Vue.js study

2. Vue.js basic (directive)

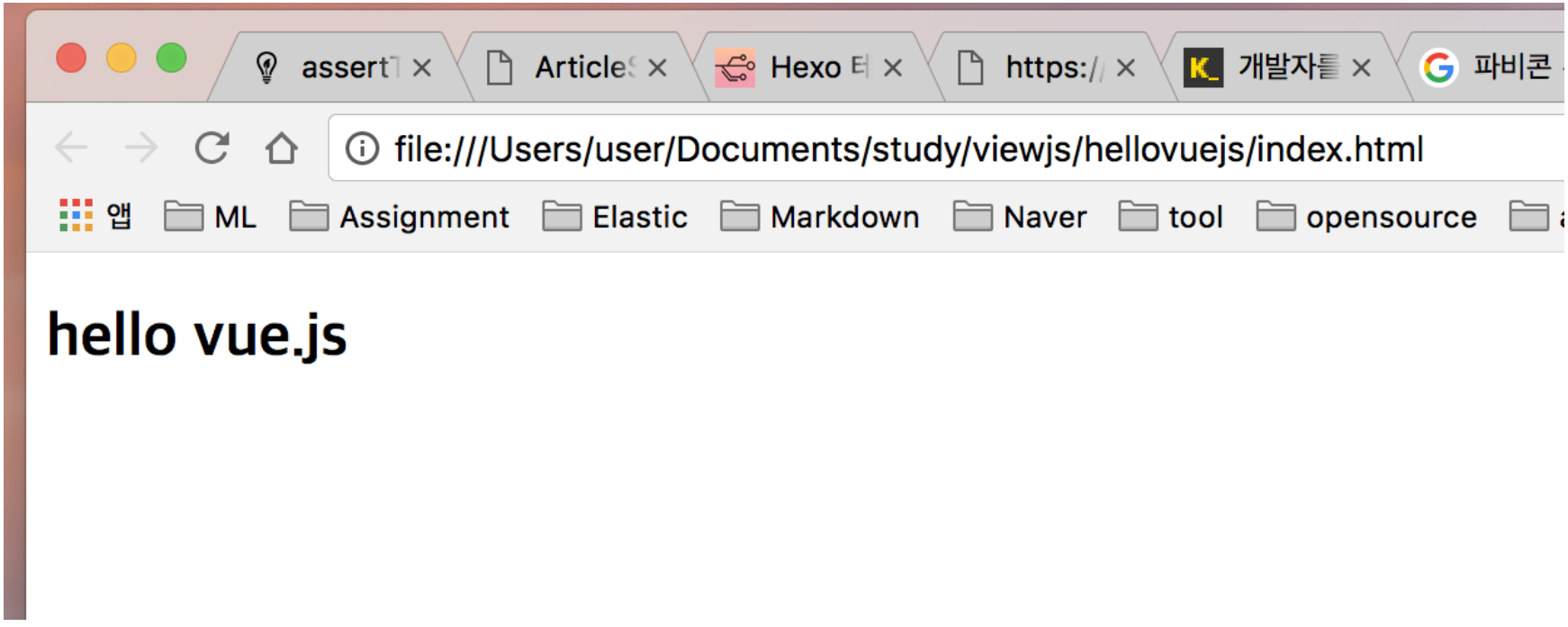
Simple example - hellovuejs

hellovuejs.html

```
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8">
    <title>Welcome to Vue</title>
    <script src="https://unpkg.com/vue"></script>
  </head>
```

```
  <body>
    <div id="simple">
      <h2>{{ message }}</h2>
    </div>
    <script type="text/javascript">
      var model = {
        message: 'hello vue.js'
      };

      var simple = new Vue({
        el: '#simple',
        data: model
      })
    </script>
  </body>

</html>
```

