

Vue 인스턴스



❖ vue 인스턴스란?

- new Vue({ ... })

```
var vm = new Vue({  
  el : '#simple',  
  data : {  
    name : '홍길동'  
  }  
})
```

el, data, computed 옵션



❑ data 옵션

- data 옵션의 속성들은 Vue 인스턴스 내부에서 직접 이용되지 않음
 - Vue 인스턴스와 Data 옵션에 주어진 객체 사이에 프록시를 두어 처리
 - Object.defineProperty() 메서드를 이용해 setter를 작성함.
 - setter가 호출될 때 watcher에게 알려 렌더링이 다시 일어나도록 제어함.
 - data 옵션은 Vue 인스턴스가 관찰 --> 변경 사항 즉시 감지
- `vm.name == vm.$data.name`

❑ el 옵션

- Vue 인스턴스에 연결한 HTML DOM 요소를 지정
- 여러 개의 요소에 연결할 수 없음

❑ computed 옵션

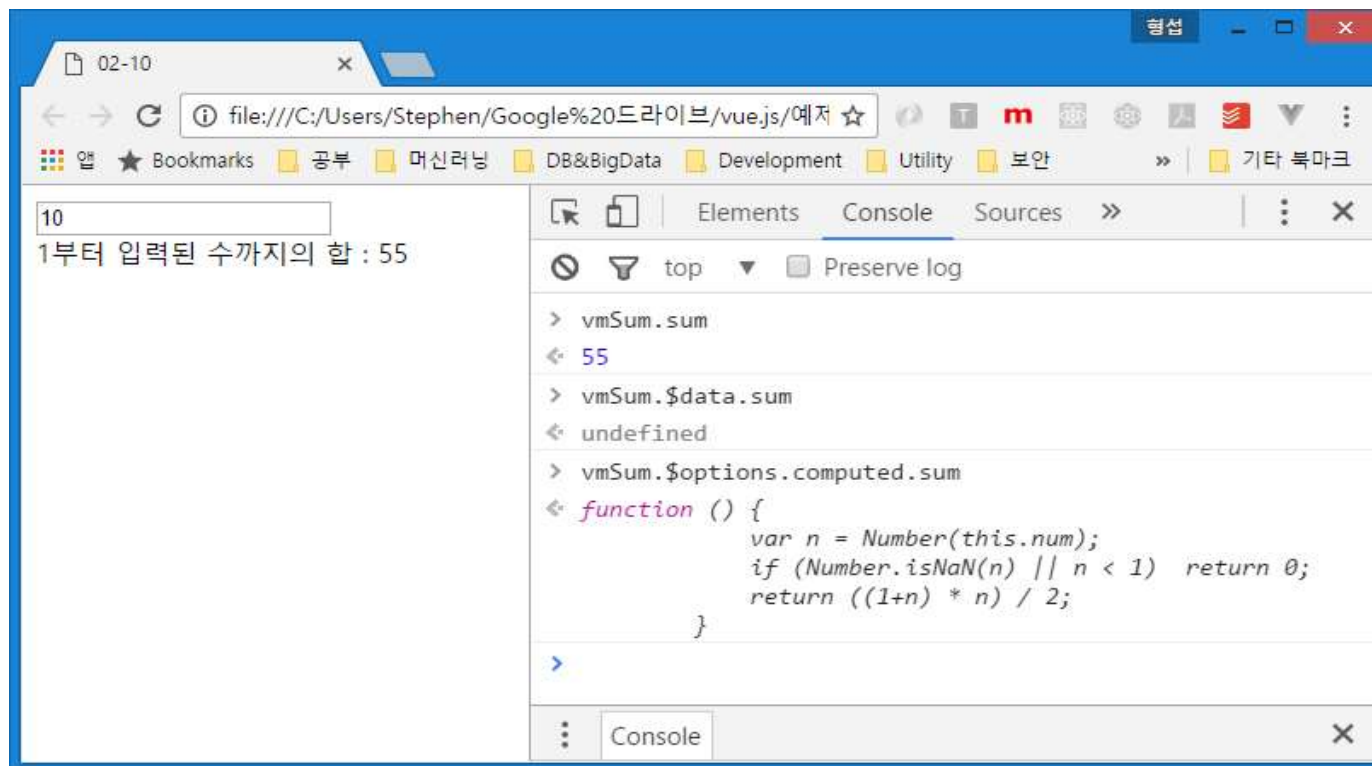
- Vue 인스턴스는 프록시 처리하여 마치 속성처럼 다룰 수 있도록 함.

