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

React의 컴포넌트는 개념적으로 props를 input으로 하고 UI가 어떻게 보여야 하는지 정의하는 React Element를 output으로 하는 함수이다. 컴포넌트는 두 가지 인스턴스 속성(property) props와 state를 가지고 있다. props는 컴포넌트의 mounting, updating 프로세스 시점에 값이 할당될 뿐 컴포넌트 내부에서 값을 변경할 수 없다. 상황에 따라 변경되어야 하는 값들은 state를 이용해야한다. 특정 state, props에 따른 render 결과가 바뀌지 않는다.

출처: <https://medium.com/little-big-programming/react%EC%9D%98-%EA%B8%B0%EB%B3%B8-%EC%BB%B4%ED%8F%AC%EB%84%8C%ED%8A%B8%EB%A5%BC-%EC%95%8C%EC%95%84%EB%B3%B4%EC%9E%90-92c923011818>

Uncontrolled component

Uncontrolled Component는 상태를 사용자가 직접 React에서 제어하지 않는 컴포넌트다. React 생태계에서는 잘 쓰이지 않으나, 렌더를 아예 타지않는다는 장점이 있기 때문에 상태를 제어할 일이 없다면 쓰는 것도 좋다. Uncontrolled Component는 key Props를 이용해 초기화 할 수 있다.

Controlled Component

Controlled Component는 사용자가 상태를 제어할 수 있는 컴포넌트다.

PureComponent

Component는 항상 render를 다시 실행하지만 PureComponent는 Props나 State를 얕은 비교해서 이전과 같으면 render를 다시 실행하지 않는다. PureComponent를 이용해 render 를 최적화 할 수 있다. 하지만, PureComponent를 잘못 사용하면 일반적인 Component보다도 성능이 나빠질 수 있다. 만약 항상 re-render 되는 PureComponent가 있으면 불필요한 비교 로직이 들어가기 때문에 더 느리다. PureComponent를 잘 사용하면 렌더링 최적화와 Controlled Component라는 두 마리 토끼를 잡을 수 있다.

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

Components Re-renders

컴포넌트의 상태(state)가 변하면, 컴포넌트의 리렌더링을 발생(trigger)시킨다. 구체적으로 아래와 같다.

- 자신의 상태(state)가 변경될 때

- 부모 컴포넌트가 리렌더링될 때

- 자신이 전달받은 props가 변경될 때

- forceUpdate 함수가 실행될 때

출처: <https://code-masterjung.tistory.com/m/67>

**2. 리액트와 뷰의 차이점과 공통점은 무엇인가?**

React와 Vue는 컴포넌트 기반, Virtual DOM 방식이라는 점에 공통점이 있고, 가볍고 빠르다는 평을 받습니다. 두 프레임워크의 공통된 속성은 아래와 같습니다.

컴포넌트 기반

React와 Vue 모두, 웹 UI를 작은 컴포넌트 단위로 구성합니다. 컴포넌트는 다른 프로젝트에서도 재사용할 수 있고, 컴포넌트 캡슐화와 확장이 가능해 개발이 유연해지는 장점이 있습니다.

Virtual DOM

Virtual DOM은 실제 DOM 변화를 최소화 시켜주는 역할을 합니다.

먼저 DOM에 대해 잠시 언급하자면, 브라우저는 HTML 파일을 스크린에 보여주기 위해 DOM 노드 트리 생성, 렌더트리 생성, 레이아웃, 페인팅 과정을 거칩니다. DOM 노드는 HTML의 각 엘리먼트와 연관되어 있기 때문에 HTML 파일에 20개의 변화가 생기면 DOM 노드가 변경되고 그 이후의 과정역시 20회 다시 이루어 집니다. 작은 변화에도 매우 복잡한 과정들이 다시 실행되기 때문에 DOM 변화가 잦을 경우 성능이 저하됩니다.

Virtual DOM은 뷰에 변화가 있다면, 그 변화가 실제 DOM에 적용되기 전에 Virtual DOM에 적용시키고 최종 결과만 실제 DOM에 전달합니다. 따라서 20개의 변화가 있다면 Virtual DOM은 변화된 부분만 가려내어 실제 DOM에 전달하고 실제 DOM은 그 변화를 1회로 인식하여 단 한번의 렌더링 과정만 거치게 됩니다.

React와 Vue는 모두 Virtual DOM 방식을 통해 성능을 향상 시킵니다. 위와 같이 두 프레임워크는 공통된 기반을 공유하고 있어 유사한 측면이 많지만, 각각 구별되는 특징 또한 존재합니다.

**Vue**

단일 파일 컴포넌트(Single File Component)

Vue는 단일 파일 컴포넌트 기반의 프레임워크입니다. 웹의 뷰(view)를 구성하는 요소인 HTML, CSS, JavaScript 코드를 .vue 확장자를 가진 하나의 파일에 모두 정의하는 방식입니다. 이러한 관리 방식은 적당한 규모의 프로젝트에서 관리의 생산성을 높이고, 협업을 수월하게 한다는 장점이 있습니다.

HTML 기반 템플릿 구문

Vue는 개발한 프론트엔드 파일을 사용자가 볼 수 있도록 브라우저 화면에 렌더링하는 과정에 템플릿이란 문법을 사용합니다. 이 템플릿을 구성하는 문법이 HTML 기반으로 이뤄져 있어 배우기 쉽다는 특징이 있습니다.

**React**

거대한 커뮤니티

페이스북의 지원을 받는 만큼 지속적인 버전 관리가 이루어지고, React 사용자가 많아 다양한 레퍼런스, 확장 라이브러리가 많습니다. 또한 이미 React를 사용하고 있는 개발 프로젝트가 많기 때문에 개발자 채용 시, React를 다룰 줄 안다는 것이 장점으로 작용할 수 있습니다.

JSX 기반 컴포넌트 구문

템플릿 방식으로 관리하는 Vue와는 달리, React는 JSX 코드로 컴포넌트를 작성하고 컴포넌트의 상태(State)를 변화시키지 않고 관리합니다. 변화가 일어나면 실제 브라우저의 DOM에 새로운 것을 적용하는 것이 아니라, 자바스크립트로 이루어진 Virtual DOM에 렌더링을 하고 기존의 DOM과 비교하여 변화가 일어난 곳만 업데이트 합니다.

템플릿은 재구조화가 어렵고, 에러에 취약하지만, React는 상태를 고유 속성대로 유지하기 때문에 대규모 프로젝트에서도 관리가 수월하고 테스트에도 적합합니다.

출처: <http://library.gabia.com/contents/infrahosting/8284>

**3. STATE와 PROPS를 설명하시오.**

리액트 컴포넌트에서 다루는 데이터는 두개로 나뉘는데, 바로 props 와 state 이다.

Props

props 는 부모 컴포넌트가 자식 컴포넌트에게 주는 값이며, 자식 컴포넌트에서는 props 를 받아오기만하고, 받아온 props 를 직접 수정 할 수 는 없다.

state

state 는 컴포넌트 내부에서 선언하며 내부에서 값을 변경 할 수 있다.

출처: <https://velopert.com/3629>