Run with Chrome

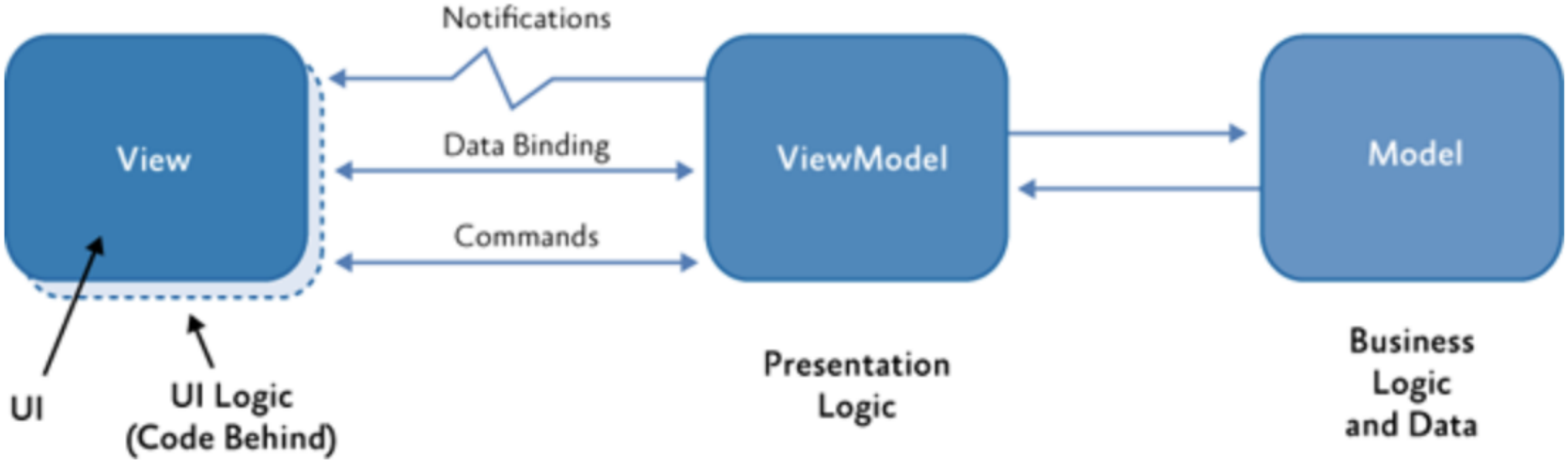


크롬 개발자 도구 (command+option+I) 의 console 탭을 사용하여 화면에 표시되는 문자를 변경할 수 있습니다.

```
> model.message = 'my first vue.js project'
```

Explanation of a simple project hellovuejs

MVVM Pattern



출처: Microsoft documentation

- 애플리케이션 로직과 UI 분리
    - 개발, 설계 문제 해결에 도움
    - 테스트, 운영, 개선이 더 쉬움
    - 코드 재사용이 쉬움
    - 개발자와 UI 디자이너 간 협업이 더 쉬움

- `View`
  - 사용자가 화면에서 보는 구조와 모양 정의
  - 경우에 따라 시각적인 동작을 위한 UI 로직이 포함될 수 있음
  - DataContext 속성을 사용해 `view model` 참조, `view` 의 control은 `view model` 을 통해 드러난 데이터
  - Data binding 정의 가능 (데이터 형식 지정, validation rules 제공 등)

- `ViewModel`
  - 기능을 위한 presentation logic encapsulation, non-visual
  - `view` 를 직접 참조하지 않음, `view` 의 data binding 을 위한 속성, 명령 구현
  - INotifyPropertyChanged, INotifyCollectionChanged interface를 사용하여 `view` 에게 상태변화를 알림
  - `view` 와 `view model` 간 상호작용 coordinate (데이터 convert, manipulate for `view` 등) -> operation
  - `view` 가 사용자에게 시각적으로 나타낼 수 있는 논리적 상태 정의 -> state

- `Model`
  - 데이터 및 비지니스 로직 encapsulation, non-visual
  - 데이터 일관성, 유효성 보장
  - `view` 와 `view model` 직접 참조하지 않음
  - INotifyPropertyChanged, INotifyCollectionChanged interface를 사용하여 상태변화를 알림 -> 쉽게 data binding 된다

Model

```
var model = {
  message: 'hello vue.js'
};
```

- data = `message`

ViewModel

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

- `model` 참조
- Data binding 을 위한 속성 구현

```
var simple = new Vue({
  el: '#simple',
  data: { num: 0 },
  computed: {
      sum: function() {
        var n = Number(this.num);
        if (Number.isNaN(n) || n < 1) return 0;
        return ((1+n)*n/2);
      }
  }
})
```

- 상호작용 coordinate operation
- view can use "num" and "sum"