el, data, computed 옵션



■ computed 옵션

- Vue 인스턴스는 프록시 처리하여 마치 속성처럼 다룰 수 있도록 함
- 예제 03-02 실행 결과



el, data, computed 옵션



예제 03-03

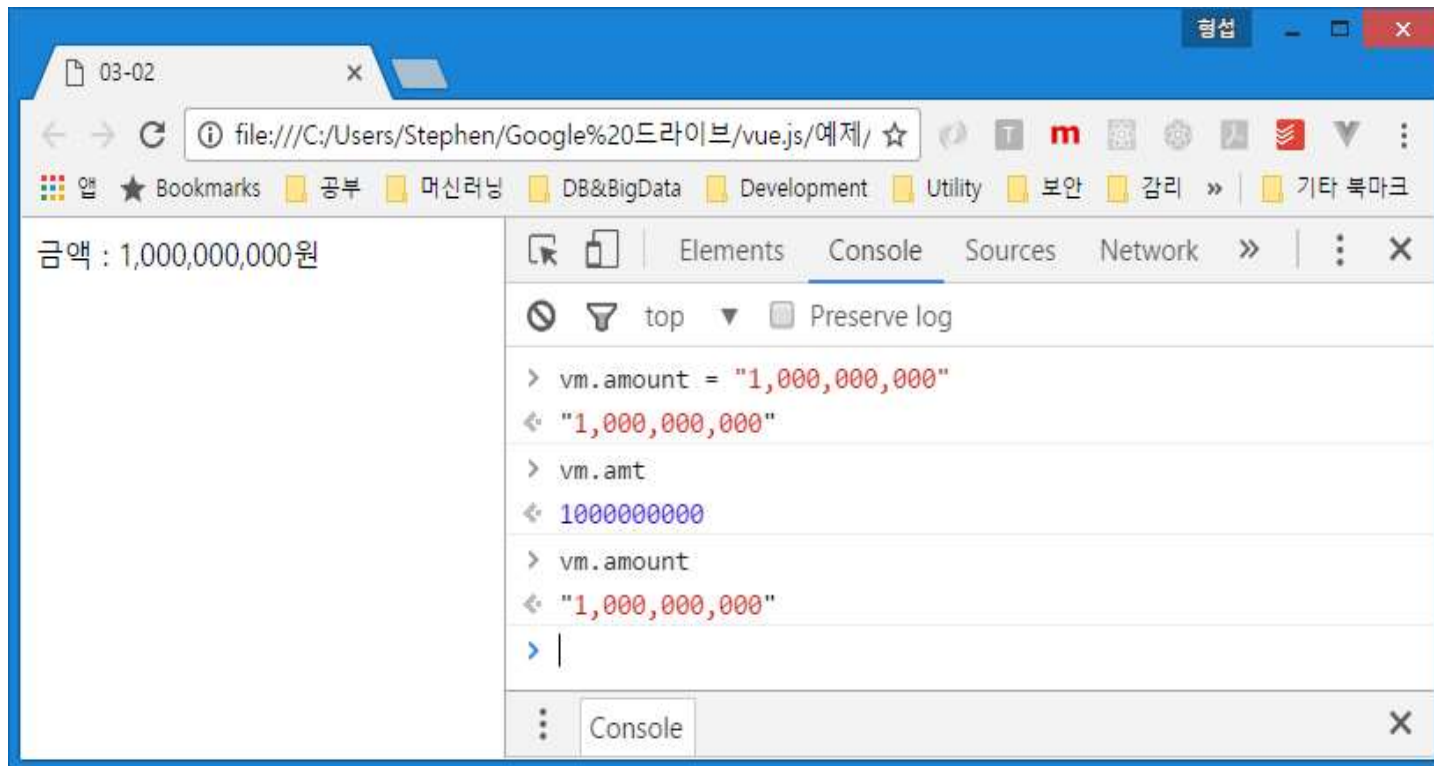
```
01: var vm = new Vue({
02:   el : "#example",
03:   data : { amt : 1234567 },
04:   computed : {
05:     amount : {
06:       get : function() {
07:         var s = new String(""+this.amt);
08:         var result = "";
09:         var num = 0;
10:         for (var i=s.length-1; i>=0; i--) {
11:           result = s[i] + result;
12:           if (num % 3 == 2 && i !== 0)
13:             result = "," + result;
14:           num++;
15:         }
16:         return result;
17:       },
18:       set : function(amt) {
19:         if (typeof(amt) === "string") {
20:           var result = parseInt(amt.replace(/,/g, ""));
21:           if (isNaN(result)) this.amt = 0;
22:           else this.amt = result;
23:         } else if (typeof(amt) === "number")
24:           this.amt = amt;
25:       }
26:     }
27:   }
28: });
```

