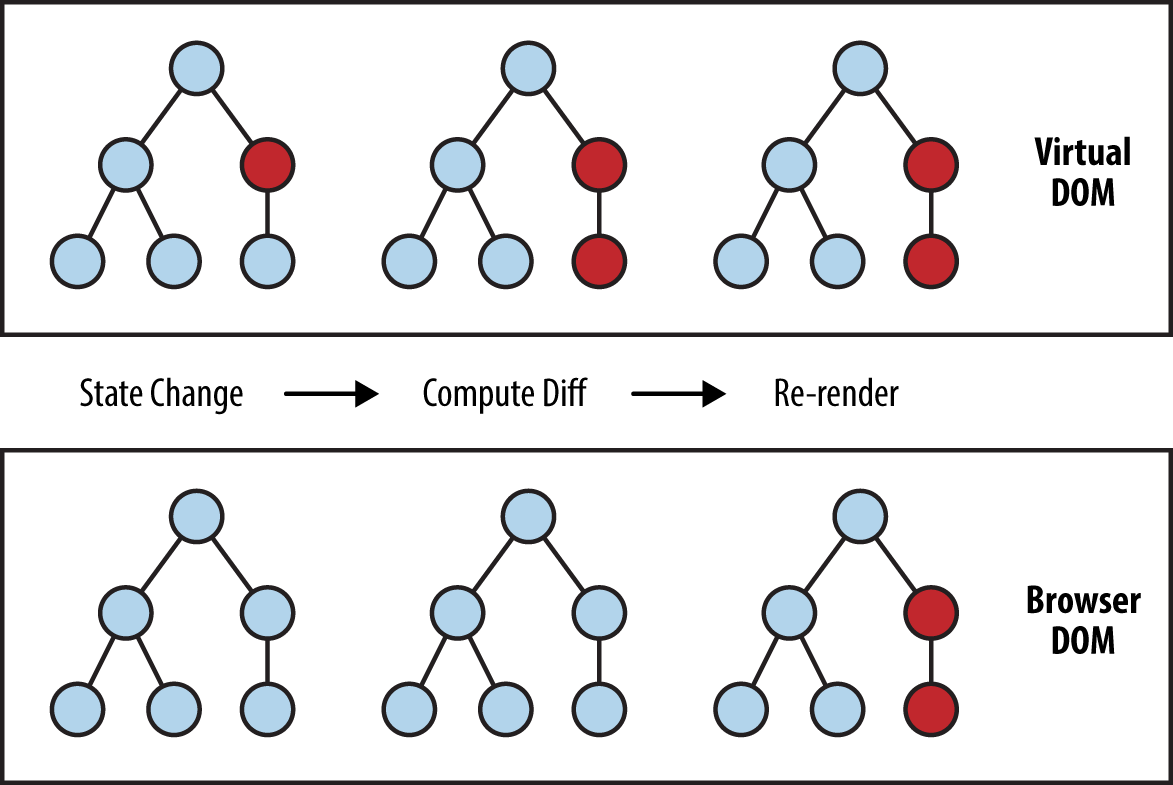
**리액트의 장점**

**1. 빠른 업데이트와 렌더링 속도**

업데이트란 웹사이트를 탐색할 때 화면에 나타나는 내용이 바뀌는 것을 의미합니다. 이러한 빠른 업데이트를 위해 리액트는 Virtual DOM(가상 돔)을 사용합니다.