View

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

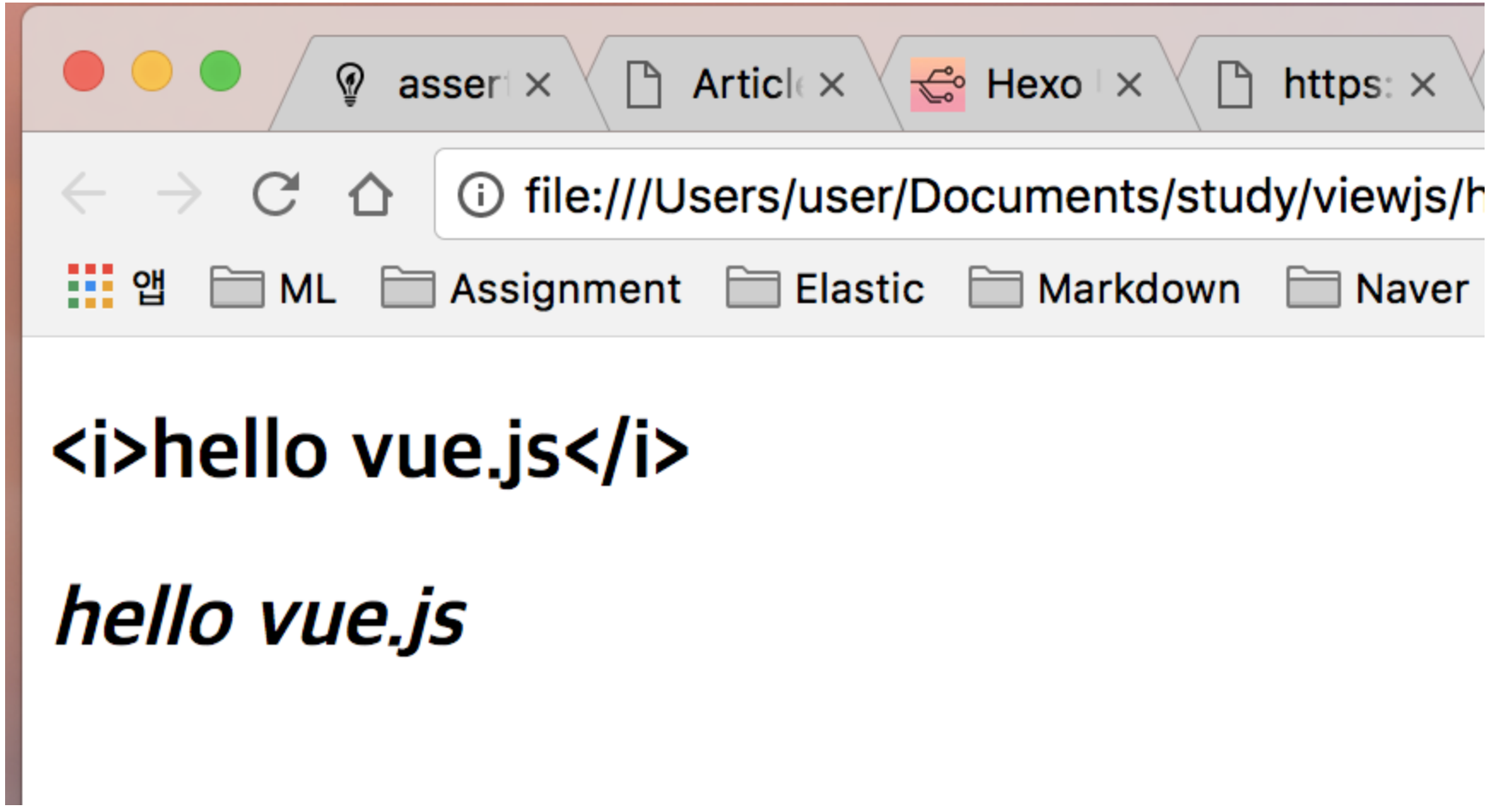
- 사용자가 화면에서 보는 구조, 모양 정의

Directives list

Element content area

- `v-text` : `innerText()` 와 유사
- `v-html` : `innerHtml()` 과 유사

```
<body>
  <div id="simple">
    <h2 v-text="message"></h2>
    <h2 v-html="message"></h2>
  </div>
  <script type="text/javascript">
    var model = {
      message: '<i>hello vue.js</i>'
    };

    var simple = new Vue({
      el: '#simple',
      data: model
    })
  </script>
</body>
```