계산형 속성이 읽기만
가능한 것은 아니다!!

el, data, computed 옵션



예제 03-03 실행 결과



메서드



☐ 메서드

- Vue 인스턴스에서 사용할 메서드를 등록
- Vue 인스턴스를 이용해 직접 호출할 수 있음
- 디렉티브 표현식, 콧수염 표현식

☐ 예제 03-04

```
01: <div id="example">
02:   <input type="text" v-model="num" /><br />
03:   1부터 입력된 수까지의 합 : <span>{{sum()}}</span>
04: </div>
05: <script type="text/javascript">
06: //1부터 입력된 수까지의 합구하기
07: var vmSum = new Vue({
08:   el : "#example",
09:   data : { num : 0 },
10:   methods : {
11:     sum : function() {
12:       var n = Number(this.num);
13:       if (Number.isNaN(n) || n < 1) return 0;
14:       return ((1+n) * n) / 2;
15:     }
16:   }
17: });
18: </script>
```

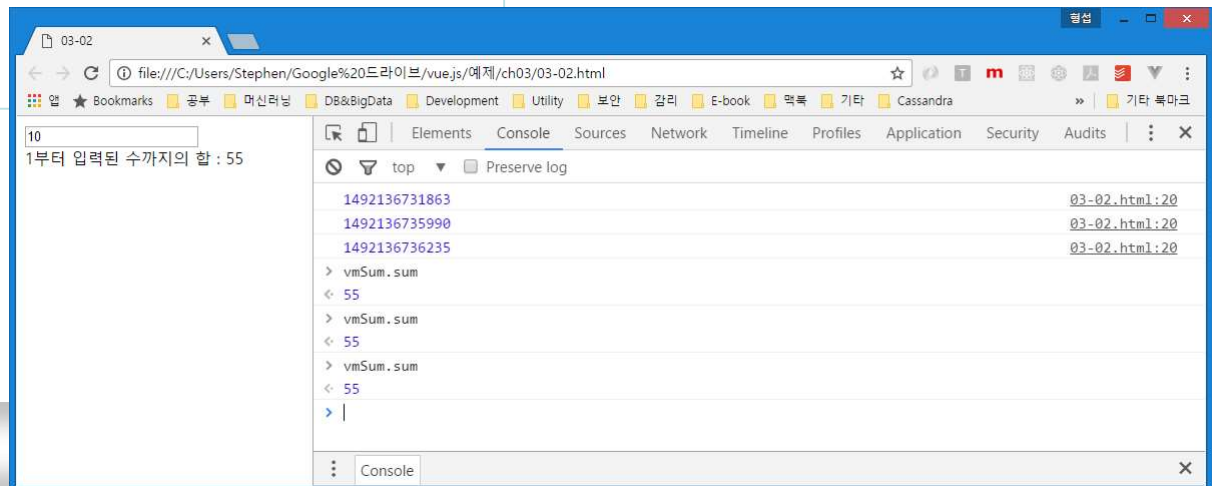
메서드



■ 계산형 속성과의 차이점

- 계산형 속성은 결과값을 캐싱함
- 메서드는 매번 호출

```
01: var vmSum = new Vue({
02:   el : "#example",
03:   data : { num : 0 },
04:   computed : {
05:     sum : function() {
06:       console.log(Date.now());
07:       var n = Number(this.num);
08:       if (Number.isNaN(n) || n < 1) return 0;
09:       return ((1+n) * n) / 2;
10:     }
11:   }
12: });
```



메서드



⚡ 주의사항

- ES6의 화살표 함수를 사용하지 않는 것이 바람직함.
- 화살표 함수 내부에서는 this가 Vue 인스턴스를 참조하지 않고 전역객체(Global Object)를 참조함
 - Vue 인스턴스 내부의 data 옵션을 이용할 수 없게 됨.

관찰 속성

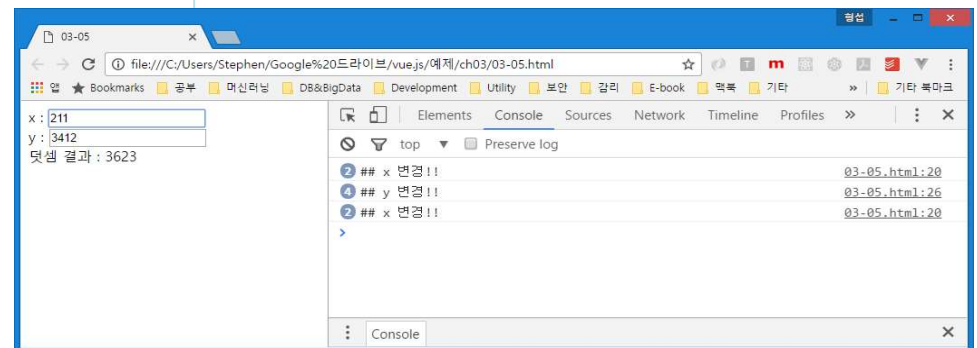


■ 관찰 속성 Watched Property

- watch 옵션을 이용해 관찰 속성을 등록함
- data 옵션의 속성값이 변경되는 것을 감지하여 등록된 함수를 호출함.

■ 예제 03-05

```
