Form 입력 바인딩

프론트엔드에서 폼을 처리할 때, 폼 입력 엘리먼트의 상태를 JavaScript의 상태와 동기화해야 하는 경우가 많습니다. 값 바인딩을 수동으로 연결하고 이벤트 리스너를 변경하는 것은 번거로운 작업입니다:

template

```
<input
 :value="text"
 @input="event => text = event.target.value">
```

v-model 디렉티브는 위의 내용을 다음과 같이 단순화하는 데 도움이 됩니다:

template

```
<input v-model="text">
```

또한 v-model 은 다른 유형의 입력인 <textarea> 및 <select> 엘리먼트에 사용할 수 있습니다. 사용되는 엘리먼트에 따라 자동으로 다른 DOM 속성 및 이벤트 쌍으로 확장됩니다.

텍스트 유형의 <input> 과 <textarea> 경우, value 속성과 input 이벤트를 사용합니다.
<input type="checkbox"> 과 <input type="radio"> 경우, checked 속성과 change 이벤트를 사용합니다.
<select> 는 value 를 속성으로 사용하고 change 를 이벤트로 사용합니다.

참고

v-model 은 모든 폼 엘리먼트에서 감지되는 초기 value , checked 또는 selected 속성 값을 무시합니다. 항상 현재 바인딩된 JavaScript 상태를 유효한 값으로 취급합니다. reactivity API를 사용하여 JavaScript에서 초기 값을 선언해야 합니다.

기본 사용법

텍스트

template

```
<p>메세지: {{ message }}</p>
<input v-model="message" placeholder="메세지 입력하기" />
```

메세지 내용:

메세지 입력하기

온라인 연습장으로 실행하기

참고

IME가 필요한 언어(중국어, 일본어, 한국어 등)의 경우 IME 구성 중에 v-model 이 업데이트되지 않는 것을 알 수 있습니다. 이러한 업데이트에도 응답하려면 v-model 을 사용하는 대신 직접 구현한 input 이벤트 리스너와 value 바인딩을 사용해 기능을 구성해야 합니다.

여러 줄 텍스트

template

```
<span>여러 줄 메세지:</span>
<p style="white-space: pre-line;">{{ message }}</p>
<textarea v-model="message" placeholder="여러 줄을 추가해보세요"></textarea>
```

여러 줄 메세지:

여러 줄을 추가해보세
요

온라인 연습장으로 실행하기

<textarea> 내부에 이중 중괄호 문법은 작동하지 않으므로 v-model 을 사용해야 합니다.

template

```
<!-- 잘못된 사례 -->
<textarea>{{ text }}</textarea>

<!-- 올바른 사례 -->
<textarea v-model="text"></textarea>
```

체크박스

단일 체크박스는 불리언을 값을 사용합니다:

template

```
<input type="checkbox" id="checkbox" v-model="checked" />
<label for="checkbox">{{ checked }}</label>
```

☐ false

온라인 연습장으로 실행하기

배열 또는 **Set**에 여러 개의 체크박스 값을 바인딩할 수도 있습니다.

js

```
const checkedNames = ref([])
```

template

```
<div>체크된 이름: {{ checkedNames }}</div>

<input type="checkbox" id="jack" value="젝" v-model="checkedNames">
<label for="jack">젝</label>

<input type="checkbox" id="john" value="존" v-model="checkedNames">
<label for="john">존</label>

<input type="checkbox" id="mike" value="마이크" v-model="checkedNames">
<label for="mike">마이크</label>
```

체크된 이름: []
☐ 젝 ☐ 존 ☐ 마이크

이 경우 checkedNames 배열은 항상 현재 체크된 순서대로 값을 포함합니다.

온라인 연습장으로 실행하기

라디오

template

```
<div>선택한 것: {{ picked }}</div>

<input type="radio" id="one" value="하나" v-model="picked" />
<label for="one">하나</label>

<input type="radio" id="two" value="둘" v-model="picked" />
<label for="two">둘</label>
```

선택한 것:
○ 하나 ○ 둘

온라인 연습장으로 실행하기

셀렉트

단일 셀렉트:

template

```
<div>선택됨: {{ selected }}</div>

<select v-model="selected">
  <option disabled value="">다음 중 하나를 선택하세요</option>
  <option>가</option>
  <option>나</option>
  <option>다</option>
</select>
```

선택됨:
다음 중 하나를 선택하세요

온라인 연습장으로 실행하기

> 참고
>
> `v-model` 표현식의 초기 값이 옵션과 일치하지 않으면 `<select>` 엘리먼트가 "선택되지 않은" 상태로 렌더링됩니다. iOS에서는 이 경우 변경 이벤트를 발생시키지 않기 때문에 사용자가 첫 번째 항목을 선택할 수 없게 됩니다. 따라서 위의 예에서 설명한 것처럼 비활성화된 옵션에 빈 값을 제공하는 것이 좋습니다.