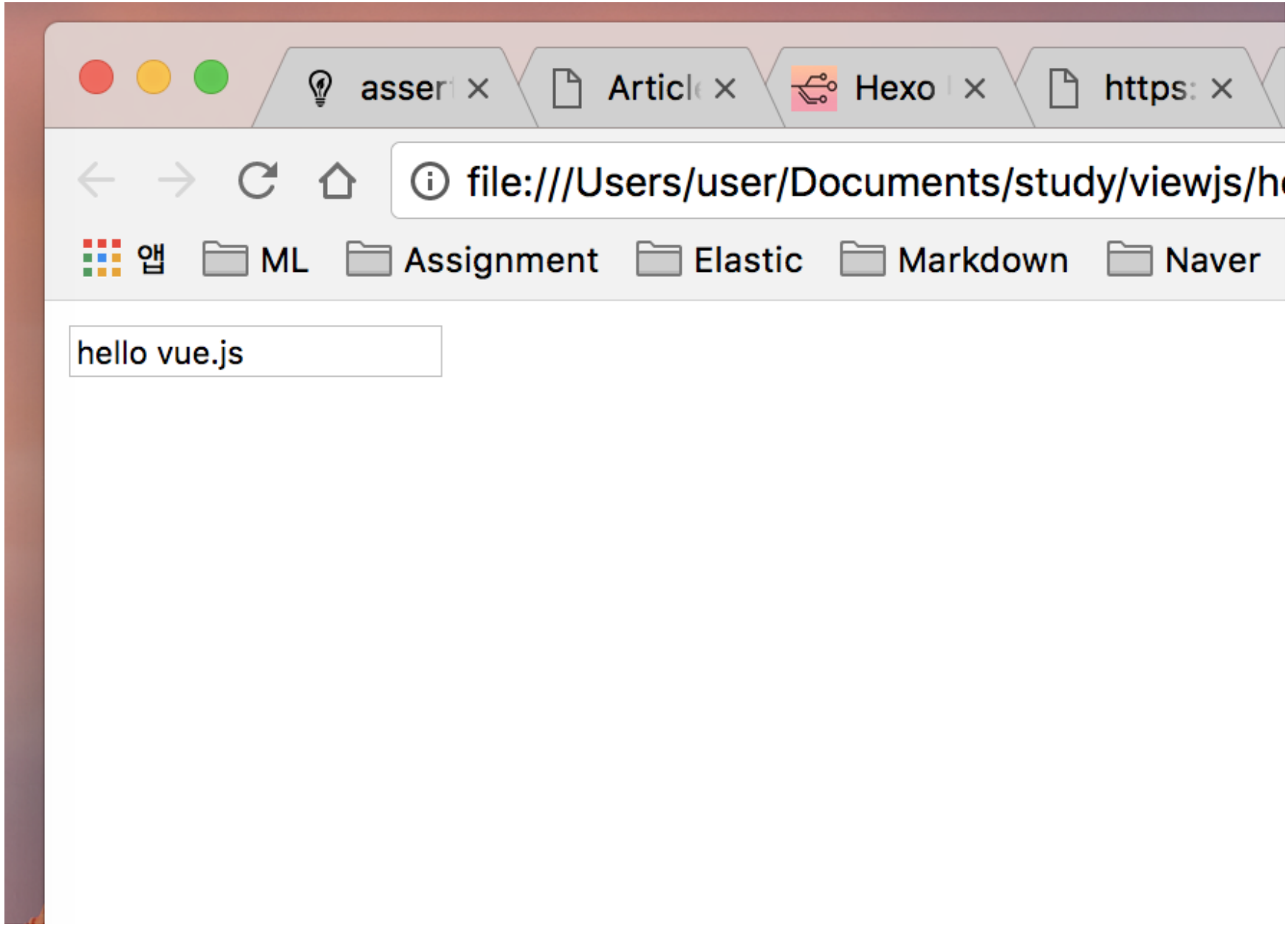

asser
Articl
Hexo
https:
file:///Users/user/Documents/study/viewjs/h
앱
ML
Assignment
Elastic
Markdown
Naver
hello vue.js
hello vue.js

