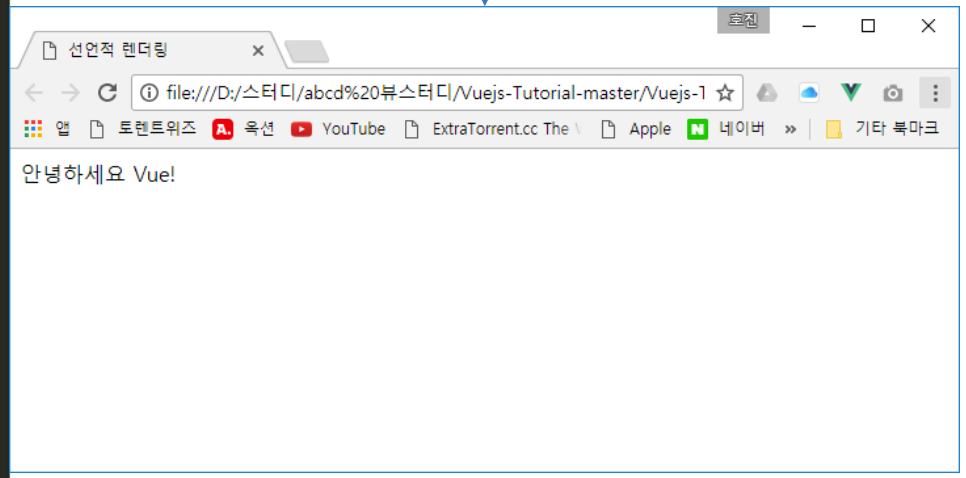
Vue는 ECMAScript 5 기능을 사용하기 때문에 IE8 이하 버전을 **지원하지 않습니다.**
하지만 모든 **ECMAScript 5 호환 브라우저**를 지원합니다

Hello world 출력

```
1  <!DOCTYPE html>
2  <html lang="ko">
3      <head>
4          <meta charset="UTF-8">
5          <title>선언적 렌더링</title>
6      </head>
7      <body>
8          <div id="app">
9              {{ message }}
10         </div>
11
12         <script src="https://unpkg.com/vue"></script>
13
14         <script>
15             var app = new Vue({
16                 el: '#app',
17                 data: {
18                     message: '안녕하세요 Vue!'
19                 }
20             })
21         </script>
22     </body>
23 </html>
24
```

① ② ③

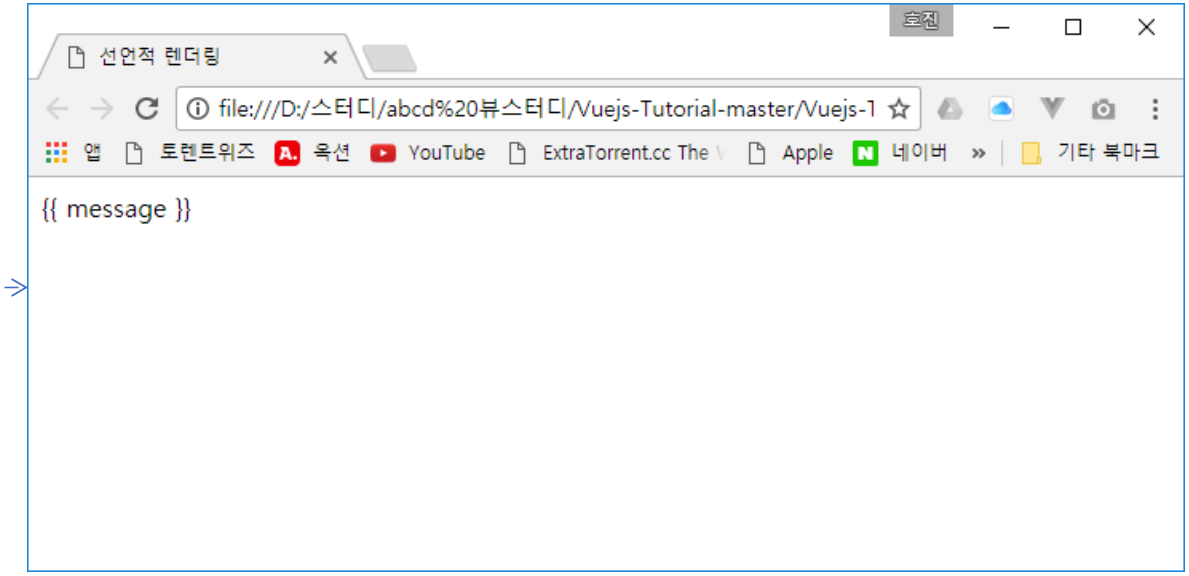
vueJS 를 이용하여 화면에 글자를 표시



DOM 처리

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

HTML에서 DIV 태그로 작성된 내용 {{ message }} 이 화면이 출력될 것입니다.



