

원쌤의 Vue.js 퀵스타트

3. Vue.js 기초와 Template



1. 보안법

❖ 1장의 예제 01-01 리뷰 : 예제 03-01

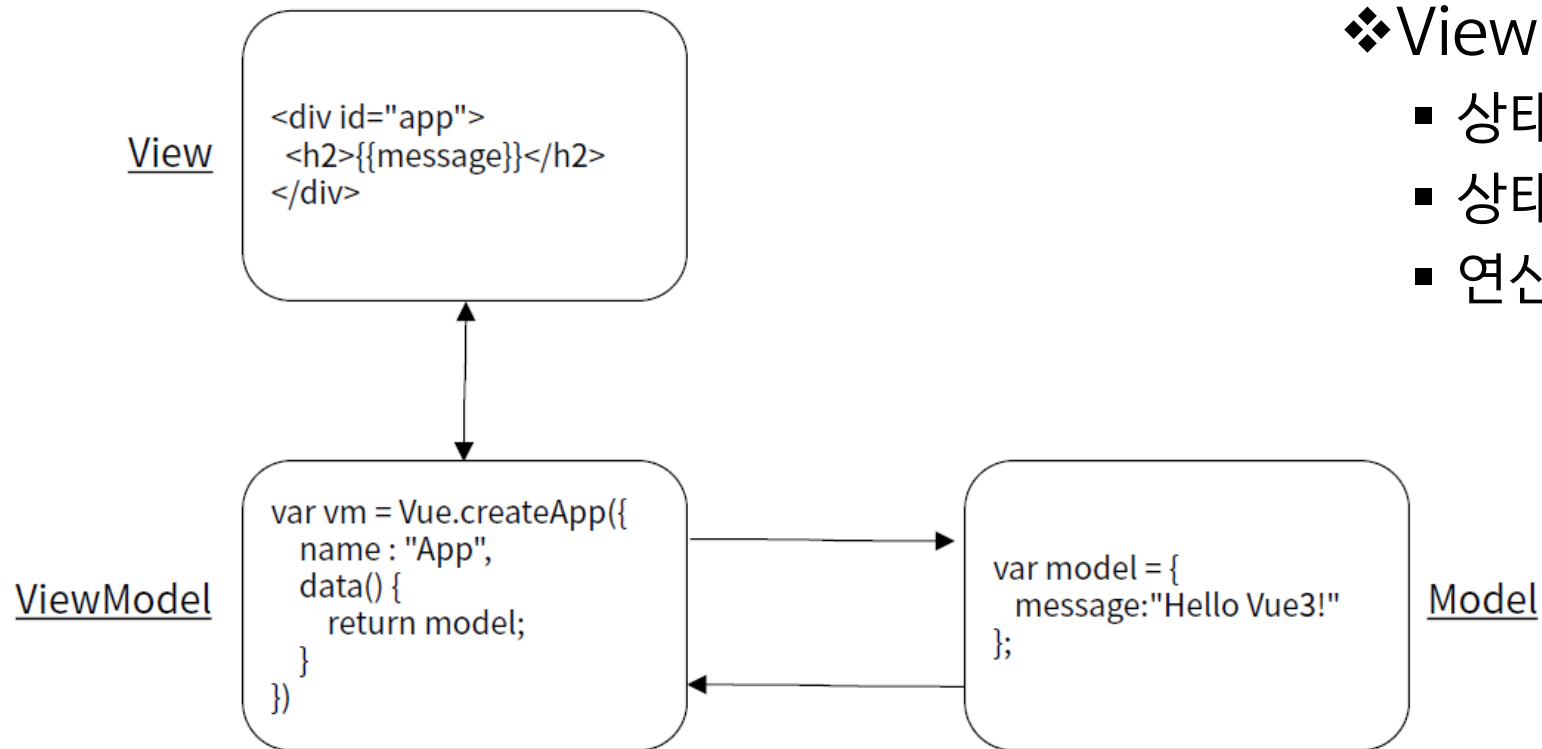
```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6      <title>03-01</title>
7    </head>
8    <body>
9      <div id="app">
10       <h2>{{message}}</h2>
11     </div>
12     <script type="text/javascript" src="https://unpkg.com/vue"></script>
13     <script type="text/javascript">
14       var model = { message: "Hello Vue3!" };
15       var vm = Vue.createApp({
16         name: "App",
17         data() {
18           return model;
19         },
20       }).mount("#app");
21     </script>
22   </body>
23 </html>
```

1. 보간법

❖보간법

- {{message}}
- {{{}} : Interpolation, 콧수염 표현식

❖MVVM 패턴과 비교



❖ViewModel

- 상태와 연산을 내부에 포함함.
- 상태 : data
- 연산 : operation, 메서드

2. 기본 디렉티브

❖ v-text

예제 03-02 : 기존 예제 03-01에 v-text 디렉티브 적용

```
09: <div id="app">
10:   <h2 v-text="message"></h2>
11: </div>
```

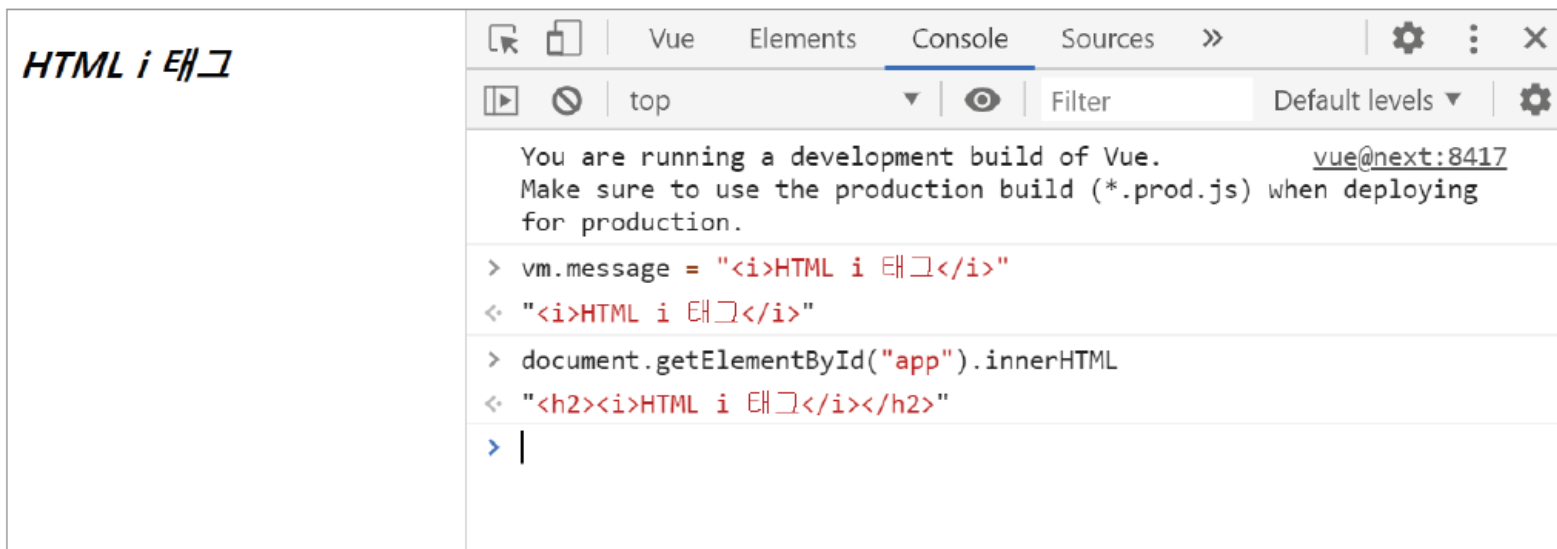
- v-text는 HTML 인코딩이 기본

<i>HTML i 태그</i>	Vue Elements Console Sources >> ⚙️ ⋮ ✕	
	▶ 🔇 top ▼ 👁 Filter Default levels ▼ ⚙️	
	You are running a development build of Vue. vue@next:8417 Make sure to use the production build (*.prod.js) when deploying for production.	
	> vm.message = "<i>HTML i 태그</i>"	< " <i>HTML i 태그</i>"
	> document.getElementById("app").innerHTML	< "<h2><i>HTML i 태그</i></h2>"
	>	

