**1. 컴포넌트 설계가 갖는 장점은 무엇인가. 컴포넌트란 무엇인가. 컴포넌트 설계는 어떻게 해야 하는가. 컴포넌트의 리랜더링은 언제 이루어지는가.**

**- 컴포넌트 설계 장점**

1. 가독성이 매우 높아집니다.

2. 유지보수가 쉬워집니다.

3. UI 수정 및 재사용이 쉬워집니다.

한개의 파일내에 수많은 코드들을 컴포넌트 단위로 분리하여, 한눈에 보이는 구조로 작업이 가능합니다. 또한 표현,제어 컴포넌트 등을 분리시킴으로 기능의 문제, 표현의 문제 등의 유지보수가 쉬워집니다.

**- 컴포넌트란?**

독립적으로도 사용할 수 있으며 제 3의 컴포넌트와 잘맞는 소프트웨어의 부품.

근본적으로 컴포넌트란 데이터를 입력받아[1] DOM Node를[2] 출력하는 함수라고 생각 합니다.

**React 공식 문서**인 “**Components and Props**“를 보면 이런 문장이 있다고 합니다.

「 Conceptually, components are like JavaScript functions. They accept arbitrary inputs (called “props”) and return React elements describing what should appear on the screen. 」

개념적으로 컴포넌트는 JavaScript의 함수와 같다고 합니다.

[1] - 입력받는 데이터란 Props나 State 같은 것들 입니다.

[2] - DOM tree[3] 구조에서 모든 각각의 개체를 node라고 표현합니다. head, body, title, script, h1, HEADER-1 등의 태그뿐 아니라 태그 안의 텍스트나 속성 등도 모두 node에 속합니다.

[3] - 최상단인 DOM 에서부터 나무처럼 뿌리와 가지들이 뻗어가며 node 들을 표현하는 자료구조

**- 컴포넌트 설계는 어떻게 해야 하는지?**

**1. 이름 붙이기**

- 컴포넌트 설계의 시작은 각 컴포넌트들의 이름을 명확히 지어야 합니다.

**2. 단일책임원칙**

( 모든 클래스는 하나의 책임만 가지며, 클래스는 그 책임을 완전히 캡슐화해야 함을 일컫는다. )

- 하나의 컴포넌트는 한 가지 일을 하는게 이상적이라는 원칙 입니다. 하나의 컴포넌트가 커지게 된다면 이는 보다 작은 하위 컴포넌트로 분리되어야 합니다.

**- 컴포넌트의 리렌더링은 언제 이루어 지는지?**

React 컴포넌트가 렌더링을 수행하는 시점은 다음과 같습니다.

1. Props가 변경되었을 때

2. State가 변경되었을 때

3. forceUpdate() 를 실행하였을 때

4. 부모 컴포넌트가 렌더링되었을 때

1~3번의 과정을 통해 컴포넌트가 렌더링될 때, 자식 컴포넌트 또한 같은 과정으로 렌더링이 진행됩니다.

즉, 데이터가 업데이트 되었을때 리렌더링이 이루어 집니다.

**출처** :

컴포넌트란 ? : <https://hyunseob.github.io/2019/06/02/react-component-the-right-way/>

DOMTree 란 ? : <https://m.blog.naver.com/magnking/220972680805>

컴포넌트설계란 ? : <https://ko.reactjs.org/docs/thinking-in-react.html>

렌더링은 언제 이루어지는지? : <https://medium.com/vingle-tech-blog/react-%EB%A0%8C%EB%8D%94%EB%A7%81-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0-f255d6569849>

**2. 리액트와 뷰의 차이점과 공통점은 무엇인가.**

- 공통점

1. 컴포넌트 기반 SPA 라이브러리

- SPA : Single Page Application 새로고침 없이 변동사항의 컴포넌트만 업데이트 하는 기술.