DOM 처리

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

<script src="https://unpkg.com/vue"></script>
```

① vueJS 스크립트를 로드합니다.

② DIV id값으로 요소를 처리합니다.

```
<script>
    var app = new Vue({
        el: '#app',
        data: {
            message: '안녕하세요 Vue!'
        }
    })
</script>
```

③ prefix 코드 처리를 통하여 메시지 내용을 DOM 치환 처리 합니다.

반응형 DOM

DOM 과 데이터가 연결이 되어 있어 스크립트에서 데이터를 변경하면 즉시 데이터가 반등이 됩니다.

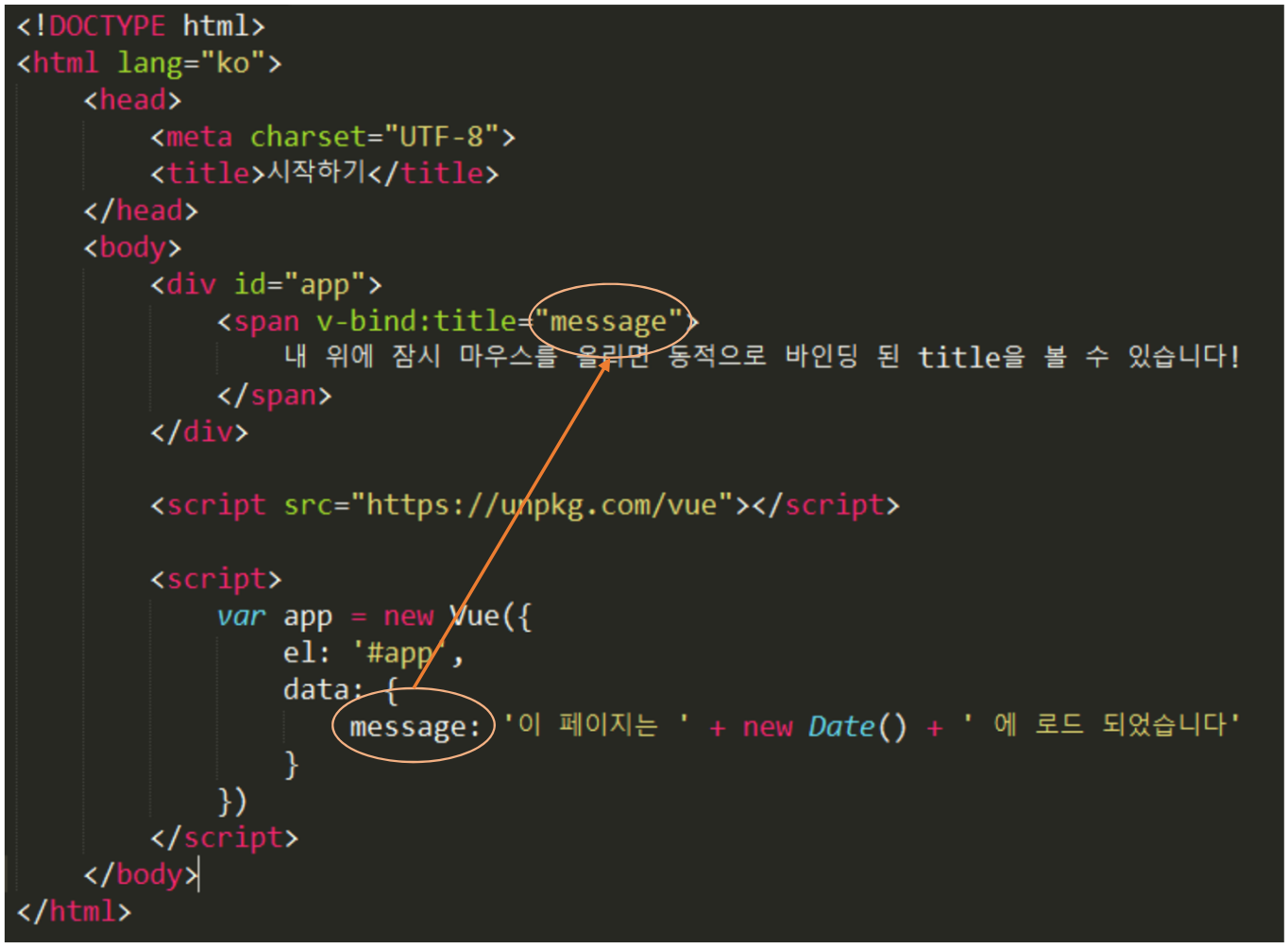
DOM의 id값과 속성을 통하여 화면출력을 제어를 하실 수 있습니다.

app 객체변수를 생성후에 메서드에 접근하여 데이터를 바로 수정할 수도 있습니다.

app.message = 'hello world!';

이처럼 화면을 갱신하지 않고 동적으로 DOM을 접근하여 데이터를 변경 및 처리가 가능합니다.

바인딩