2. 기본 디렉티브

❖ v-html

- v-html로 변경했을 때



v-text, {{ }}	innerText 속성에 연결됨. 태그 문자열을 HTML 인코딩하여 나타내기 때문에 화면에는 태그 문자열이 그대로 나타남.
---------------	--

v-html	innerHTML 속성에 연결됨. 태그 문자열을 파싱하여 화면에 나타냄
--------	---

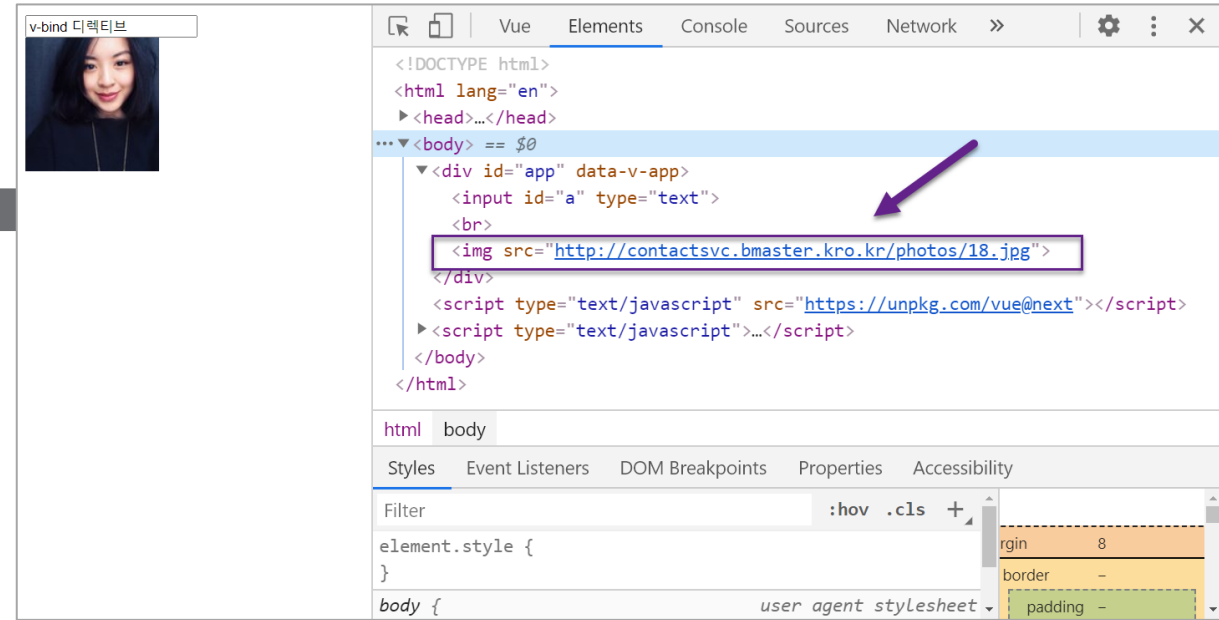
2. 기본 디렉티브

❖ v-bind 디렉티브

- 요소의 속성을 바인딩하기 위해 사용함

예제 03-03 : v-bind 디렉티브

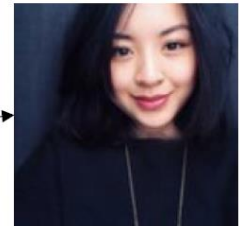
```
01: <body>
02:   <div id="app">
03:     <input id="a" type="text" v-bind:value="message">
04:     <br />
05:     
06:   </div>
07:   <script type="text/javascript" src="https://unpkg.com/vue"></script>
08:   <script type="text/javascript">
09:     var vm = Vue.createApp({
10:       name : "App",
11:       data() {
12:         return {
13:           message : "v-bind 디렉티브",
14:           imagePath : "https://contactsvc.bmaster.kro.kr/photos/18.jpg"
15:         };
16:       }
17:     }).mount('#app')
18:   </script>
19: </body>
```



Vue 인스턴스의 데이터

imagePath : "https://contactsvc.bmaster.kro.kr/photos/18.jpg"

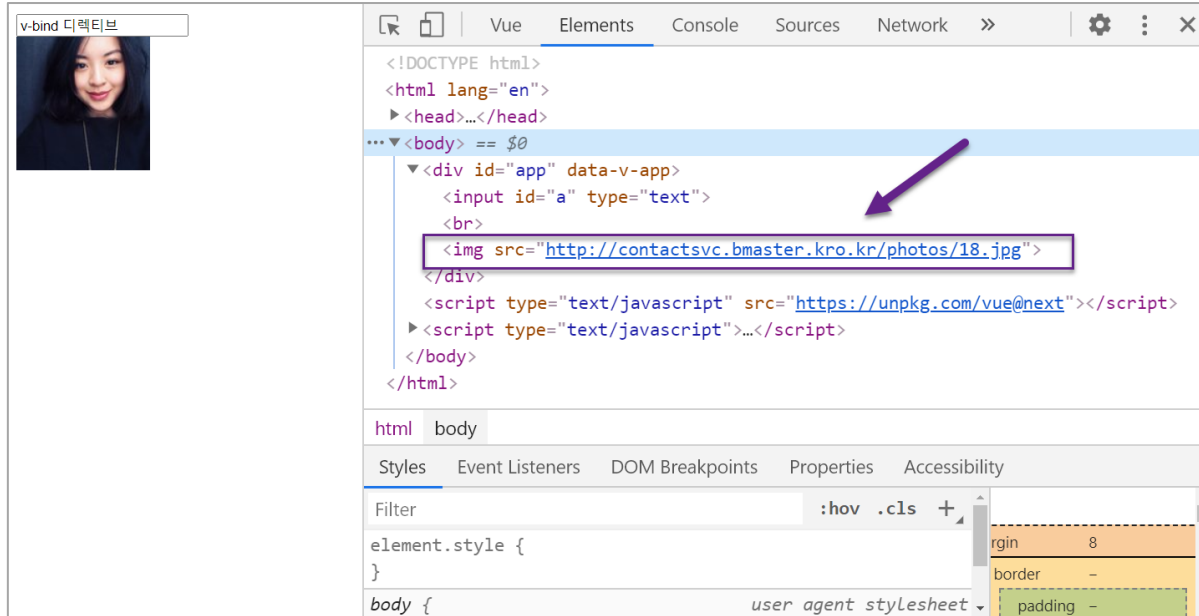
Template



v-bind:src = :src

2. 기본 디렉티브

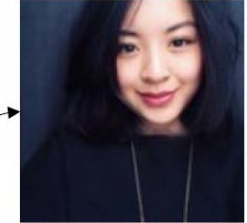
❖v-bind 디렉티브 예제 실행 결과



Vue 인스턴스의 데이터

imagePath : "https://contactsvc.bmaster.kro.kr/photos/18.jpg"

