

## 컴포넌트이론.txt

```

1  컴포넌트는 전역 또는 로컬에 등록 한 후, 사용자 정의 태그로 사용 가능
2
3  Vue.component 메소드를 사용해서 전역에 등록하면 자동으로 모든 컴포넌트에서 사용 가
   능
4  메소드의 매개변수로 전달되는 정보
5  1) 사용자 정의 태그로 사용할 이름
6  2) 컴포넌트의 옵션 객체
7
8  컴포넌트의 정의
9  Vue.component( 'my-component', {
10    template: '<p>MyComponent</p>'
11  })
12
13  컴포넌트 사용
14  <div id="app">
15    <my-component></my-component>
16  </div>
17
18  실제 렌더링 결과
19  <div id="app">
20    <p>MyComponent</p>
21  </div>
22
23
24  로컬에 등록
25  컴포넌트 정의를 특정한 컴포넌트의 components 옵션에 등록
26  -> 로컬로 등록된 해당 컴포넌트 범위(스코프;scope) 내부에서만 해당 컴포넌트를 사용할
   수 있도록 제한 가능
27
28  // 컴포넌트 정의
29  var myComponent = {
30    template = '<p>MyComponent</p>'
31  }
32
33  new Vue( {
34    el: '#app',
35    components: {
36      // <my-component>가 루트에서만 사용 가능
37      'my-component': myComponent
38    }
39  } );
40
41
42  컴포넌트 옵션
43  루트 생성자 new Vue() 의 옵션과 같이 컴포넌트 전용 템플릿 외에도 데이터와 메소드 정
   의 가능
44
45  Vue.component( 'my-component', {
46    //템플릿
47    template: '<p> {{ msg }} </p>',
48    // 데이터는 객체를 리턴하는 함수 지정
49    // data는 함수여야 하며, 객체를 리턴하는 함수로 정의해야 함!
50    data: function () {

```

컴포넌트이론.txt

---

```
51     return {
52       msg: 'Hello! Vue~~'
53     }
54   },
55   methods: {
56     // 메소드, 산출 속성, 와치의 정의 방법
57     // -> 루트 생성자의 객체와 동일
58   }
59 } )
60
61 ## 템플릿의 루트 요소는 반드시 하나여야만 한다!
62 // p 요소가 하나를 초과하기 때문에 사용 불가
63 template: '<p>내용1</p><p>내용2</p>'
64
65 // 여러 개의 요소가 필요한 경우 전체를 다른 요소로 감싸주면 결과적으로 1개의 요소이
   므로 조건 만족
66 template: '<div id="app"><p>내용1</p><p>내용2</p></div>'
67
68
69 컴포넌트 인스턴스(instance)
70 <div id="app">
71   <my-component></my-component>
72   <my-component></my-component>
73 </div>
74
75 컴포넌트를 여러 번 사용한 경우 my-component를 기반으로 만들어진 완전히 다른 인스턴
    스로 취급됨
76 별도의 고유 속성을 가지고 있지 않으면 두 가지는 완전히 같은 동작을 함
77
```

## 01\_컴포넌트통신.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>컴포넌트 간의 통신</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <h3>프로퍼티로 문자열 전달</h3>
12    <comp-child val="자식A"></comp-child>
13    <comp-child val="자식B"></comp-child>
14    <hr>
15    <h3>프로퍼티로 데이터 전달</h3>
16    <comp-child :val="valueA"></comp-child>
17    <comp-child :val="valueB"></comp-child>
18  </div>
19
20  <!-- Vue.js -->
21  <script
22    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
23  </script>
24    // 컴포넌트에서 프로퍼티를 전달하기 위한 props 정의
25    Vue.component( 'comp-child', {
26      // 템플릿에서 val 사용하기
27      template: '<p>{{ val }}</p>',
28      // 받을 속성 이름 지정하기
29      props: ['val']
30    } );
31
32    new Vue( {
33      el: '#app',
34      data: {
35        valueA: '자식A',
36        valueB: '자식B'
37      }
38    } );
39  </script>
40 </body>
41 </html>
```

## 02\_리스트렌더링.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>리스트 렌더링</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <ul>
12      <comp-child v-for="item in list"
13        v-bind:key="item.id"
14        v-bind:name="item.name"
15        v-bind:hp="item.hp"></comp-child>
16    </ul>
17  </div>
18
19  <!-- Vue.js -->
20  <script
21    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
22  </script>
23    // 자식 컴포넌트
24    Vue.component( 'comp-child', {
25      template: '<li>{{ name }} HP.{{ hp }}</li>',
26      props: ['name', 'hp']
27    } );
28
29    // 부모 컴포넌트
30    new Vue( {
31      el: '#app',
32      data: {
33        list: [
34          { id: 1, name: '슬라임', hp: 100 },
35          { id: 2, name: '고블린', hp: 200 },
36          { id: 3, name: '드래곤', hp: 500 }
37        ]
38      }
39    } );
40  </script>
41 </body>
42 </html>
```