06: <script type="text/javascript">
07: var vm = new Vue({
08:   el : "#example",
09:   data : { x:0, y:0, sum:0 },
10:   watch : {
11:     x : function(v) {
12:       var result = Number(v) + Number(this.y);
13:       if (isNaN(result)) this.sum = 0;
14:       else this.sum = result;
15:     },
16:     y : function(v) {
17:       this.y = v;
18:       var result = Number(this.x) + Number(v);
19:       if (isNaN(result)) this.sum = 0;
20:       else this.sum = result;
21:     }
22:   }
23: })
24: </script>
```



관찰 속성



특징

- 값이 바뀔 때마다 매번 등록된 함수가 호출됨
- 단순한 경우라면 관찰 속성보다 계산형 속성이 더 편리하지만....
- 비동기 처리의 경우라면 관찰 속성이 더 적합함.

```
01: <script type="text/javascript">
02: var vm = new Vue({
03:   el : "#example",
04:   data : { x:0, y:0 },
05:   computed : {
06:     sum : function() {
07:       var result = Number(this.x) + Number(this.y);
08:       if (isNaN(result)) return 0;
09:       else return result;
10:     }
11:   }
12: })
13: </script>
```

관찰 속성



■ 관찰 속성을 이용한 비동기 처리를 위한 샘플 API

- <http://sample.bmaster.kro.kr>
- 1초의 지연시간 API : GET /contacts_long/search/검색어

The screenshot displays two browser windows. The left window shows the homepage of '연락처서비스 v2.0' (Contact Service v2.0), which is built with node.js + Express + sqlite3. The right window shows the API endpoint `sample.bmaster.kro.kr/contacts/search/jo` returning a JSON array of contact data.