Template

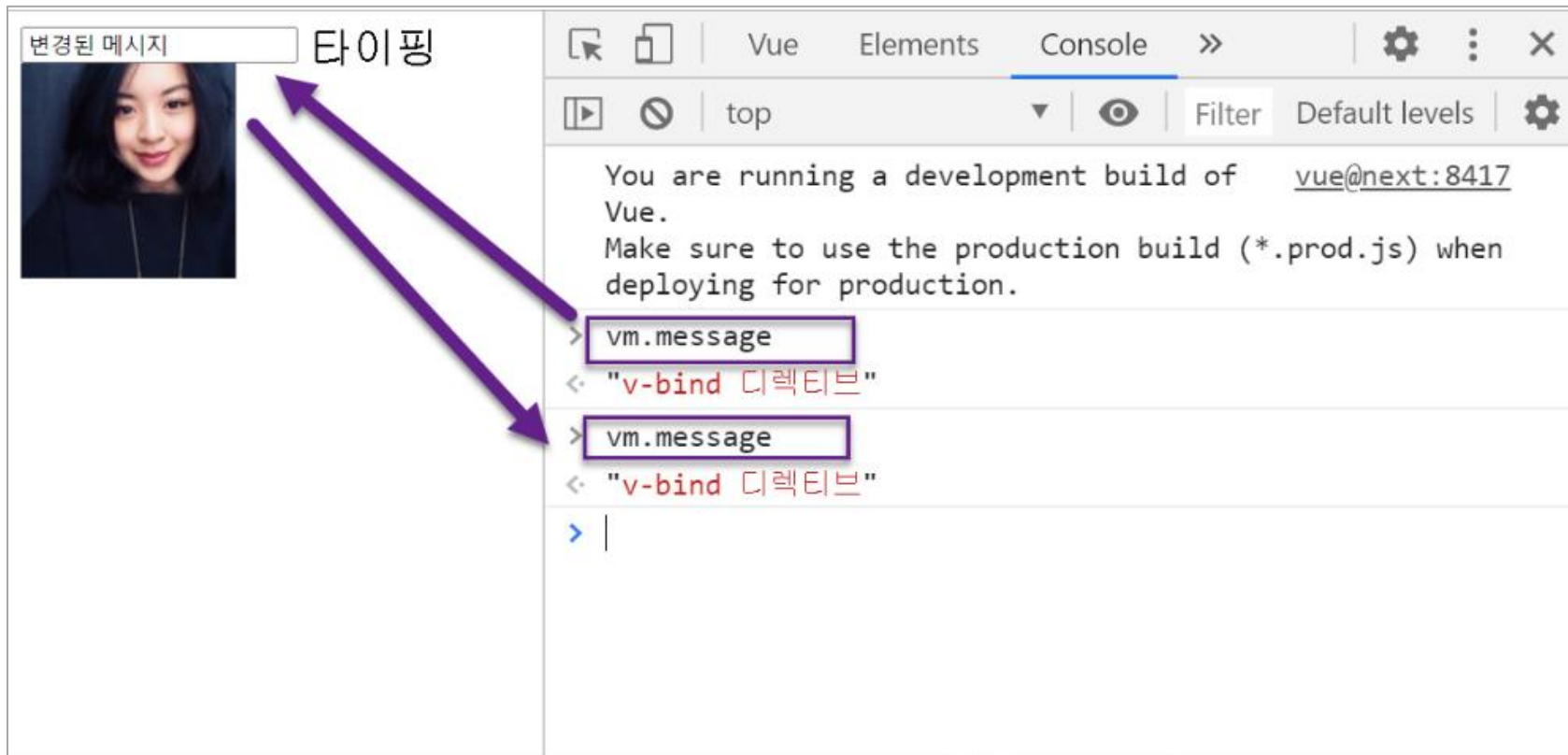


v-bind:src = :src

2. 기본 디렉티브

❖ 단방향 데이터 바인딩

- 데이터 변경 ---> UI 변경(O)
- UI에서 입력 ---> 데이터 변경(X)



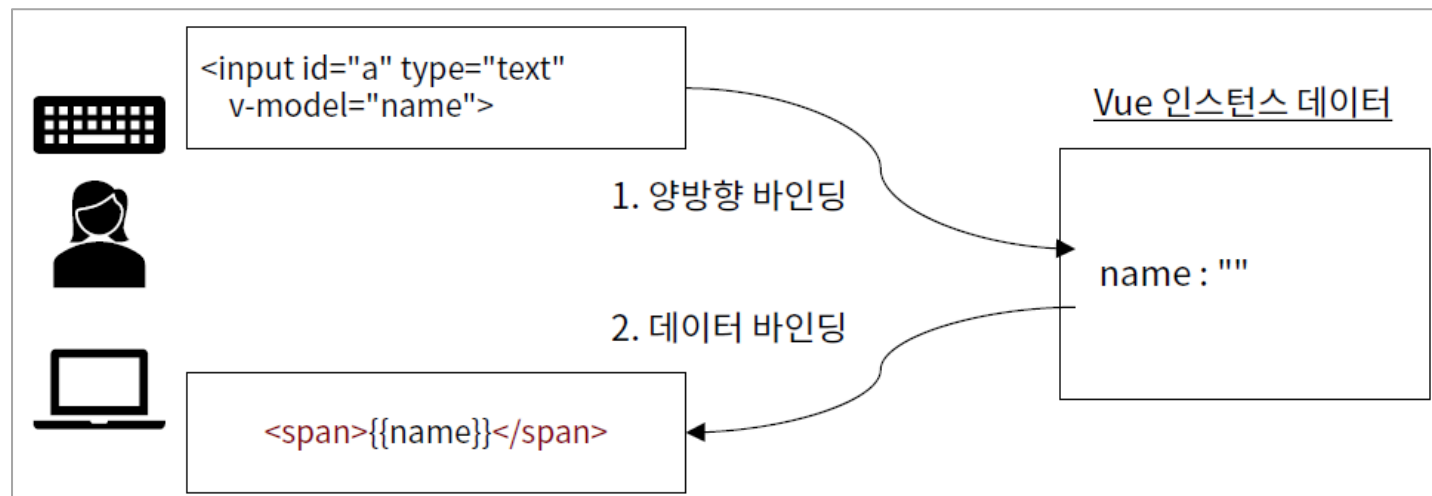
3. v-model 디렉티브

❖v-model 디렉티브

■ 양방향 데이터 바인딩 지원 : 예제 03-04

예제 03-04

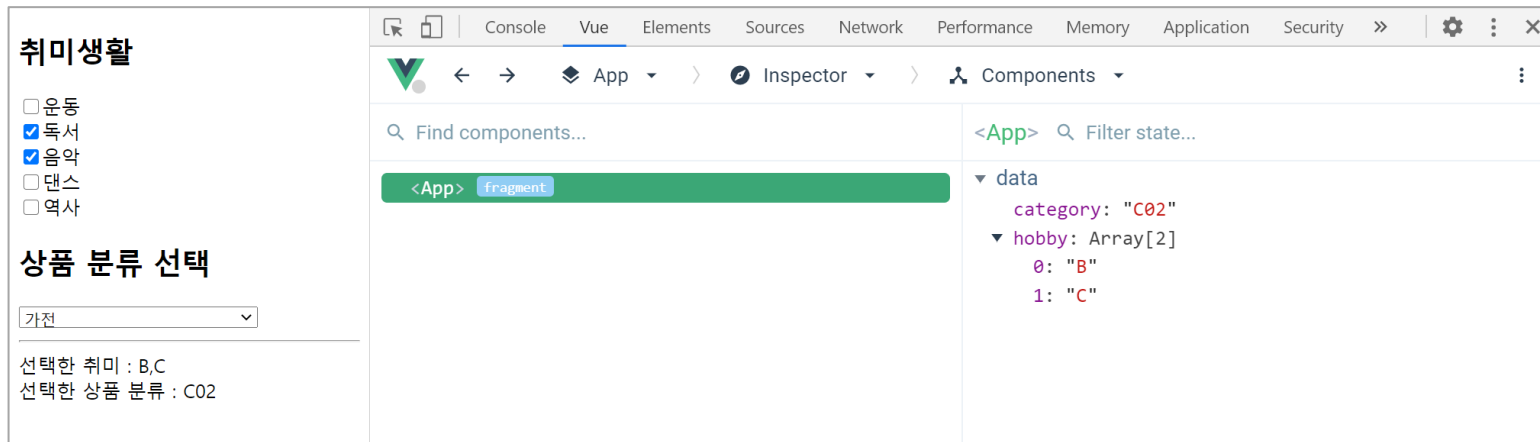
```
01: <body>
02:   <div id="app">
03:     <input id="a" type="text" v-model="name">
04:     <br />
05:     입력하신 이름 : <span>{{name}}</span>
06:   </div>
07:   <script type="text/javascript" src="https://unpkg.com/vue"></script>
08:   <script type="text/javascript">
09     var vm = Vue.createApp({
10       name : "App",
11       data() {
12         return { name : "" };
13       }
14     }).mount('#app')
15   </script>
16 </body>
```



3. v-model 디렉티브

❖ v-model + checkbox, select, radio

- 예제 03-05 확인
- 다중 선택 vs 단일 선택
 - 다중 선택 : 배열 데이터로 값을 받아냄
 - 단일 선택 : 문자열 값으로 받아냄
- checkbox, radio 등의 input 요소를 사용하는 경우
 - input 요소마다 v-model 디렉티브 사용
- select와 같이 값을 선택하는 요소를 감싸는 부모요소가 있는 경우
 - 부모 요소인 select에 v-model 디렉티브를 지정함