## 03\_props로받을자료형지정.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>props로 받을 자료형 지정</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <example v-bind:value="value"></example>
12  </div>
13
14  <!-- Vue.js -->
15  <script
16    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
17  </script>
18    // 1~4 까지 하나씩 확인!
19    // 1. 자료형 확인을 생략하는 경우
20    // Vue.component( 'example', {
21    //   props: ['value']
22    // } );
23
24    // 2. 자료형만 확인하는 경우
25    // Vue.component( 'example', {
26    //   props: {
27    //     value: 자료형
28    //   }
29    // } );
30
31    // 3. 인스턴스를 확인하는 경우
32    // function Cat(name) {
33    //   this.name = name
34    // }
35    // Vue.component( 'example', {
36    //   props: {
37    //     value: Cat // Cat 데이터만 허가
38    //   }
39    // } );
40
41    // new Vue( {
42    //   data: {
43    //     value: new Cat('구름') // value는 Cat 데이터
44    //   }
45    // } );
46
47    // 4. 옵션도 사용하는 경우
48    // Vue.component( 'example', {
49    //   props: {
50    //     value: {
51    //       type: [String, Number], // 자료형, 배열
52    //       default: 100,           // 데이터, 함수
53    //       required: true,         // Boolean, 필수 여부
54    //       validator: function (value) { // 사용자 정의 유효성 검사 함수,
```

03\_props로받을자료형지정.html

---

```
        Boolean 값 리턴
53      //      return value > 10
54      //      }
55      //      }
56      //      }
57      //      } );
58  </script>
59 </body>
60 </html>
```

## 04\_자식이벤트를부모에서잡기.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>자식 이벤트를 부모에서 잡기</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <!-- 부모 템플릿 -->
12    <comp-child v-on:childs-event="parentsMethod">□</comp-child>
13  </div>
14
15  <!-- Vue.js -->
16  <script
17    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
18  <script>
19    // 자식 컴포넌트
20    Vue.component( 'comp-child', {
21      template: '<button v-on:click="handleClick">이벤트</button>',
22      methods: {
23        // 버튼 클릭 이벤트 핸들러로 childs-event 실행
24        handleClick: function () {
25          this.$emit('childs-event')
26        }
27      }
28    });
29    // 부모 컴포넌트
30    new Vue( {
31      el: '#app',
32      methods: {
33        // childs-event가 실행되었을 경우 실행
34        parentsMethod: function () {
35          alert('자식에서 전달받은 이벤트!')
36        }
37      }
38    });
39  </script>
40 </body>
41 </html>
```

## 05\_부모데이터조작.html

```

1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>부모 데이터 조작</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <ul>
12      <comp-child v-for="item in list"
13        v-bind:key="item.id"
14        v-bind="item"
15        v-on:attack="handleAttack"></comp-child>
16    </ul>
17  </div>
18
19  <!-- Vue.js -->
20  <script
21    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
22  </script>
23  // 자식 컴포넌트
24  Vue.component( 'comp-child', {
25    template: '<li>{{ name }} HP.{{ hp }}\
26      <button v-on:click="doAttack">공격하기</button></li>',
27    props: {
28      id: Number,
29      name: String,
30      hp: Number
31    },
32    methods: {
33      // 버튼 클릭 이벤트 핸들러에서 $emit을 호출해서 attack 이벤트 실행
34      doAttack: function () {
35        // 매개 변수로 자신의 ID 전달
36        this.$emit('attack', this.id)
37        // $emit : 컴포넌트에 연결되어 있는 이벤트를 명시적으로 실행시키는 메소드
38        // jQuery의 trigger()와 유사함
39      }
40    }
41  });
42
43  new Vue( {
44    el: '#app',
45    data: {
46      list: [
47        { id: 1, name: '슬라임', hp: 100 },
48        { id: 2, name: '고블린', hp: 200 },
49        { id: 3, name: '드래곤', hp: 500 }
50      ]
51    },
52    methods: {
53      // attack가 실행된 경우