다중 선택(배열로 바인딩 됨):

template

```
<div>선택됨: {{ selected }}</div>

<select v-model="selected" multiple>
  <option>가</option>
  <option>나</option>
  <option>다</option>
</select>
```

선택됨: []

```
가
나
다
```

온라인 연습장으로 실행하기

셀렉트 옵션은 `v-for` 로 동적으로 렌더링할 수 있습니다:

```js
const selected = ref('1')

const options = ref([
  { text: '하나', value: '1' },
  { text: '둘', value: '2' },
  { text: '셋', value: '3' }
])
```

```template
<select v-model="selected">
  <option v-for="option in options" :value="option.value">
    {{ option.text }}
  </option>
</select>

<div>선택됨: {{ selected }}</div>
```

온라인 연습장으로 실행하기

값 바인딩 하기

라디오, 체크박스 및 셀렉트 옵션의 경우, `v-model` 에 바인딩된 값은 일반적으로 정적 문자열(체크박스의 경우 불리언)입니다:

```template
<!-- `picked`는 선택 시 문자열 "가"입니다. -->
<input type="radio" v-model="picked" value="가" />

<!-- `toggle`은 true 또는 false입니다. -->
<input type="checkbox" v-model="toggle" />

<!-- `selected`는 첫 번째 옵션이 선택될 때 문자열 "한글"입니다. -->
<select v-model="selected">
  <option value="한글">한글</option>
</select>
```

그러나 때로는 현재 활성 인스턴스의 동적 속성에 값을 바인딩하고 싶을 수도 있습니다. 이것을 구현하기 위해서는 `v-bind` 를 사용해야 합니다. 또한 `v-bind` 를 사용하면 입력 값을 문자열이 아닌 값에 바인딩할 수 있습니다.

Checkbox

```template
<input
  type="checkbox"
  v-model="toggle"
  true-value="네"
  false-value="아니오" />
```

true-value 및 false-value 속성은 v-model 을 사용하는 경우에만 작동하는 Vue 전용 속성입니다. 여기에서 toggle 속성의 값은 체크박스가 선택되면 네 로 설정되고 선택되지 않을 때는 아니오 로 설정됩니다. 이 전용 속성에 v-bind (:)를 사용하여 동적인 값을 바인딩할 수도 있습니다.

template

```
<input
 type="checkbox"
 v-model="toggle"
 :true-value="dynamicTrueValue"
 :false-value="dynamicFalseValue" />
```

Tip

브라우저는 폼 제출 시 체크되지 않은 상자는 포함하지 않기 때문에, true-value 와 false-value 속성은 입력의 value 속성에 영향을 주지 않습니다. 두 값 중 하나가 폼으로 제출되도록 하려면(예: "네" 또는 "아니오") 체크박스 대신 라디오로 구현해야 합니다.

라디오

template

```
<input type="radio" v-model="pick" :value="first" />
<input type="radio" v-model="pick" :value="second" />
```

pick 는 첫 번째 라디오 입력이 확인되면 first 값으로 설정되고 두 번째 라디오 입력이 확인되면 second 값으로 설정됩니다.

셀렉트 옵션

template

```
<select v-model="selected">
 <!-- 인라인 객체 리터럴 -->
 <option :value="{ number: 123 }">123</option>
</select>
```

v-model 은 문자열이 아닌 값의 바인딩도 지원합니다! 위의 예에서 옵션이 선택되면 selected 는 { number: 123 } 객체 값으로 설정됩니다.

수식어

.lazy

기본적으로 v-model 은 각 input 이벤트 이후 데이터와 입력을 동기화합니다(위에 언급된 IME 구성 제외). 대신 change 이벤트 이후에 동기화하기 위해 .lazy 수식어를 추가할 수 있습니다.

template

```
<!-- "input" 대신 "change" 이벤트 후에 동기화됨 -->
<input v-model.lazy="msg" />
```

.number

사용자 입력이 자동으로 숫자로 유형 변환되도록 하려면, v-model 수식어로 .number 를 추가하면 됩니다:

template

```
<input v-model.number="age" />
```

값을 parseFloat() 로 파싱할 수 없으면 원래 값이 대신 사용됩니다.

인풋에 type="number" 가 있으면 .number 수식어가 자동으로 적용됩니다.

.trim

사용자 입력의 공백이 자동으로 트리밍되도록 하려면 v-model 수식어로 .trim 을 추가하면 됩니다:

template

```
<input v-model.trim="msg" />
```

컴포넌트에 v-model 사용하기

Vue의 컴포넌트에 아직 익숙하지 않은 경우, 지금은 건너뛰어도 됩니다.

HTML의 기본 입력 유형이 항상 사용자의 요구를 충족하는 것은 아닙니다. 다행히도 Vue 컴포넌트를 사용하면 완전히 사용자 정의된 동작으로 재사용 가능한 입력을 구축할 수 있습니다. 이러한 입력은 v-model 에서도 작동합니다! 자세한 내용은 컴포넌트 가이드의 **v-model** 과 함께 사용하기를 참조하세요