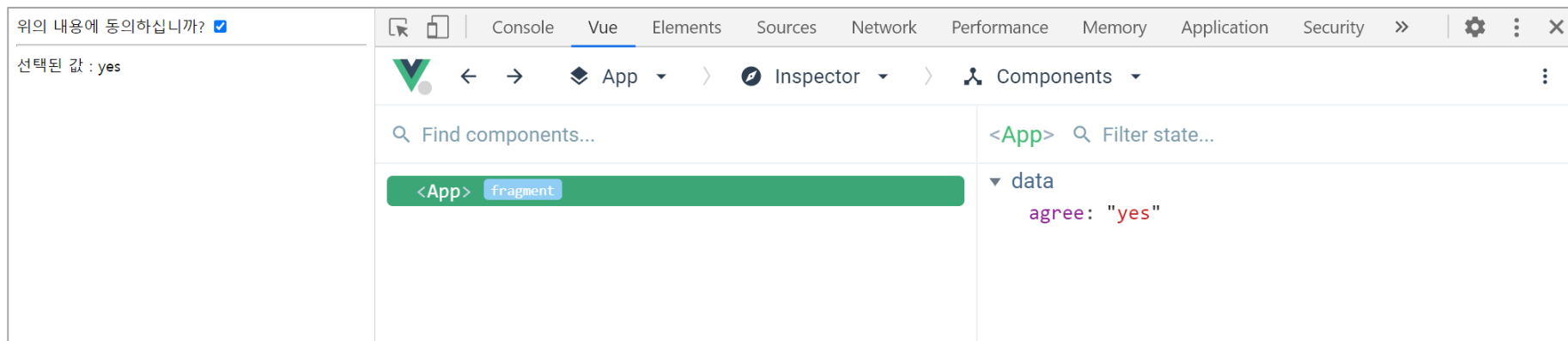
3. v-model 디렉티브

❖ checkbox로 선택, 미선택으로 구분하여 값을 전달받으려면...

■ 예제 03-06

예제 03-06

```
01: <div id="app">
02:   위의 내용에 동의하십니까?
03:   <input type="checkbox" v-model="agree" true-value="yes" false-value="no" />
04:   <hr />
05:   <span>선택된 값 : {{agree}}</span>
06: </div>
07: <script type="text/javascript" src="https://unpkg.com/vue"></script>
08: <script type="text/javascript">
09:   var vm = Vue.createApp({
10:     name : "App",
11:     data() {
12:       return {
13:         agree : "no"
14:       };
15:     }
16:   }).mount('#app')
17: </script>
```



3. v-model 디렉티브

❖v-model 디렉티브 수식어

- lazy: 입력폼에서 다른 요소로 포커스가 이동하는 이벤트가 발생할 때 입력한 값을 데이터와 동기화합니다.

예) `<input type="text" v-model.lazy="name" />`

- number: 이 수식어를 지정하면 숫자가 입력될 경우 number 타입의 값으로 자동 형변환되어 데이터 옵션 값으로 반영됩니다.

예) `<input type="text" v-model.number="num" />`

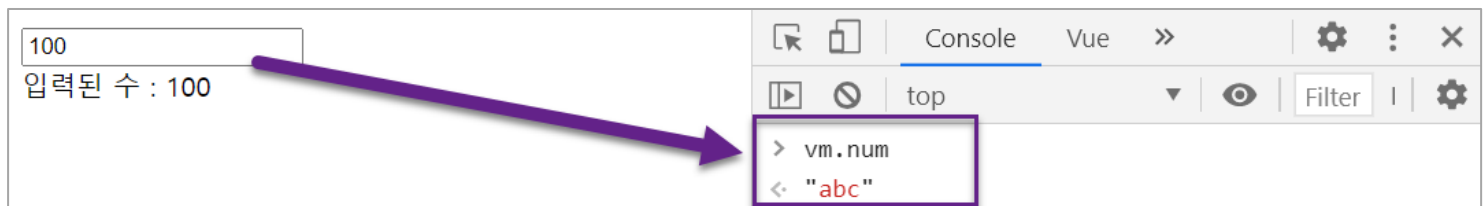
- trim: 이 수식어를 지정하면 문자열의 앞뒤 공백을 자동으로 제거합니다.

예) `<input type="text" v-model.trim="message" />`

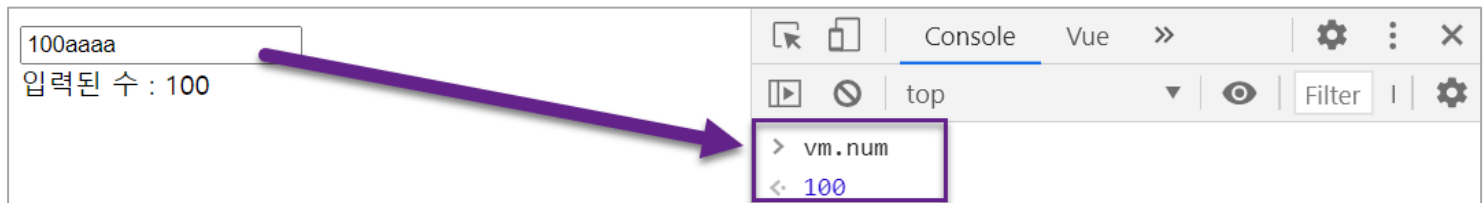
3. v-model 디렉티브

❖예제 03-07

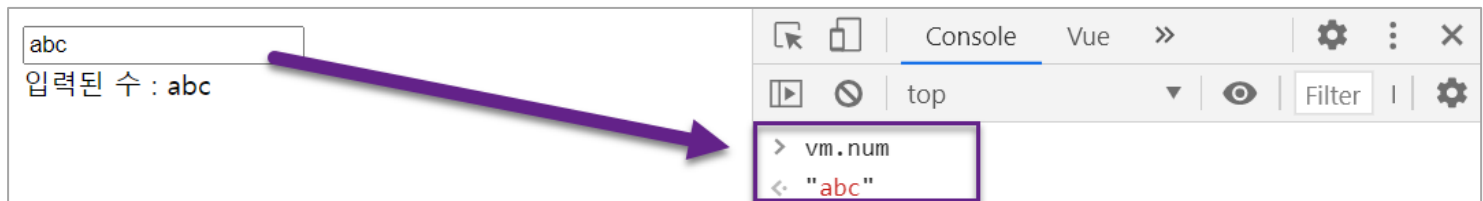
```
8 <body>
9 <div id="app">
10 <input type="text" v-model.number="num" /><br />
11 입력된 수 : <span>{{ num }}</span>
12 </div>
13 <script type="text/javascript" src="https://unpkg.com/vue"></script>
14 <script type="text/javascript">
15   var vm = Vue.createApp({
16     name: "App",
17     data() {
18       return { num: 0 };
19     },
20   }).mount("#app");
21 </script>
22 </body>
```



The screenshot shows a web application with a text input field containing the value '100'. Below the input field, the text '입력된 수 : 100' is displayed. A purple arrow points from the input field to the console log. The console log shows the command '> vm.num' and the result '< "abc"'. The console interface includes tabs for 'Console' and 'Vue', and a 'Filter' button.



The screenshot shows a web application with a text input field containing the value '100aaaa'. Below the input field, the text '입력된 수 : 100' is displayed. A purple arrow points from the input field to the console log. The console log shows the command '> vm.num' and the result '< 100'. The console interface includes tabs for 'Console' and 'Vue', and a 'Filter' button.



The screenshot shows a web application with a text input field containing the value 'abc'. Below the input field, the text '입력된 수 : abc' is displayed. A purple arrow points from the input field to the console log. The console log shows the command '> vm.num' and the result '< "abc"'. The console interface includes tabs for 'Console' and 'Vue', and a 'Filter' button.