```



05\_부모데이터조작.html

---

```
53     handleAttack: function (id) {
54         // 매개 변수의 ID를 기반으로 요소 검색
55         var item = this.list.find(function (el) {
56             return el.id === id
57         })
58         // HP가 0보다 크면 지정한 수치만큼 줄이기
59         if (item !== undefined && item.hp > 0) item.hp -= 70
60     }
61 }
62 } );
63 </script>
64 </body>
65 </html>
```

## 06\_컴포넌트간통신.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>컴포넌트 간 통신</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    부모 자식 관계가 아닌 경우 컴포넌트 사이의 통신
12  </div>
13
14  <!-- Vue.js -->
15  <script
16    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
17  <script>
18    // 이벤트 bus 전용 인스턴스 생성
19    var bus = new Vue( {
20      data: {
21        count: 0
22      }
23    } );
24
25    Vue.component( 'component-b', {
26      template: '<p> bus: {{ bus.count }} </p>',
27      computed: {
28        // bus 데이터를 산출 속성으로 사용
29        bus: function () {
30          return bus.$data
31        }
32      },
33      created: function () {
34        bus.$on( 'bus-event', function () {
35          // bus-event 이벤트가 발생할 때 처리
36          // 익명 함수 내부의 this는 bus의 인스턴스
37          this.count++
38        } )
39      }
40    } );
41  </script>
42 </body>
43 </html>
```

## 07\_자식컴포넌트참조.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>자식 컴포넌트를 참조하는 $refs</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <comp-child ref="child"></comp-child>
12  </div>
13
14  <!-- Vue.js -->
15  <script
16    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
17  <script>
18    // 자식 컴포넌트
19    Vue.component( 'comp-child', {
20      template: '<div>...</div>',
21      created: function () {
22        // 자신의 처리
23        this.$on('open', function () {
24          console.log('무언가 처리하기')
25        })
26      }
27    });
28
29    // 부모 컴포넌트
30    new Vue({
31      el: '#app',
32      methods: {
33        handleClick: function () {
34          // 자식 컴포넌트의 이벤트 실행
35          this.$refs.child.$emit('open')
36        }
37      }
38    });
39  </script>
40 </body>
41 </html>
```

## 08\_사용자정의컴포넌트-슬롯.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title></title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <!-- 부모 컴포넌트 / 슬롯 콘텐츠 정의 -->
11  <comp-child>
12    <header slot="header">
13      Hello Vue.js!
14    </header>
15    Vue.js는 JavaScript 프레임워크입니다.
16  </comp-child>
17
18  <!-- 자식 컴포넌트 / 슬롯 사용 -->
19  <section class="comp-child">
20    <slot name="header">
21      <header>
22        디폴트 타이틀
23      </header>
24    </slot>
25    <div class="content">
26      <slot>디폴트 콘텐츠</slot>
27    </div>
28    <slot name="footer">
29      <!-- 없다면 아무 것도 출력하지 않습니다. -->
30    </slot>
31  </section>
32
33  <!-- Vue.js -->
34  <script
35    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
36  <script>
37    var app = new Vue( {
38      el: '#app',
39    } );
40  </script>
41 </body>
42 </html>
```

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>v-model 사용자 정의</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11
12  </div>
13
14  <!-- Vue.js -->
15  <script
16    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
17  <script>
18    Vue.component( 'my-calendar', {
19      model: {
20        // 현재 값을 value가 아니라 current로 할당하고 싶은 경우
21        prop: 'current',
22        // 이벤트를 change로 사용하고 싶은 경우
23        event: 'change'
24      },
25      // props에서 설정하기
26      props: {
27        current: String
28      },
29      created: function () {
30        this.$emit('change', '2021-01-01')
31      }
32    } );
33  </script>
34 </body>
35 </html>
```

## 10\_양방향데이터바인딩-vsync.html

```

1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>양방향 데이터 바인딩 - .vsync</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <my-component
12      v-bind:name.sync="name"
13      v-bind:hp.sync="hp">
14    </my-component>
15  </div>
16
17  <!-- Vue.js -->
18  <script
19    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
20  <script>
21    /* 자식 컴포넌트 */
22    Vue.component('my-component', {
23      template: '<div class="my-component">\
24        <p>이름.{{ name }} HP.{{ hp }}</p>\
25        <p>이름 <input v-model="localName"></p>\
26        <p>HP <input size="5" v-model.number="localHp"></p>\
27        </div>',
28      props: {
29        name: String,
30        hp: Number
31      },
32      computed: {
33        // 산출 속성의 세터와 게터를 통해 v-model 사용
34        localName: {
35          get: function () {
36            return this.name
37          },
38          set: function (val) {
39            this.$emit('update:name', val)
40          }
41        },
42        localHp: {
43          get: function () {
44            return this.hp
45          },
46          set: function (val) {
47            this.$emit('update:hp', val)
48          }
49        }
50      }
51    });
52    /* 부모 컴포넌트 */