연락처서비스 v2.0
(node.js + Express + sqlite3)

- 간단한 RESTful Service 예정입니다. 내용은 사이트 메뉴를 열어서 확인하세요.
- 이 API의 데이터는 매일 새벽2시에 초기화 되어 100건의 데이터만 남습니다.
- 사진 정보를 변경할 수 있도록 업로드 기능을 추가하였습니다.
- API를 직접 테스트해볼 수 있는 API Console 기능을 추가하였습니다. 사이트 메뉴를 확인하세요.

API Response (GET /contacts_long/search/검색어):

```
[
  {
    no: 5,
    name: "Quan Johnson",
    tel: "010-3456-8204",
    address: "서울시",
    photo: "http://sample.bmaster.kro.kr/photos/5.jpg"
  },
  {
    no: 43,
    name: "Tracy Johnson",
    tel: "010-3456-8242",
    address: "서울시",
    photo: "http://sample.bmaster.kro.kr/photos/43.jpg"
  },
  {
    no: 92,
    name: "Victoria Johnson",
    tel: "010-3456-8291",
    address: "서울시",
    photo: "http://sample.bmaster.kro.kr/photos/92.jpg"
  }
]
```

관찰 속성



예제 03-07~09

```
01: <script type="text/javascript">
02: var vm = new Vue({
03:   el : '#example',
04:   data : {
05:     name : "",
06:     isProcessing : false,
07:     contactlist : []
08:   },
09:   watch : {
10:     name : function(val) {
11:       if (val.length >= 2) {
12:         this.fetchContacts();
13:       } else {
14:         this.contactlist = [];
15:       }
16:     }
17:   },
```

```
18:   methods : {
19:     fetchContacts : _.debounce(function() {
20:       this.contactlist = [];
21:       this.isProcessing = true;
22:       var url = "http://sample.bmaster.kro.kr/contacts_long/search/"
23:         + this.name;
24:       var vm = this;
25:       fetch(url)
26:         .then(function(response) {
27:           return response.json()
28:         }).then(function(json) {
29:           vm.contactlist = json;
30:           vm.isProcessing = false;
31:         }).catch(function(ex) {
32:           console.log('parsing failed', ex);
33:           vm.contactlist = [];
34:           vm.isProcessing = false;
35:         })
36:     }, 300)
37:   }
38: },
39: })
40: </script>
```

관찰 속성



예제 03-07~09 실행 결과

03-08

file:///D:/googledrive/worksp...

이름 : jal

번호	이름	전화번호	주소
1528650003480	Blake James	010-3456-8228	서울시
1528650003542	Isabella James	010-3456-8290	서울시
1528650003518	Jain Collins	010-3456-8266	서울시
1528650003473	Maima Jackson	010-3456-8221	서울시
1528650003493	Noel James	010-3456-8241	서울시
1528650003500	Sadie Jackson	010-3456-8248	서울시

- 이 예제는 계산형 속성으로 구현 불가
 - 계산형 속성은 즉시 값을 리턴해야 하기 때문에.....
 - 계산형 속성은 동기적 처리만 수행할 수 있음

v-cloak 디렉티브



■ v-cloak 디렉티브

- 예제 03-09를 실행해보면 콧수염(Mustache) 표현식의 템플릿 문자열이 잠깐 나타났다 가 사라지는 현상이 발생할 수 있음
 - Vue 인스턴스가 el 옵션의 템플릿을 컴파일할 때 발생하는 시간 때문에 일어나는 현상
- v-cloak 디렉티브를 이용해 컴파일되지 않은 템플릿은 나타나지 않도록 처리

```
<style>
  #list { width: 600px; border:1px solid black; border-collapse:collapse; }
  #list td, #list th { border:1px solid black; text-align:center; }
  #list > thead > tr { color:yellow; background-color: purple; }
  [v-cloak] { display: none; }
</style>
<div id="example" v-cloak>
.....
</div>
```

Vue 인스턴스 라이프 사이클