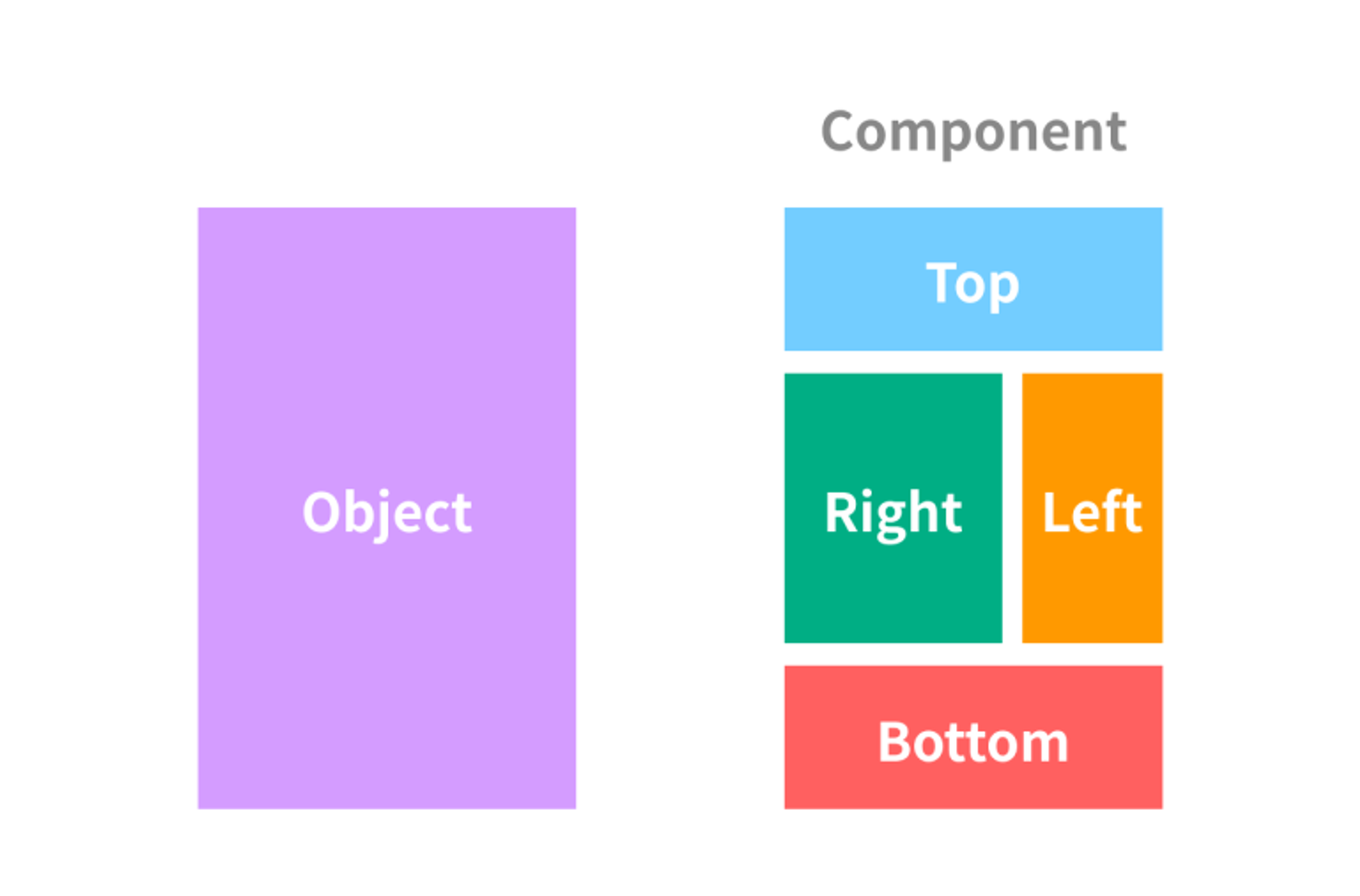
JS에서 사용하는 DOM과는 살짝 다른 개념입니다. 기존의 DOM은 화면이 업데이트 하려면 DOM을 직접 수정해야 하는데, 이는 전체를 다시 불러서 변경하므로 시간과 비용이 많이 소모됩니다.

리액트는 DOM을 수정하는 것이 아니라 상태의 변경(State Change)가 일어났을 때만 Virtual DOM에서 업데이트해야 될 최소한의 부분만을 검색합니다. 그렇게 검색한 최소한의 부분만 렌더링하면서 변경된 내용을 빠르게 보여줄 수 있습니다.



**2. 컴포넌트 기반 구조**

컴포넌트는 구성요소라는 뜻을 갖고 있습니다. 리액트에서는 모든 페이지가 컴포넌트로 구성되어 있고, 하나의 컴포넌트는 또 다른 여러 개의 컴포넌트의 조합으로 구성될 것입니다. 마치 작은 레고 블록들이 모여서 하나의 완성된 모형이 되는 것과 비슷합니다.



1. **재사용성**

**재사용성이란 말 그대로 다시 사용이 가능한 성질을 의미합니다.** 재활용과는 다른 개념입니다.

재사용의 장점

1. 전체 소프트웨어의 개발 기간이 단축됩니다. 비슷한 소프트웨어를 개발한다고 할 때, 기존에 개발해둔 모듈을 곧바로 재사용하여 개발하면 되기 때문에 개발 기간을 굉장히 많이 줄일 수 있습니다.
2. 유지 보수가 용이합니다. 여러 소프트웨어에서 공통으로 사용하는 모듈에 문제가 생긴다면 해당 모듈만 수정해서 다시 배포하면 되기 때문입니다. 또한 소프트웨어를 개발하다 보면 예상치 못한 버그를 마주치는 상황이 빈번하게 발생합니다.