3. v-model 디렉티브

❖ 한글 처리 문제

- 값을 받아내는 시점 : 한글 한글자가 입력이 완료될 때

홍길도

입력하신 이름 : 홍길

- 해결책 : 이벤트 핸들러를 이용해야 함
 - 5장에서 다루는 내용
 - 예제 03-08

```
<body>
  <div id="app">
    <input id="a" type="text" :value="name" @input="changeName" />
    <br />
    입력하신 이름 : <span>{{name}}</span>
  </div>
  <script type="text/javascript" src="https://unpkg.com/vue"></script>
  <script type="text/javascript">
    var vm = Vue.createApp({
      name: "App",
      data() {
        return { name: "" };
      },
      methods: {
        changeName(e) {
          this.name = e.target.value;
        },
      },
    }).mount("#app");
  </script>
</body>
```

4. 조건 렌더링 디렉티브

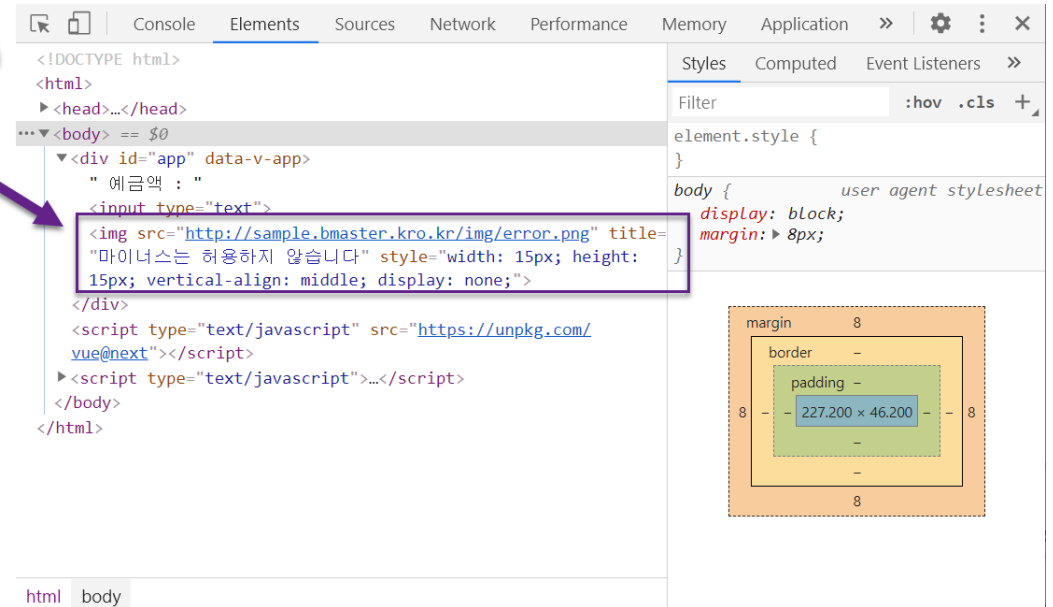
❖ v-show

■ 화면에 보여줄지 말지를 결정하는 디렉티브

- 렌더링은 수행하지만(HTML 요소는 생성하지만) 화면에 보여주지 않을 수 있음
- display CSS 속성을 none으로 지정하여 보이지 않게 함

예제 03-09

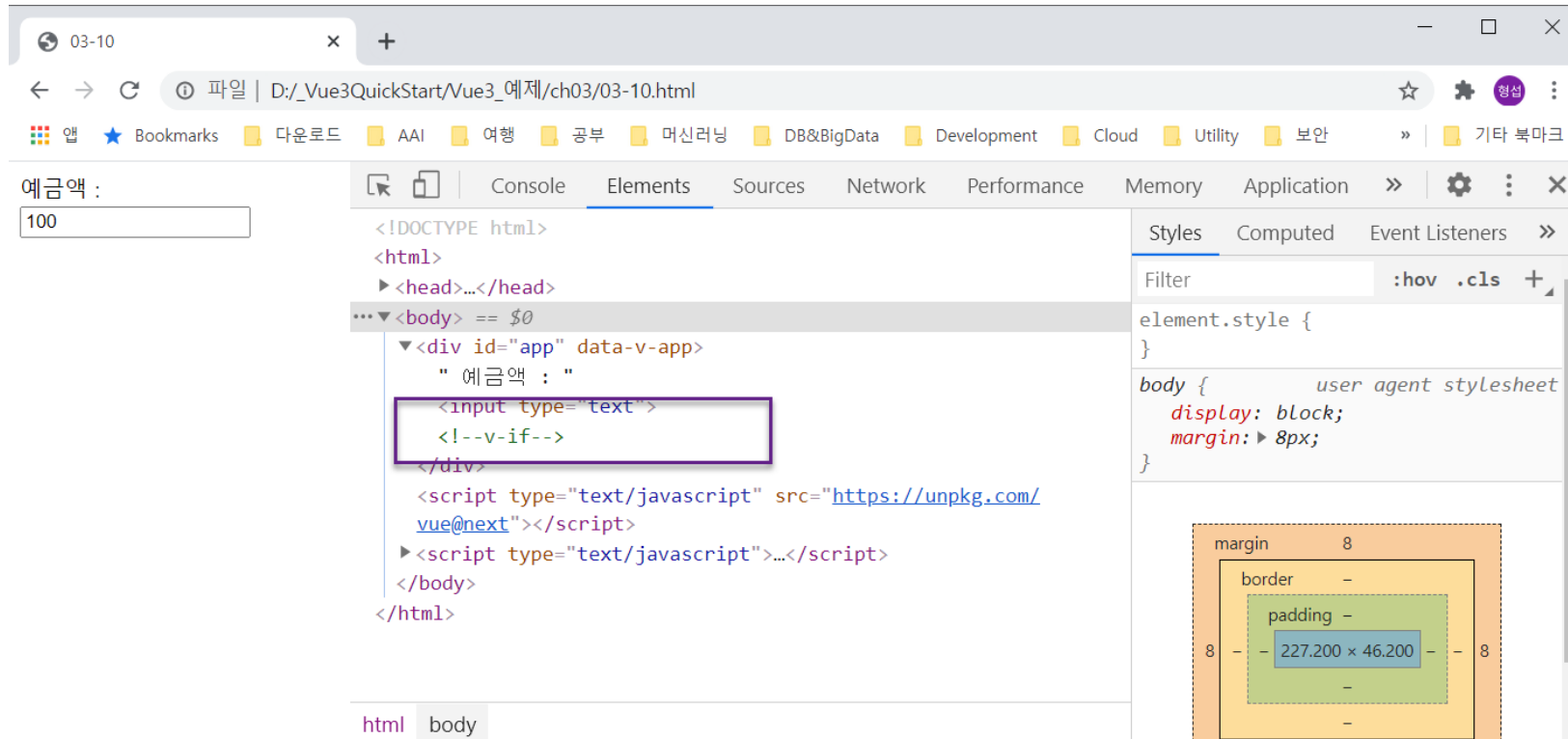
```
01: <body>
02:   <div id="app">
03:     예금액 : <input type="text" v-model="amount" />
04:     
07:   </div>
08:   <script type="text/javascript" src="https://unpkg.com/vue"></script>
09:   <script type="text/javascript">
10:     var vm = Vue.createApp({
11:       data() {
12:         return { amount : "" };
13:       },
14:     }).mount('#app')
15:   </script>
16: </body>
```



4. 조건 렌더링 디렉티브

❖v-if

- 조건에 부합되지 않을 때 렌더링을 수행하지 않도록 함.
- 예제 03-09의 v-show를 v-if로 변경해 봄



4. 조건 렌더링 디렉티브

❖v-else, v-else-if

```
<body>
  <div id="app">
    잔고 : <input type="text" v-model="balance" />
    <br />
    회원님의 등급 :
    <span v-if="balance >= 1000000">Gold</span>
    <span v-else-if="balance >= 500000">Silver</span>
    <span v-else-if="balance >= 200000">Bronze</span>
    <span v-else>Basic</span>
  </div>
  <script type="text/javascript" src="https://unpkg.com/vue"></script>
  <script type="text/javascript">
    var vm = Vue.createApp({
      name: "App",
      data() {
        return { balance: 0 };
      },
    }).mount("#app");
  </script>
</body>
```

잔고 :

회원님의 등급 : Silver

5. 반복 렌더링(1)