```
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="UTF-8">
        <title>시작하기</title>
    </head>
    <body>
        <div id="app">
            <span v-bind:title="message">
                내 위에 잠시 마우스를 올리면 동적으로 바인딩 된 title을 볼 수 있습니다!
            </span>
        </div>

        <script src="https://unpkg.com/vue"></script>

        <script>
            var app = new Vue({
                el: '#app',
                data: {
                    message: '이 페이지는 ' + new Date() + ' 에 로드 되었습니다'
                }
            })
        </script>
    </body>
</html>
```

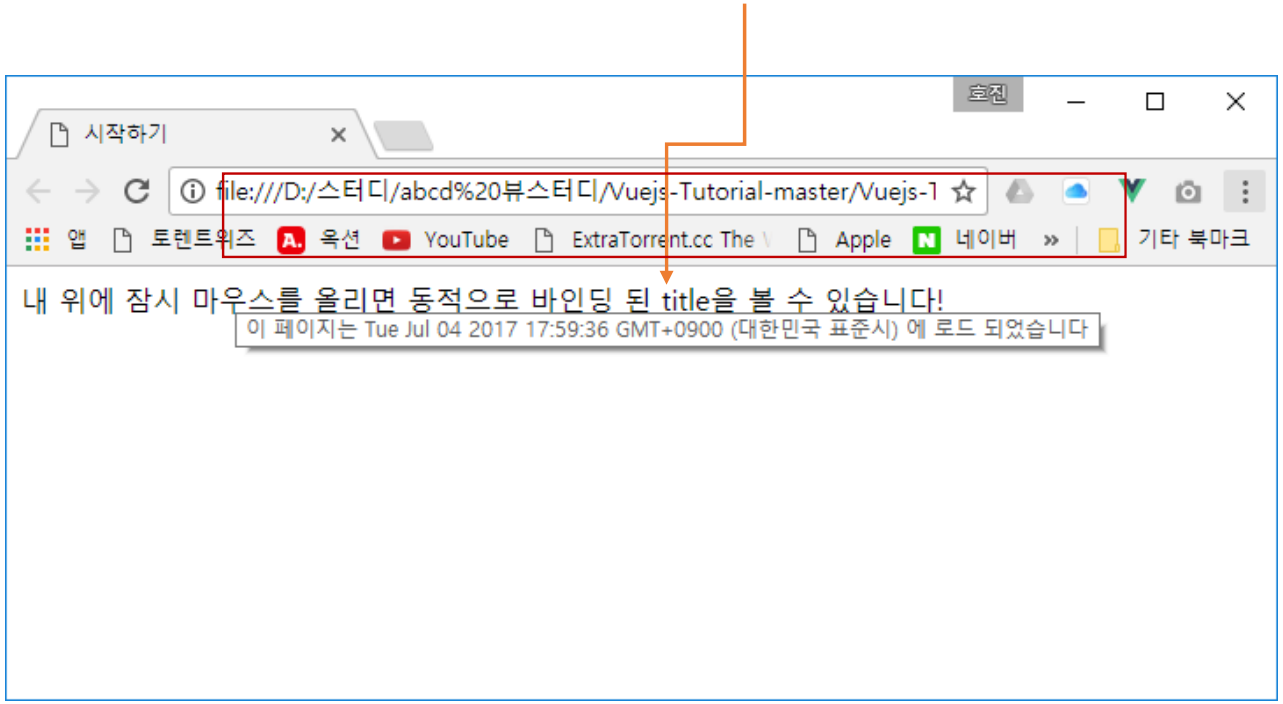
엘리먼트 속성을 바인딩할 수 있습니다.

Vue는 ***디렉티브*** 라고 하여 속성을 제어할 수 있는 전용 키워드 를 제공합니다.

v-bind: 접두어로 시작하는 특수 속성을 제공합니다.

타이틀 바인딩

앞 전의 예제를 실행후에 마우스를 올려놓으면 실시간으로 날짜가 표시된 타이틀 마크를 확인할 수 있습니다.

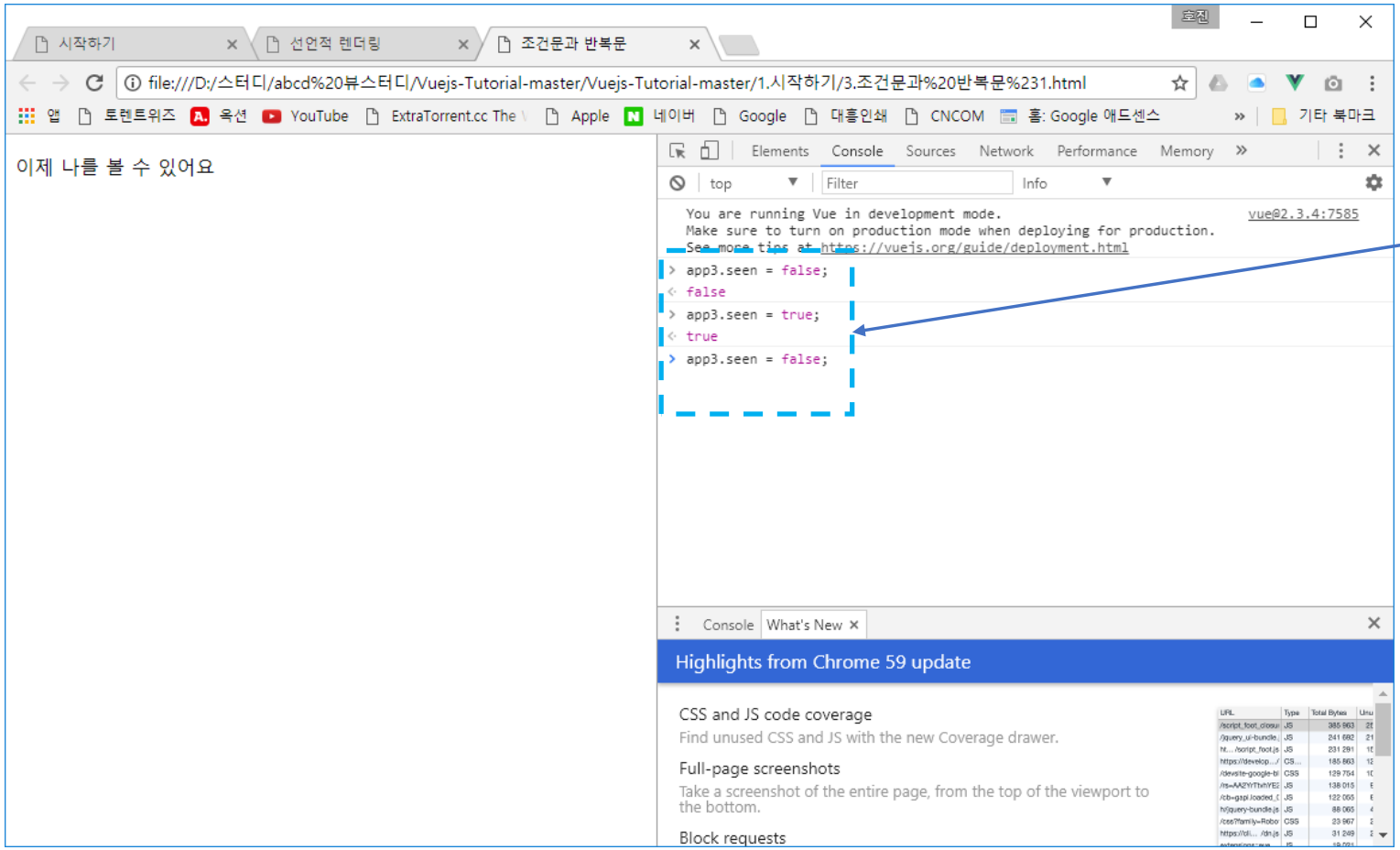


v-if 조건문