2. virtual DOM을 기반으로 하며, tree 알고리즘을 통해 변화사항을 찾아낸다 ( diff )

3. 위와같은 내용들로 인해 제한사항도 닮은점이 있습니다.

3-1 컴포넌트의 HTML에 해당하는 코드는 단일 노드로 감싸져 있어야 한다.

3-2 목록을 출력할때 각각의 항목이 구분될 수 있는 고유한 문자열을 할당해주어야 한다.(key Props) 등.

4. props, state, lifecycle

- 둘 다 컴포넌트 단위로 코드를 작성하므로 기본적인 구성요소도 겹치는 것들이 있습니다. 대표적으로 props, state, lifecycle 입니다.

명칭이나 사용법 등은 조금씩 다르지만 지향점은 같습니다.

5. 라우터, 전역 상태관리 도구

SPA를 통해 여러 페이지를 보여주기 위한 라우터 역시 둘 다 존재합니다.

6. cli, webpack , ...

둘 다 기본 프로젝트 구조를 만들어주는 툴을 cli로 제공하고 있습니다. webpack 기반으로 번들링 해주는 기능 역시 존재합니다.

CLI : Command Line Interface의 약자

7. mobile application을 위한 Native 지원

8. SSR Framework

둘 다 서버사이드 렌더링용 웹서버 프레임워크가 있습니다. WebPack 기반 자바스크립트이며, 모두 nodeJS 기반입니다.

출처 : <https://velog.io/@vraimentres/react-vs-vue-1>

- 차이점

1. 전역상태관리 도구

react는 3rd party library 에서 지원하고 있으며, vue는 공식 라이브러리도 함께 만들어서 제공 합니다.

전역 상태관리 도구 역시 react는 제 3자에 의해 redux 같은것이 만들어졌으며, 후발주자인 vue는 시작할 때 부터 redux 와 redux-saga가 합쳐진 형태의 전역 상태관리 도구인 vuex를 만들어서 공식으로 제공하였습니다.

2. mobile Native 지원

1번과는 정반대로 reactNative는 페이스북에서 공식적으로 지원하는 라이브러리이며

뷰 네이티브는 서드파티 라이브러리 입니다.

3. SSR Framework

react 는 nextJs 를 사용하며 뷰는 nuxtJs 를 사용합니다.

4. Mutate Data

데이터를 변경할때 차이점이 발생합니다.

vue : 반드시 데이터 객체를 생성한 후에 data를 자유롭게 업데이트할 수 있습니다.

react : state 객체를 만들고 업데이트를 하려면 조금 더 많은 작업이 필요합니다.

즉, vue 에서는 도우미 없이도 값을 바꿀 수 있도록 setState를 알아서 결합해주며, react에서는 state 값의 변경을 돕는 도우미 setState() 가 필요합니다. (권장사항)

출처 : <https://medium.com/@erwinousy/%EB%82%9C-react%EC%99%80-vue%EC%97%90%EC%84%9C-%EC%99%84%EC%A0%84%ED%9E%88-%EA%B0%99%EC%9D%80-%EC%95%B1%EC%9D%84-%EB%A7%8C%EB%93%A4%EC%97%88%EB%8B%A4-%EC%9D%B4%EA%B2%83%EC%9D%80-%EA%B7%B8-%EC%B0%A8%EC%9D%B4%EC%A0%90%EC%9D%B4%EB%8B%A4-5cffcbfe287f>

**3. STATE와 PROPS를 설명하시오.**

두 객체 모두 렌더링 결과물에 영향을 주는 정보를 갖고 있으며, 한가지 차이가 있습니다.

props는 컴포넌트에 전달되는 반면 ( 함수의 매게변수처럼 ) , state 는 컴포넌트 안에서 관리됩니다. ( 함수 내에 선언된 변수처럼 )

props 는 수정이 불가능한 일기전용 속성 입니다.

출처 : <https://ko.reactjs.org/docs/faq-state.html>