❖ v-for 디렉티브

- 반복적인 데이터(배열, 객체)를 렌더링할 때 사용
- Javascript의 for문과 유사함

```
<div id="app">
  <table id="list">
    <thead>
      <tr>
        <th>번호</th>
        <th>이름</th>
        <th>전화번호</th>
      </tr>
    </thead>
    <tbody id="contacts">
      <tr v-for="contact in contacts" :key="contact.no">
        <td>{{contact.no}}</td>
        <td>{{contact.name}}</td>
        <td>{{contact.tel}}</td>
      </tr>
    </tbody>
  </table>
</div>
```

```
<script src="https://unpkg.com/vue"></script>
<script type="text/javascript">
  var vm = Vue.createApp({
    name: "App",
    data() {
      return {
        pageno: 1,
        pagesize: 4,
        totalcount: 100,
        contacts: [
          { no: 1011, name: "RM", tel: "010-3456-8299" },
          { no: 1012, name: "정국", tel: "010-3456-8298" },
          { no: 1013, name: "제이홉", tel: "010-3456-8297" },
          { no: 1014, name: "슈가", tel: "010-3456-8296" },
        ],
      };
    },
  }).mount("#app");
</script>
```

5. 반복 렌더링(2)

■ 실행 결과

The screenshot shows a web browser with a table of contacts. The table has three columns: 번호 (Number), 이름 (Name), and 전화번호 (Phone Number). The data is as follows:

번호	이름	전화번호
1011	RM	010-3456-8299
1012	정국	010-3456-8298
1013	제이홉	010-3456-8297
1014	슈가	010-3456-8296

The Chrome DevTools Elements panel is open, showing the DOM tree. A purple arrow points from the 'Elements' tab to a specific row in the table. The DOM tree structure is as follows:

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <div id="app" data-v-app> == $0
      <table id="list">
        <thead>...</thead>
        <tbody id="contacts">
          <tr>
            <td>1011</td>
            <td>RM</td>
            <td>010-3456-8299</td>
          </tr>
          <tr>...</tr>
          <tr>...</tr>
          <tr>...</tr>
        </tbody>
      </table>
    </div>
  </body>
</html>
```

The Styles panel on the right shows the default user agent styles for a div, including display: block. A box model diagram is also visible, showing margin, border, padding, and the content area (493 x 121).

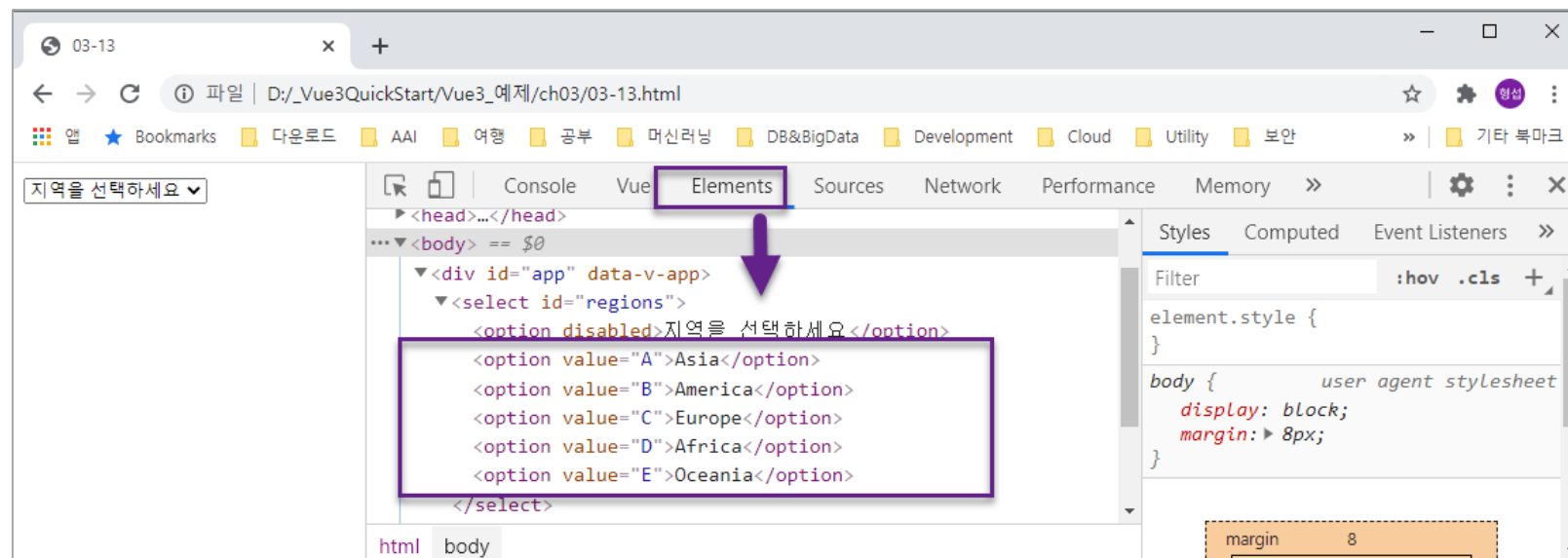
```
<tr v-for="contact in contacts" :key="contact.no"> ... </tr>
```

5. 반복 렌더링(3)

❖ 원본이 객체인 경우

■ 예제 03-13

```
<div id="app">
  <select id="regions">
    <option disabled="disabled" selected>지역을 선택하세요</option>
    <option v-for="(val, key) in regions" :value="key" :key="key">{{val}}</option>
  </select>
</div>
<script src="https://unpkg.com/vue"></script>
<script type="text/javascript">
  var vm = Vue.createApp({
    name: "App",
    data() {
      return {
        regions: {
          A: "Asia",
          B: "America",
          C: "Europe",
          D: "Africa",
          E: "Oceania",
        },
      },
    },
  }).mount("#app");
</script>
```



5. 반복 렌더링(4)

❖인덱스 번호가 필요한 경우

- 배열 데이터인 경우(예제 03-12의 코드와 비교해보세요)

```
<tr v-for="(contact, index) in contacts" ... > ... </tr>
```

- 객체 데이터인 경우(예제 03-13의 코드와 비교해보세요)

```
<option v-for="(val, key, index) in regions" ... > ... </option>
```

예제 03-14

```
<tbody id="contacts">
  <tr v-for="(contact, index) in contacts" :key="contact.no">
    <td>{{index+1}}</td>
    <td>{{contact.name}}</td>
    <td>{{contact.tel}}</td>
  </tr>
</tbody>
```

5. 반복 렌더링(5)

❖여러 요소를 묶어서 반복 렌더링

▪ 예제 03-15