```
<div id="app">
    <p v-if="seen">이제 나를 볼 수 있어요</p>
</div>

<script src="https://unpkg.com/vue"></script>

<script>
    var app3 = new Vue({
        el: '#app',
        data: {
            seen: true
        }
    })
</script>
```

텍스트와 속성뿐 아니라 DOM의 **구조**에도 데이터를 바인딩 할 수 있음을 보여줍니다.

v-if 조건문

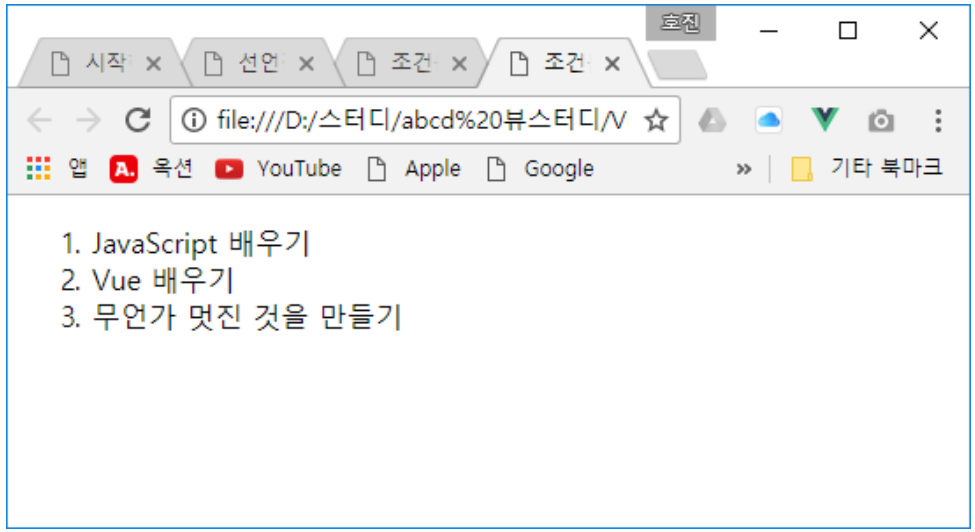


스크립트에서 값을 변경하여 v-if 조건에 따라서 데이터 출력 여부를 제어할 수 있습니다.

v-for 반복문

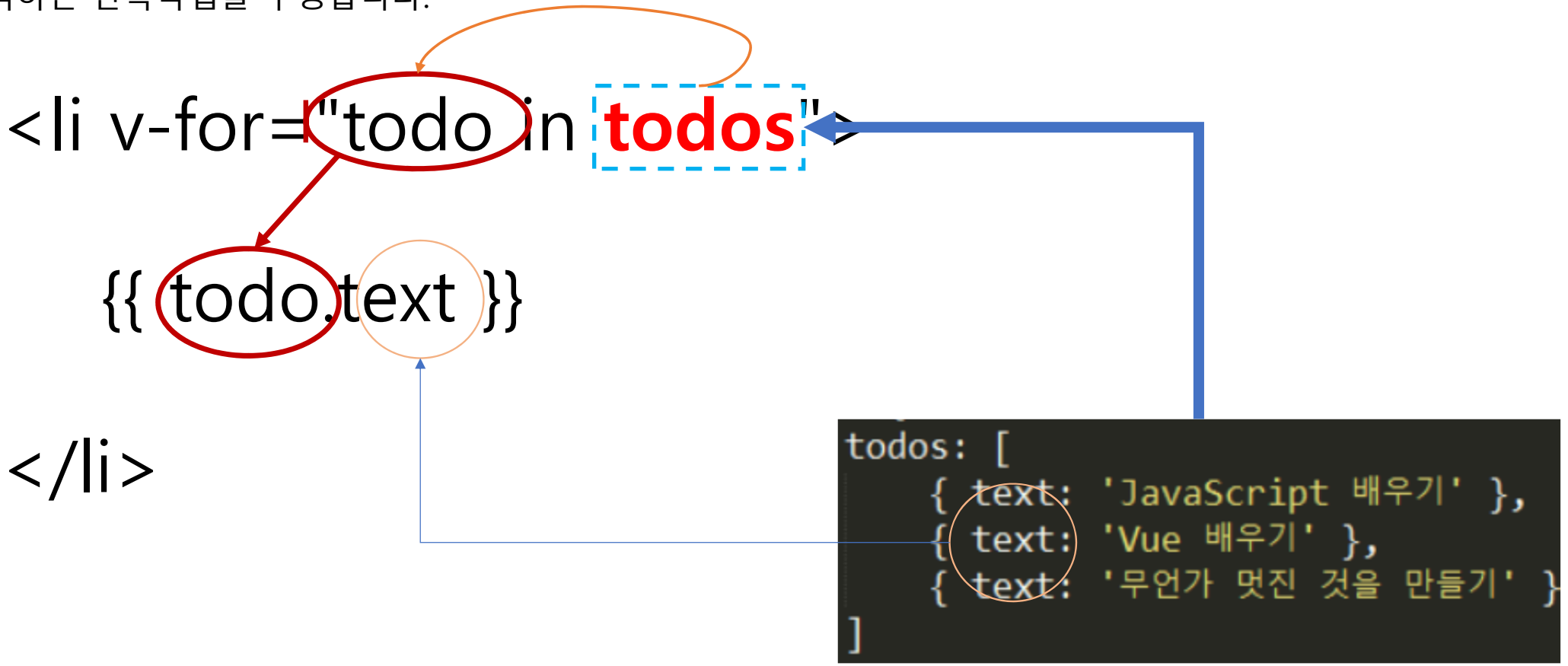
```
<div id="app">
    <ol>
        <li v-for="todo in todos">
            {{ todo.text }}
        </li>
    </ol>
</div>