Element tag attribute

- `v-bind` : 요소 태그의 속성 변경

```
<(tag) [v-bind]:(attribute)="(value)">
```

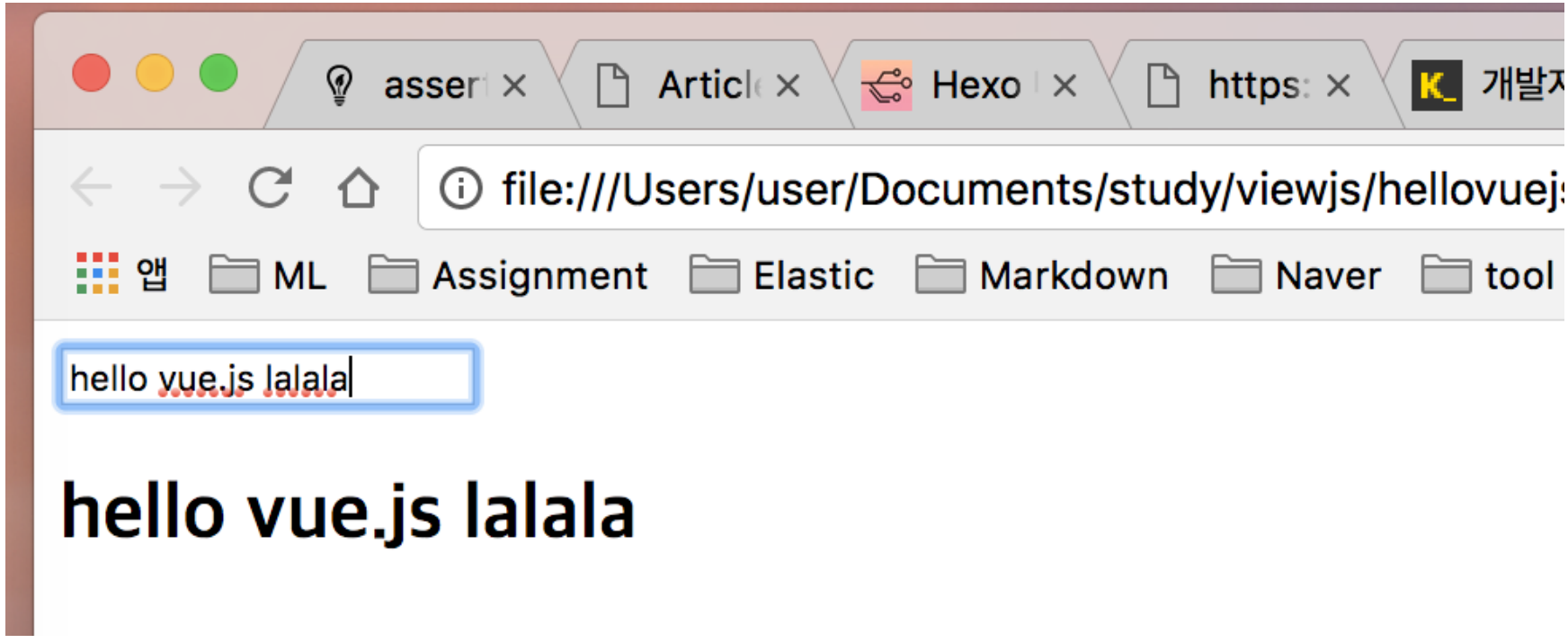
```
<body>
  <div id="simple">
    <input v-bind:value="message">
  </div>
  <script type="text/javascript">
    var model = {
      message: 'hello vue.js'
    };

    var simple = new Vue({
      el: '#simple',
      data: model
    })
  </script>
</body>
```


asser
Articl
Hexo
https:
file:///Users/user/Documents/study/viewjs/h
앱
ML
Assignment
Elastic
Markdown
Naver
hello vue.js