```
<tbody id="contacts">
  <template v-for="(contact, index) in contacts" :key="contact.no">
    <tr>
      <td>{{contact.no}}</td>
      <td>{{contact.name}}</td>
      <td>{{contact.tel}}</td>
    </tr>
    <tr class="divider" v-if="index % 4 === 3">
      <td colspan="3"></td>
    </tr>
  </template>
</tbody>
```

번호	이름	전화번호
1011	RM	010-3456-8299
1012	정국	010-3456-8298
1013	제이홉	010-3456-8297
1014	슈가	010-3456-8296
1014	진	010-3456-8296
1014	뷔	010-3456-8296
1014	지민	010-3456-8296

5. 반복 렌더링(6)

❖v-for 디렉티브와 key 특성

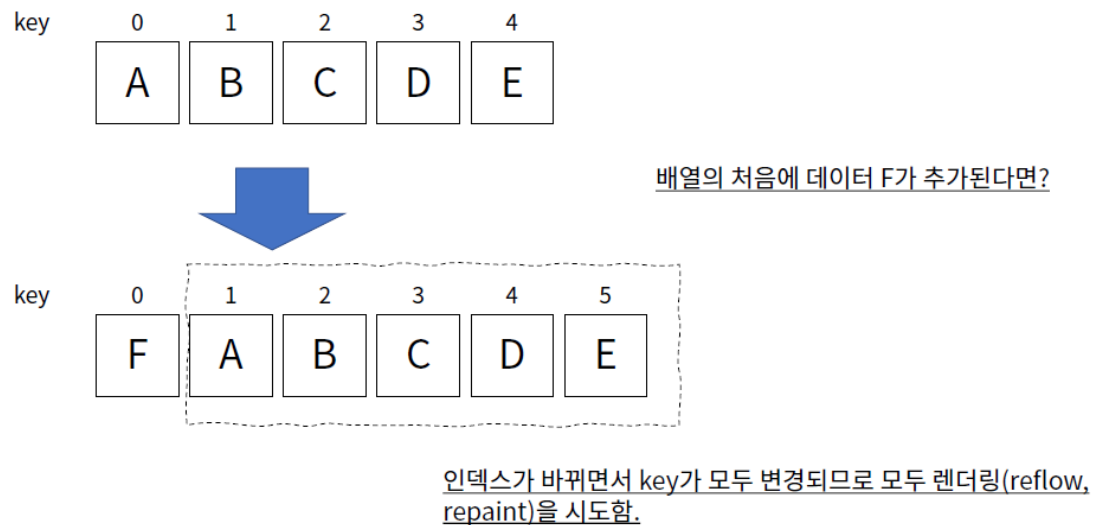
```
<tr v-for="contact in contacts" :key="contact.no"> ... </tr>
```

- 가상 DOM은 변경된 부분만을 브라우저 DOM으로 업데이트
- key 특성을 부여하지 않으면?
 - 렌더링은 수행됨
 - 하지만 Vue.js의 가상 DOM은 v-for로 렌더링한 배열 데이터의 순서가 변경되면 DOM 요소를 이동시키지 않고 기존 DOM 요소의 데이터를 변경 ---> Re-render!! --> 성능 저하
- 만일 DOM 요소를 추적하여 DOM 요소의 위치를 변경하려면? --> key 특성을 지정해야 함
 - key 특성에는 고유한 값을 부여해야 함, index 번호(X)
 - 예) 데이터베이스를 조회했다면 Primary Key 값

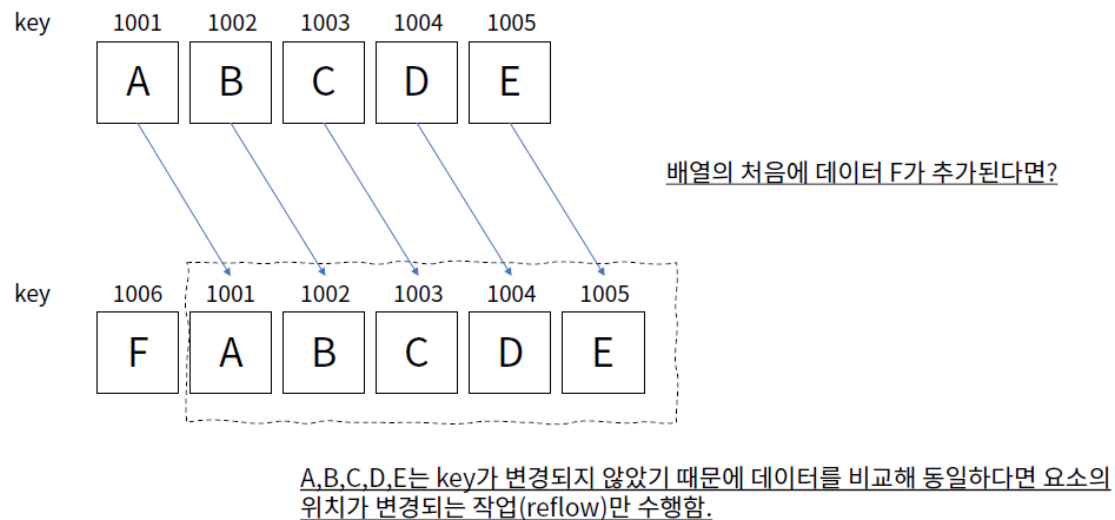
5. 반복 렌더링(7)

❖index를 부여한 경우와 고유키를 부여한 경우를 비교

Key에 index를 부여하는 경우



Key에 고유키를 부여하는 경우

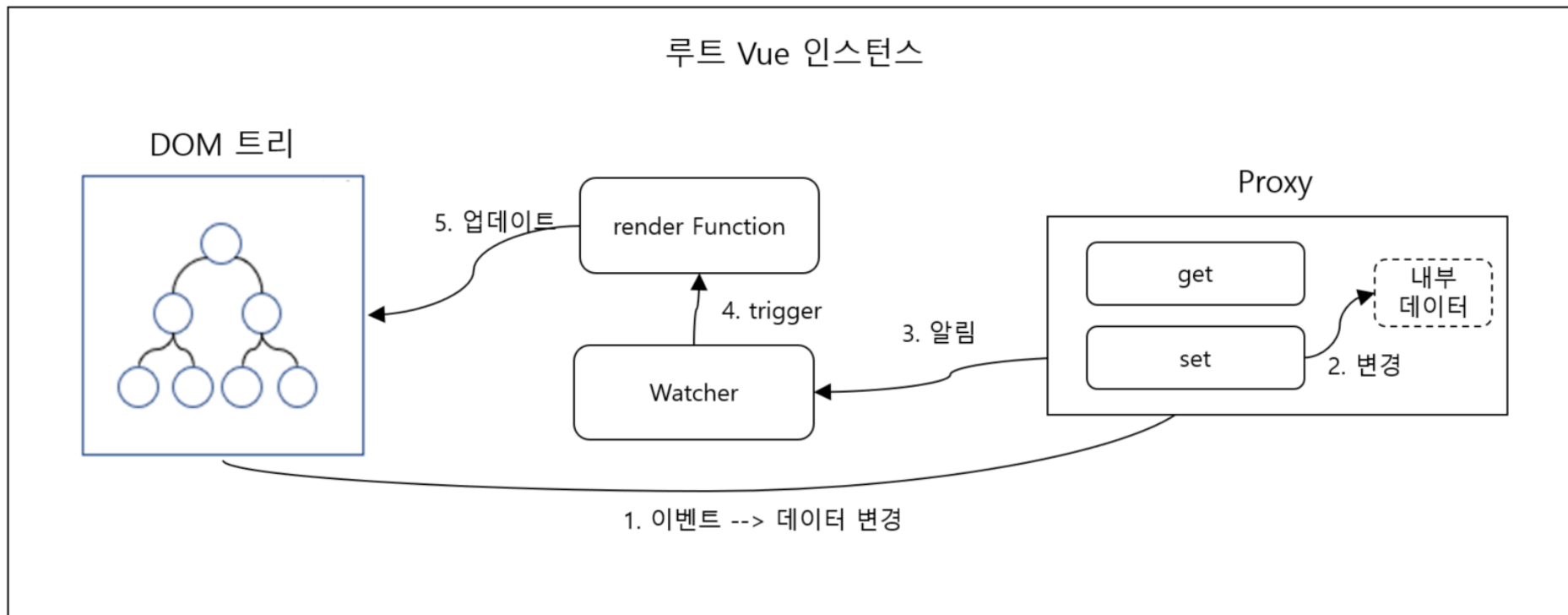


- key 특성 반드시 지정합니다.
 - 고유한 값으로

5. 반복 렌더링(8)

❖데이터 변경시 주의사항

- Vue의 반응성은 Proxy가 제공함
- 반드시 Proxy를 통해서 변경해야 함
 - 만일 Proxy를 거치지 않으면 : data 변경 ----> UI 변경(X)



5. 반복 렌더링(9)

❖예제 03-16

```
<div id="app">
  <table id="list">
    <thead> ...
  </thead>
  <tbody id="contacts">
    <tr v-for="contact in contacts" :key="contact.no">
      <td>{{contact.no}}</td>
      <td>{{contact.name}}</td>
      <td>{{contact.tel}}</td>
    </tr>
  </tbody>
</table>
</div>
<script src="https://unpkg.com/vue"></script>
<script type="text/javascript">
  var model = {
    contacts: [
      { no: 1011, name: "RM", tel: "010-3456-8299" },
      { no: 1012, name: "정국", tel: "010-3456-8298" },
      { no: 1013, name: "제이홉", tel: "010-3456-8297" },
      { no: 1014, name: "슈가", tel: "010-3456-8296" },
    ],
  };
  var vm = Vue.createApp({
    name: "App",
    data() {
      return model;
    },
  }).mount("#app");
</script>
```

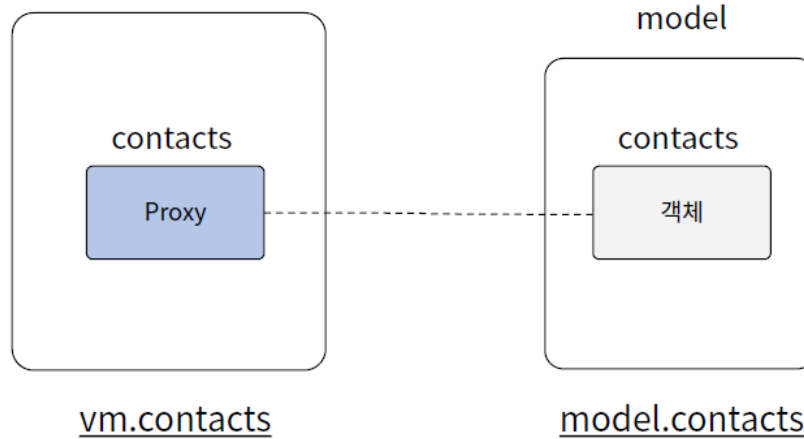
```
> vm.contacts
< ▶ Proxy {0: {...}, 1: {...}, 2: {...}, 3: {...}}