<script src="https://unpkg.com/vue"></script>

<script>
    var app = new Vue({
        el: '#app',
        data: {
            todos: [
                { text: 'JavaScript 배우기' },
                { text: 'Vue 배우기' },
                { text: '무언가 멋진 것을 만들기' }
            ]
        }
    })
</script>
```

v-for 바이딩 디렉티브는 반복처리를 통하여 데이터 바인딩을 할 수 있습니다.



v-for 반복문

반복문은 foreach 처럼 배열형태의 데이터를 가지고 와서 데이터를 출력하는 반복작업을 수행합니다.



사용자 입력처리

```
<div id="app-5">
    <p>{{ message }}</p>
    <button v-on:click="reverseMessage">메시지 뒤집기</button>
</div>

<script src="https://unpkg.com/vue"></script>

<script>
    var app5 = new Vue({
        el: '#app-5',
        data: {
            message: '안녕하세요! Vue.js!'
        },
        methods: {
            reverseMessage: function () {
                this.message = this.message.split('').reverse().join('')
            }
        }
    })
</script>
```

v-on: 과 같은 디렉티브는 사용자의 동작입력을 처리할 수 입니니다.

<button **v-on:**click="reverseMessage">메시지 뒤집기</button>

Click 시 정의된 reverseMessage 매소드 함수를 실행합니다.

```
methods: {
    reverseMessage: function () {
        this.message = this.message.split('').reverse().join('')
    }
}
```

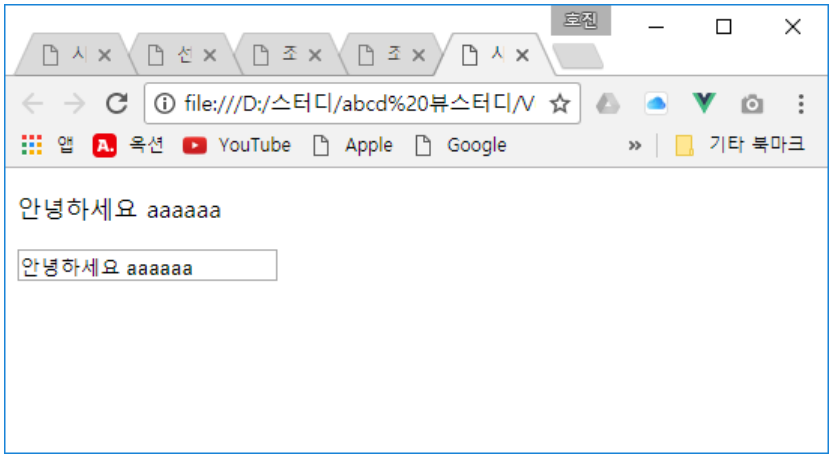
클릭시 문자열을 분석하여 반대로 뒤집기 하는 셈플 예제입니다.

v-model 사용자입력

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

<script src="https://unpkg.com/vue"></script>

<script>
    var app = new Vue({
        el: '#app',
        data: {
            message: '안녕하세요 Vue!'
        }
    })
</script>
```

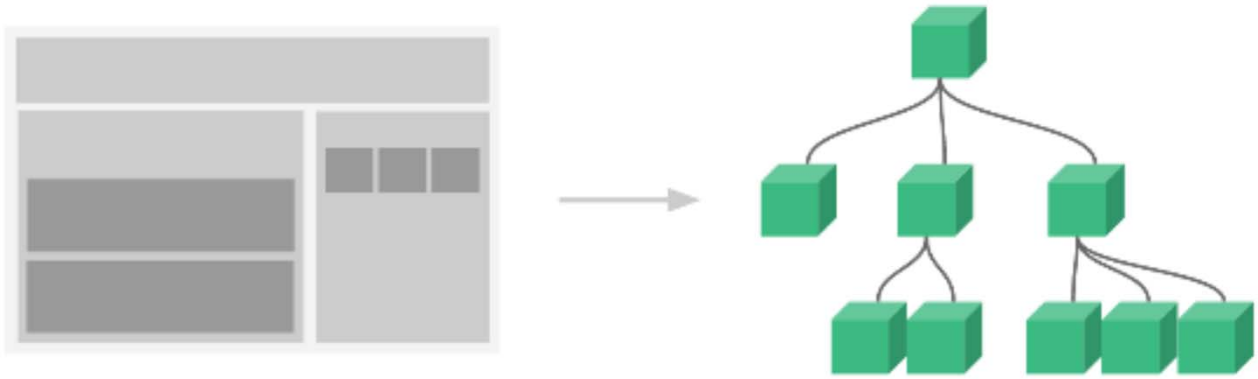
v-model 과 같은 디렉티브는 form input 사용자 입력과 같은 처리를 할 수 있습니다.