Two-way data binding

- `v-model` : `view` 에서 입력한 값이 `model` 에 바로 반영

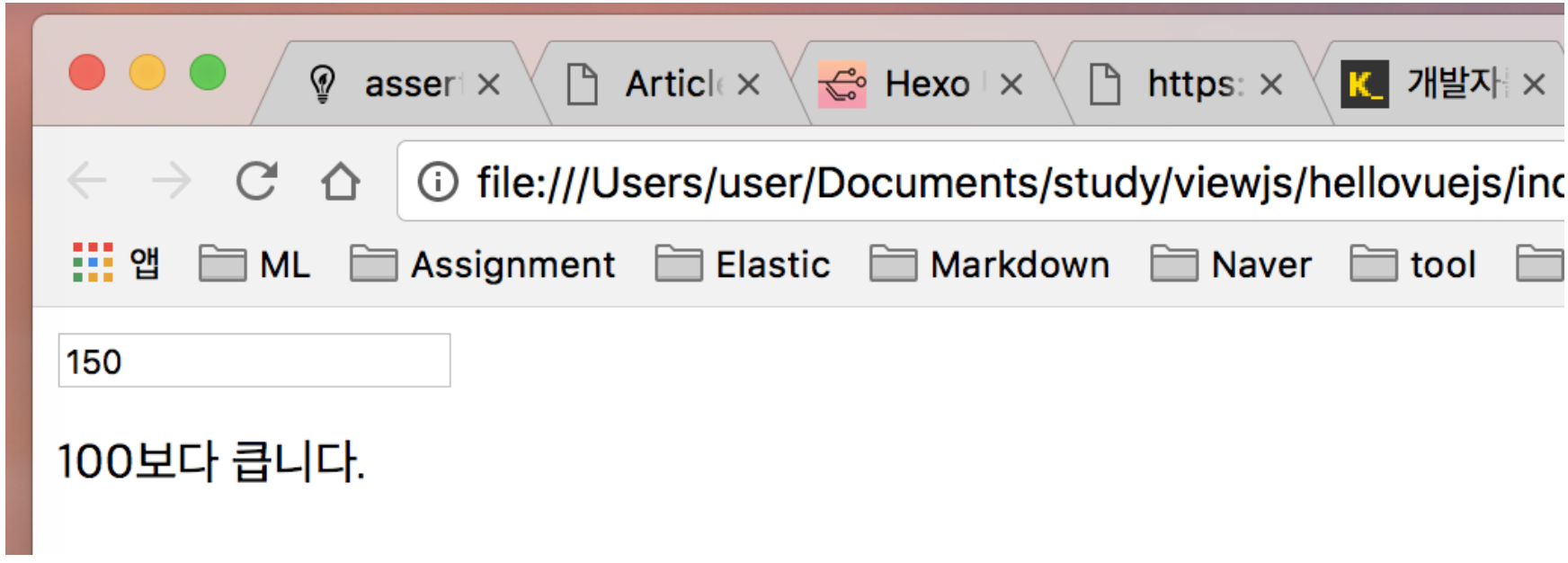
```
<body>
  <div id="simple">
    <input v-model:value="message">
    <h2 v-text="message"></h2>
  </div>
  <script type="text/javascript">
    var model = {
      message: 'hello vue.js'
    };

    var simple = new Vue({
      el: '#simple',
      data: model
    })
  </script>
</body>
```


asser
Articl
Hexo
https:
file:///Users/user/Documents/study/viewjs/hellovuej
앱
ML
Assignment
Elastic
Markdown
Naver
tool
hello vue.js lalala
hello vue.js lalala

If logic

- `v-show` : 렌더링 후 `display` 속성 사용하여 노출 여부 결정
- `v-if` , `v-else` , `v-else-if` : 조건에 부합하지 않으면 렌더링 하지 않음

```
<body>
  <div id="simple">
    <input v-model:value="amount">
    <p v-if="amount < 0">마이너스입니다.</p>
    <p v-else-if="amount > 200">200보다 큽니다.</p>
    <p v-else-if="amount > 100">100보다 큽니다.</p>
    <p v-else>마이너스가 아니고 100보다 작습니다.</p>
  </div>
  <script type="text/javascript">
    var model = {
      amount: 1000
    };

    var simple = new Vue({
      el: '#simple',
      data: model
    })
  </script>
</body>
```

