Vue.js

starting vue.js

- 자바스크립트 프레임워크
- https://vuejs.org/
- https://github.com/vuejs/awesome-vue
- Data Driven
- Template, Directive
- Data binding
- Nuxt.js, VuePress
- Vuex, Vue Router
- https://curated.vuejs.org/
- https://element.eleme.io/
- https://onsen.io/
- Vue Devtool @chrome store

Vue.js 다운로드

- https://vuejs.org/js/vue.js

CDN 이용

```
<!-- development version, includes helpful console warnings -->
<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
```

```
<!-- production version, optimized for size and speed -->
<script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
```

Vue.js 기본 구조

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <title>Vue.js App</title>
  <link href="main.css" rel="stylesheet">
</head>
<body>
  <div id="app">
    <!-- template 출력 -->
  </div>
  <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
  <script src="main.js"></script>
</body>
</html>
```

```
var app = new Vue({
  el: '#app'
})
```

기본 기능

- 텍스트 바인딩

```
var app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue.js!'
  }
})
```

```
<p>{{ message }}</p>
```

- 반복 렌더링; v-for directive, list in data:
- 이벤트 사용; v-on directive, methods:
- 입력 양식과 동기화; v-model directive, data:
- 조건 분기; v-if directive, boolean in data:
- 트랜지션과 애니메이션; <transition>
- {디렉티브}{:매개변수}{.장식자}={값}; ex) v-bind:value.sync = “message”

Options in Vue Objects

```
var app = new Vue({

  // el
  el: '#app',
  // data
  data: {
    message: 'Vue.js'
  },
  // computed
  computed: {
    computedMessage: function() {
      return this.message + '!'
    }
  },
  // lifecycle hooks
  created: function() {
```

```
    // something to what you do
  },
  // methods
  methods: {
    myMethod: function() {
      // something to what you do
    }
  }

})
```

Lifecycles	
Methods	**Timing**
beforeCreate	인스턴스가 생성되고, 리액티브 초기화가 일어나기 전
created	인스턴스가 생성되고, 리액티브 초기화가 일어난 후
beforeMount	인스턴스가 마운트되기 전
mounted	인스턴스가 마운트된 후
beforeUpdate	데이터가 변경되어 DOM에 적용되기 전
updated	데이터가 변경되어 DOM에 적용된 후
beforeDestroy	Vue 인스턴스가 제거되기 전
destroyed	Vue 인스턴스가 제거된 후
errorCaptured	임의의 자식 컴포넌트에서 오류가 발생했을 때

Data Binding

- 템플릿에 사용하는 모든 데이터를 리액티브 데이터로 정의
- 리액티브 데이터 정의; 컴포넌트의 data 옵션에 정의
- 렌더링; mustache {{ 속성이름 }}
  - 표현식만 가능, 문장식은 불가능
  - 3항 연산자의 경우 산출 속성 computed 사용을 권장
  - 문자열이나 숫자를 변환할 때는 필터를 권장
  - 속성에는 사용 불가, 속성에 바인딩 하려면 v-bind 디렉티브 사용
  - data 상태 json으로 출력; <pre> <nowiki>
    data
    </pre>
- this; 콜백으로 익명 함수를 사용하거나, 다른 라이브러리와 함께 사용할 경우 this 변경되므로 사용에 주의
- 클래스 이름에 하이픈을 넣을 때는 작은 따옴표(')로 감쌈.
- 템플릿에서 조건 분기; v-if → 주석처리, v-show → 스타일로 보이지 않게만.
- v-if, v-else-if, v-else; key 설정
- 요소를 반복해서 렌더링; v-for = “<각 요소를 할당할 변수 이름> in <반복 대상 배열 또는 객체>” v-bind:key=“..”
  - 값, 키, 인덱스 순
  - 반복 처리 순서; Object.keys()의 순서에 기반
  - v-for 안에 v-if
- 리스트
  - 추가; push, unshift ex) this.list.push(새로운 값)
  - 제거; splice ex) this.list.splice(index, 1)

- push, pop, shift, unshift, splice, sort, reverse
- Vue.set 메서드; this.$set(변경할 데이터, 인덱스 또는 키, 새로운 값)
- this.list.filter(function(el) { ... })

v-bind 장식자	
장식자	의미
.prop	속성 대신에 DOM 속성으로 바인딩, DOM 속성과 직접 바인딩
.camel	케밥케이스[1] 속성 이름을 카멜 케이스로 변환
.sync	양방 바인딩

References

- 고양이도 할 수 있는 Vue.js 지원 페이지
- 基礎から学ぶ Vue.js 書籍用サポートページ 基礎から学ぶ Vue.js 書籍用サポートページ

1)

kebab-case, lisp-case, spinal-case; 하이픈으로 구분