> model.contacts
< ▶ (4) [{...}, {...}, {...}, {...}]

> vm.contacts[0]
< ▶ Proxy {no: 1011, name: "RM", tel: "010-3456-8299"}

> model.contacts[0]
< ▶ {no: 1011, name: "RM", tel: "010-3456-8299"}
```

vm (루트 컴포넌트 인스턴스)



5. 반복 렌더링(10)

❖데이터 변경 시 주의 사항

- 절대 Vue 인스턴스 밖에서 선언된 객체, 배열을 직접 변경하지 않도록 함

❖데이터로 배열이 사용된 경우

- 배열 내부의 데이터 뿐만 아니라 배열의 메서드(push, splice, sort 등)도 래핑함

❖예제 03-16 + 콘솔 에서 다음 명령 실행

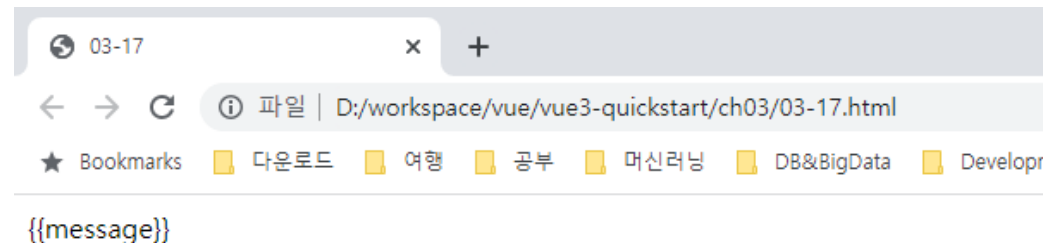
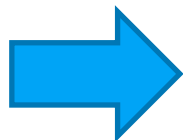
- `model.contacts[0].tel = "010-9999-9999"`
 - 변화 없음
- `vm.contacts[1].tel = "010-8888-8888"`
 - 이전 명령으로 인한 데이터 변경까지 렌더링됨
- `model.contacts.push({ no:1201, name:"진", tel:"010-7777-7777" })`
 - 변화 없음
- `vm.contacts.push({ no:1202, name:"뷔", tel:"010-5555-5555" })`
 - 이전 명령으로 추가된 데이터도 함께 렌더링됨

6. 기타 디렉티브

❖ v-pre

- HTML 요소에 대한 컴파일을 수행하지 않음

```
<div id="app">  
  <span v-pre>{{message}}</span>  
</div>
```

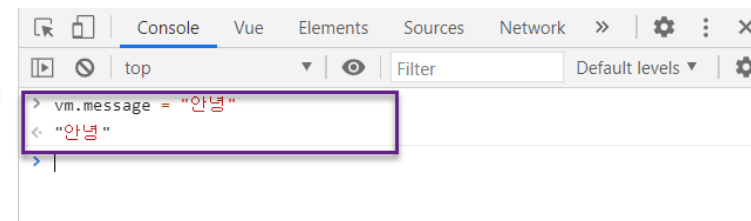


❖ v-once

- 최초 한번만 렌더링
- 데이터가 변경되더라도 다시 렌더링하지 않음

```
<div id="app">  
  <span v-once>{{message}}</span>  
</div>
```

Hello World



6. 기타 디렉티브

❖v-cloak

- 화면에 많은 데이터를 출력할 때 콧수염 표현식이 일시적으로 나타나기도 함
 - Vue 인스턴스가 템플릿을 컴파일할 때 발생하는 시간으로 인한 현상

예제 03-19

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: .....(중략)
05: <style>
06:     #list { width: 600px; border:1px solid black; border-collapse:collapse; }
07:     #list td, #list th { border:1px solid black; text-align:center; }
08:     #list > thead > tr { color:yellow; background-color: purple; }
09:     [v-cloak] { display:none; }
10: </style>
11: </head>
12: <body>
13:     <div id="app" v-cloak>
14:         .....(중략)
15:     </div>
16:     .....(중략)
17: </body>
18: </html>
```


7. 동적 아규먼트

❖ 동적 아규먼트란?

- 연결하고자 하는 특성의 이름을 데이터나 속성으로 지정할 수 있는 문법
- v-bind나 이벤트 처리를 위한 v-on 디렉티브에서 사용할 수 있음

【 v-bind 예시 】

```
<element v-bind:[attrName] = "[attrValue]"></element>  
<element :[attrName] = "[attrValue]"></element>
```

【 v-on 예시 】

```
<element v-on:[eventName] = "[function code]"></element>  
<element @[eventName] = "[function code]"></element>
```

예제 03-20

```
01: <body>  
02:   <div id="app">  
03:     <img v-bind:[image1.srcattr]="image1.src"  
04:         :[image1.titleattr]="image1.title"/>  
05:   </div>  
06:   <script type="text/javascript" src="https://unpkg.com/vue"></script>  
07:   <script type="text/javascript">  
08:     var vm = Vue.createApp({  
09:       name : "App",  
10:       data() {  
11:         return {  
12:           image1 : {  
13:             srcattr:"src", src:"https://contactsvc.bmaster.kro.kr/photos/18.jpg",  
14:             titleattr: "title", "title" : "Lily's photo"  
15:           }  
16:         };  
17:       }  
18:     }).mount('#app')  
19:   </script>  
20: </body>
```

7. 동적 아규먼트

❖ 동적 아규먼트 제약

- `v-bind:[image1.srcattr]` 형식에서 키 이름은 대문자를 사용해도 소문자인 데이터를 액세스함
 - `image1.srcAttr` 로 사용해도 `image1.srcattr`을 액세스함
- `[]` 내부는 표현식으로 연산식을 사용할 수 없음
 - `v-bind['image'+num]` --> 허용하지 않음
 - 이 작업을 하려면 계산된 속성을 사용해야 함



계산된 속성(computed property)이란?

계산된 속성은 data나 다른 속성이 변경될 때 함수가 실행되어 계산된 값을 이용하기 위해 사용합니다. 이에 대한 자세한 내용은 4장에서 학습합니다.

- 전역 화이트 리스트에 등록된 문자열은 키 이름으로 사용할 수 없음
 - Infinity, NaN, Math, Number 등
 - <https://github.com/vuejs/core/blob/main/packages/shared/src/globalsWhitelist.ts>

7. 동적 아규먼트

❖여러개의 data를 한번에 속성으로 전달하고 싶다면?

```
<div id="app">  
  <img v-bind="image1" />  
</div>  
<script type="text/javascript" src="https://unpkg.com/vue"></script>  
<script type="text/javascript">  
  var vm = Vue.createApp({  
    name: "App",  
    data() {  
      return {  
        image1: {  
          src: "https://contactsvc.bmaster.kro.kr/photos/18.jpg",  
          title: "Lily's photo",  
        },  
      };  
    },  
  }).mount("#app");  
</script>
```

```

```

8. 마무리

이제까지 Vue.js의 기초적인 내용으로 여겨지는 보간법과 디렉티브를 이용하는 템플릿 작성 방법에 대해 살펴보았습니다. 그리 어렵지 않은 내용이지만 v-model을 이용한 양방향 바인딩에서의 한글 문제나 반복 렌더링에서의 key 특성 문제와 같이 세세하게 신경 써야 하는 부분들이 있습니다. 템플릿의 대한 학습은 데이터를 화면으로 표현하는 가장 기본적인 내용이므로 여러 번 학습하여 내 것으로 만들어주세요.