```

## 10\_양방향데이터바인딩-vsync.html

---

```
53     new Vue( {  
54         el: '#app',  
55         data: {  
56             name: '슬라임',  
57             hp: 100  
58         }  
59     } );  
60 </script>  
61 </body>  
62 </html>
```

## 11\_함수형컴포넌트.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>함수형 컴포넌트</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11
12  </div>
13
14  <!-- Vue.js -->
15  <script
16    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
17  <script>
18    Vue.component( 'functional-component', {
19      functional: true,
20      render: function (createElement, context) {
21        return createElement('div', context.props.message)
22      },
23      props: {
24        message: String
25      }
26    } );
27  </script>
28 </body>
29 </html>
```



## 12\_동적컴포넌트.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>동적 컴포넌트</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <button v-on:click="current^=1">컴포넌트변경</button>
12
13    <div v-bind:is="component"></div>
14  </div>
15
16  <!-- Vue.js -->
17  <script
18    src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
19  <script>
20    /* 자식 컴포넌트 */
21    // 컴포넌트A
22    Vue.component( 'my-component-a', {
23      template: '<div class="my-component-a">component A</div>'
24    } );
25    // 컴포넌트B
26    Vue.component( 'my-component-b', {
27      template: '<div class="my-component-b">component B</div>'
28    } );
29
30    /* 부모 컴포넌트 */
31    new Vue( {
32      el: '#app',
33      data: {
34        // 컴포넌트 리스트
35        componentTypes: ['my-component-a', 'my-component-b'],
36        // 렌더링할 컴포넌트를 선택하는 index
37        current: 1
38      },
39      computed: {
40        component: function () {
41          // current와 일치하는 index 컴포넌트 사용
42          return this.componentTypes[this.current]
43          // `return current ? 'my-component-b' : 'my-component-a'` 사용 가능
44        }
45      }
46    } );
47  </script>
48 </body>
49 </html>
```

## 13\_공통처리-Mixin.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>공통 처리 - Mixin</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10
11   <div id="app">
12
13   </div>
14
15   <!-- Vue.js -->
16   <script
17     src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
18   <script>
19     /* 믹스인 정의 */
20     var mixin = {
21       created: function () {
22         this.hello()
23       },
24       methods: {
25         hello: function () {
26           console.log('hello from mixin!')
27         }
28       }
29     }
30
31     /* 믹스인 사용 */
32     Vue.component( 'my-component-a', {
33       mixins: [mixin], // 믹스인 등록
34       template: '<p>MyComponentA</p>'
35     } );
36     Vue.component( 'my-component-b', {
37       mixins: [mixin], // 믹스인 등록
38       template: '<p>MyComponentB</p>'
39     } );
40   </script>
41 </body>
42 </html>
```

## 14\_상태유지-keep-alive.html

```
1 <!DOCTYPE html>
2 <html lang="ko">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>상태 유지 - keep-alive</title>
7   <link rel="stylesheet" href="../css/main.css">
8 </head>
9 <body>
10  <div id="app">
11    <button v-on:click="current='comp-board'">메시지 목록</button>
12    <button v-on:click="current='comp-form'">입력 양식</button>
13
14    <keep-alive>
15      <div v-bind:is="current"></div>
16    </keep-alive>
17  </div>
18
19  <!-- Vue.js -->
20  <script
21  src='https://cdnjs.cloudflare.com/ajax/libs/vue/2.6.9/vue.min.js'></script>
22  <script>
23    /* 자식 컴포넌트 */
24    // 메시지 목록 전용 컴포넌트
25    Vue.component( 'comp-board', {
26      template: '<div>Message Board</div>',
27    });
28    // 입력 양식 전용 컴포넌트
29    Vue.component( 'comp-form', {
30      template: '<div>Form<textarea v-model="message"></textarea></div>',
31      data: function () {
32        return {
33          message: ''
34        }
35      }
36    });
37
38    /* 부모 컴포넌트 */
39    var app = new Vue( {
40      el: '#app',
41      data: {
42        current: 'comp-board' // 동적으로 변경
43      }
44    });
45  </script>
46 </body>
47 </html>
```