이런 경우 재사용성이 높은 형태로 개발되어 있다면 좀 더 버그의 원인을 찾기 쉽습니다. 재사용성이 높다는 것은 결국 여러 모듈 간의 의존성이 낮다는 뜻이기 때문입니다. 이는 각 부분들이 잘 분리되어 있고 쉽게 버그를 찾아서 수정할 수 있습니다.

1. **든든한 지원군**

메타(구 페이스북)라는 든든한 지원군을 들 수 있습니다. 프로젝트가 성장하고 꾸준히 유지되기 위해서는 수많은 개발자의 노력이 필요합니다. 영향력있는 라이브러리들은 꾸준히 업데이트가 되고 영향력이 지속될 겁니다. 이는 최대 IT 기업 중 하나인 메타에서 많은 개발자들이 꾸준히 업데이트를 하고 몇 년 동안은 그 영향력이 지속될 겁니다.

1. **활발한 지식 공유 & 커뮤니티**

오픈 소스 플랫폼인 GitHub에서 가장 많은 관심을 가지고있는 커뮤니티는 react이다. 특정 기술의 생태계 규모를 판단하는 지표로 스택오버플로(StackOverflow) 웹사이트의 해당 기준을 참고 했을 때 리액트는 많은 관심을 받고 있습니다. 스택오버플로는 개발자 간의 개발과 관련된 질문과 답변을 공유하는 커뮤니티이다.

1. **모바일 앱 개발 가능**

리액트의 또 하나의 장점으로 리액트를 배운 이후에 리액트 네이티브(React Native)라는 모바일 환경 UI 프레임를 사용하여 모바일 앱도 개발할 수 있다는 것 입니다.

모바일 앱을 개발하기 위해서 안드로이드 앱의 경우 코틀린이라는 프로그래밍 언어를 배워야하고, IOS 앱은 스위프트 라는 별도의 프로그래밍 언어를 배워야 합니다. 프로그래밍 언어 뿐 아니라 각각의 개발 프레임워크에 대해서도 학습을 해야 합니다.

리액트 네이티브를 사용하면 자바스크립트 코딩을 통해 안드로이드 앱과 IOS 앱을 동시에 출시할 수 있습니다. 네이티브 앱보다 성능이나 속도 면에서 부족한 부분을 보일 수 있지만 차이점을 느끼지 못할 정도로 개발할 수 있습니다.

**리액트의 단점**

1. **방대한 학습량**

리액트의 경우, 다른 방식의 UI 라이브러리이기 때문에 배워야 할 것들이 많습니다. 리액트의 경우 지속적인 버전 업데이트가 이뤄지고 있어 새로운 내용들이 꾸준히 등장해 업데이트가 된다면 해당 내용을 학습하고 숙지하고 있어야 실무 환경에서 원할하게 작업할 수 있습니다.

1. **높은 상태 관리 복잡도**

리액트의 또 다른의 단점으로 높은 상태 관리 복잡도를 들 수 있습니다. 리액트에는 **state**라는 굉장히 중요한 개념이 있습니다. **state**는 쉽게 말해 리액트 컴포넌트 상태를 의미합니다. Virtual DOM을 언급할 때 바뀐 정보는 바뀐 **state**를 의미합니다. 따라서 **state**는 리액트에서 굉장히 중요한 역할을 담당하고 있습니다. 성능 최적화를 위해 **state**를 잘 관리하는 것이 중요한데 프로젝트의 규모가 점차 커질경우 컴포넌트의 갯수가 많아지면서 상태 관리의 복잡도가 증가합니다. 이는 프로젝트를 관리하는데에 있어서 어려움을 줍니다.

그래서 보통 규모가 큰 프로젝트에서는 상태 관리를 위해서 Redux, MobX, Recoil 등의 외부 상태 관리 라이브러리를 사용하는 경우가 많습니다. 이를 관리하기 위해 다른 라이브러리를 공부하는 것도 방대한 학습량에 포함이 됩니다.

1. **라이브러리의 의존성**

리액트의 경우 메타에서 개발한 JavaScript 기반의 라이브러리입니다. 하지만 라이브러리는 언제든지 개발, 지원이 멈출 수 있습니다. 지금은 메타가 잘나가는 대기업이라고 하지만 언제 어떻게 될지는 모르는 겁니다. 이에 리액트 문법만 공부하는게 아니라 언제든지 자바스크립트 문법에 환경에 적응 할 수 있게 자바스크립트 공부도 필요로 합니다.