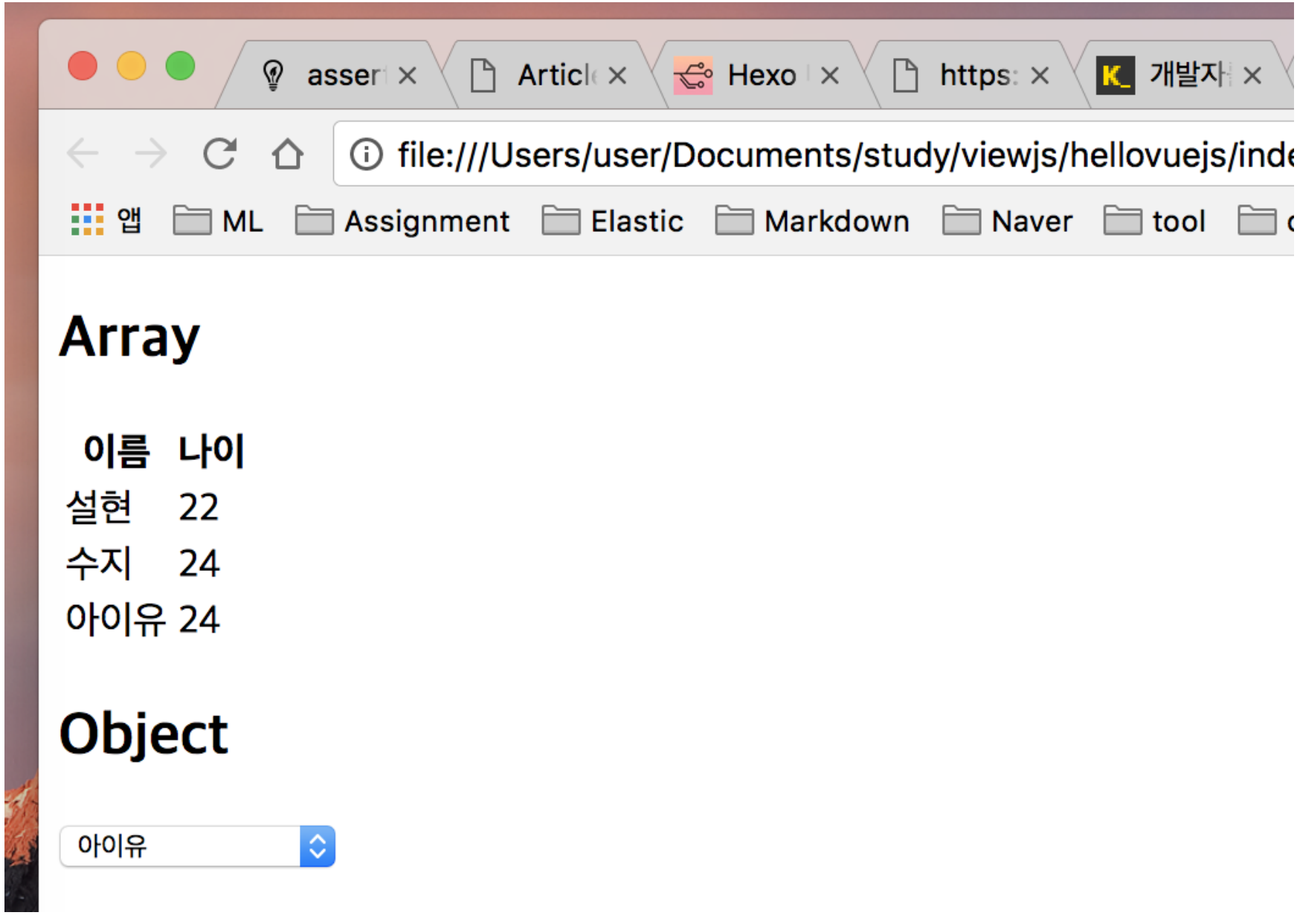

asser
Articl
Hexo
https:
개발자
file:///Users/user/Documents/study/viewjs/hellovuejs/inc
앱
ML
Assignment
Elastic
Markdown
Naver
tool
150
100보다 큽니다.

For logic

- `v-for`

```
<body>
  <div id="simple">
    <h2>Array</h2>
    <table>
      <tr>
        <th>이름</th>
        <th>나이</th>
      </tr>
      <tr v-for="item in array">
        <td v-text="item.name"></td>
        <td v-text="item.age"></td>
      </tr>
    </table>

    <h2>Object</h2>
    <select>
      <option disabled="disabled" selected>그녀를 선택하세요.<
      <option v-for="(val, key) in object" v-bind:value="
    </select>
  </div>
```

```
  <script type="text/javascript">
    var model = {
      array: [{
        "name": "설현",
        "age": "22"
      }, {
        "name": "수지",
        "age": "24"
      }, {
        "name": "아이유",
        "age": "24"
      }],
      object: {
        "A": "설현",
        "B": "수지",
        "C": "아이유"
      }
    };

    var simple = new Vue({
      el: '#simple',
      data: model
    })
  </script>
</body>
```


asser
Articl
Hexo
https:
개발자
file:///Users/user/Documents/study/viewjs/hellovuejs/inde
앱
ML
Assignment
Elastic
Markdown
Naver
tool
Array
이름 나이
설현 22
수지 24
아이유 24
Object
아이유

Other

- `v-pre` : 특수 문자열 컴파일 안 함
- `v-once` : 한 번만 렌더링

lifecycle

https://kr.vuejs.org/v2/guide/instance.html#인스턴스-라이프사이클-훅