또한 양방향으로 데이터를 바인딩으로 되어 있어 실시간으로 데이터를 처리 할 수 있습니다.



컴포넌트를 사용한 작성방법

컴포넌트 시스템은 Vue의 또 다른 중요한 개념입니다. 이는 작고 그 자체로 제 기능을 하며 재사용할 수 있는 컴포넌트로 구성된 대규모 응용 프로그램을 구축할 수 있게 해주는 추상적 개념입니다. 생각해보면 거의 모든 유형의 응용 프로그램 인터페이스를 컴포넌트 트리로 추상화할 수 있습니다.



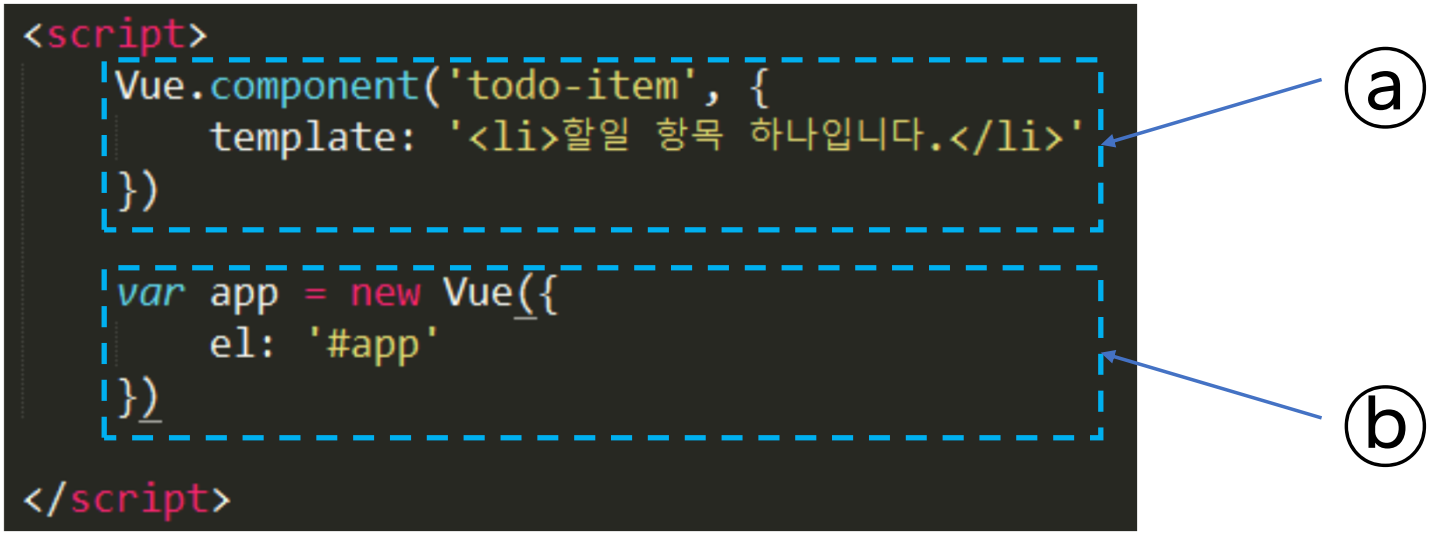
① 컴포넌트 정의

Vue에서 컴포넌트를 사용하는 방법은 매우 간단합니다. 아래 예제처럼 Vue에서, 컴포넌트를 정의합니다.

```JS
// todo-item 이름을 가진 컴포넌트를 정의합니다
Vue.component('todo-item', {
  template: '<li>할일 항목 하나입니다.</li>'
})
```

vueJS는 미리 정의된 Vue 인스턴스를 가지고 있습니다.

컴포넌트 이름은 *'todo-item'*입니다. 또한 이 컴포넌트는 하나의 *template*를 가지고 있습니다.


<script>
Vue.component('todo-item', {
template: '
할일 항목 하나입니다.
'
})
var app = new Vue({
el: '#app'
})
</script>
ⓐ
ⓑ

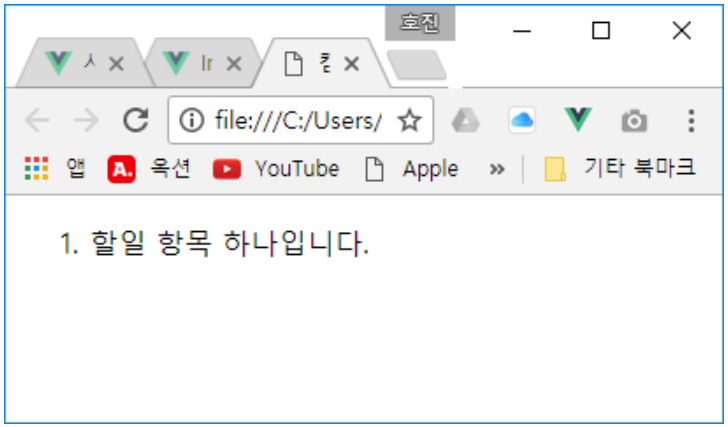
② 컴포넌트 사용

HTML 코드상에서 정의한 컴포넌트 이름으로 테그를 삽입합니다.

```HTML
<ol>
  <!-- todo-item 컴포넌트의 인스턴스 만들기 -->
  <todo-item></todo-item>
</ol>
```

ⓑ

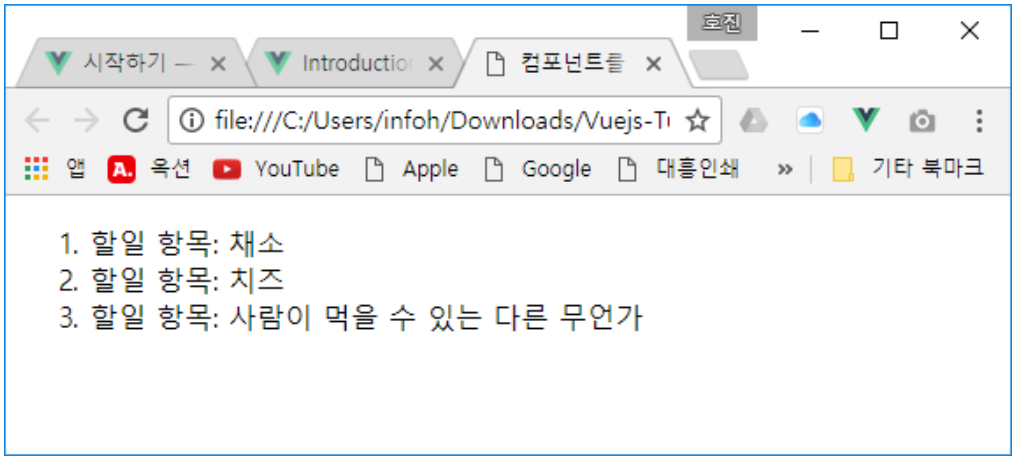
```
<div id="app">
    <ol>
        <todo-item></todo-item>
    </ol>
</div>
```



③ Prop 설정 및 데이터반복

```
<div id="app">
    <ol>
      <todo-item v-for="item in groceryList" v-bind:todo="item"></todo-item>
    </ol>

</div>
```

```
<script>
    Vue.component('todo-item', {
        props: ['todo'],
        template: '<li>할일 항목: {{ todo.text }}</li>'
    })

    var app = new Vue({
        el: '#app',
        data: {
            groceryList: [
                { text: '채소' },
                { text: '치즈' },
                { text: '사람이 먹을 수 있는 다른 무언가' }
            ]
        }
    })

</script>
```


호진
시작하기 —
Introductio
컴포넌트를
file:///C:/Users/infoh/Downloads/Vuejs-T
앱
옥션
YouTube
Apple
Google
대흥인쇄
기타 북마크
1. 할일 항목: 채소
2. 할일 항목: 치즈
3. 할일 항목: 사람이 먹을 수 있는 다른 무언가

```html
<div id="app">
    <ol>
        <todo-item v-for="item in groceryList" v-bind:todo="item"></todo-item>
    </ol>

    <ol>
        <todo-item v-for="item in aaa" v-bind:todo="item"></todo-item>
    </ol>
</div>

<script>
    Vue.component('todo-item', {
        props: ['todo'],
        template: '<li>할일 항목: {{ todo.text }}</li>'
    })

    var app = new Vue({
        el: '#app',
        data: {
            groceryList: [
                { text: '채소' },
                { text: '치즈' },
                { text: '사람이 먹을 수 있는 다른 무언가' }
            ],
            aaa: [
                { text: '채소1' },
                { text: '치즈2' },
                { text: '사람3' }
            ]
        }
    })

</script>
```