React가 VirtualDOM을 활용하는 방법

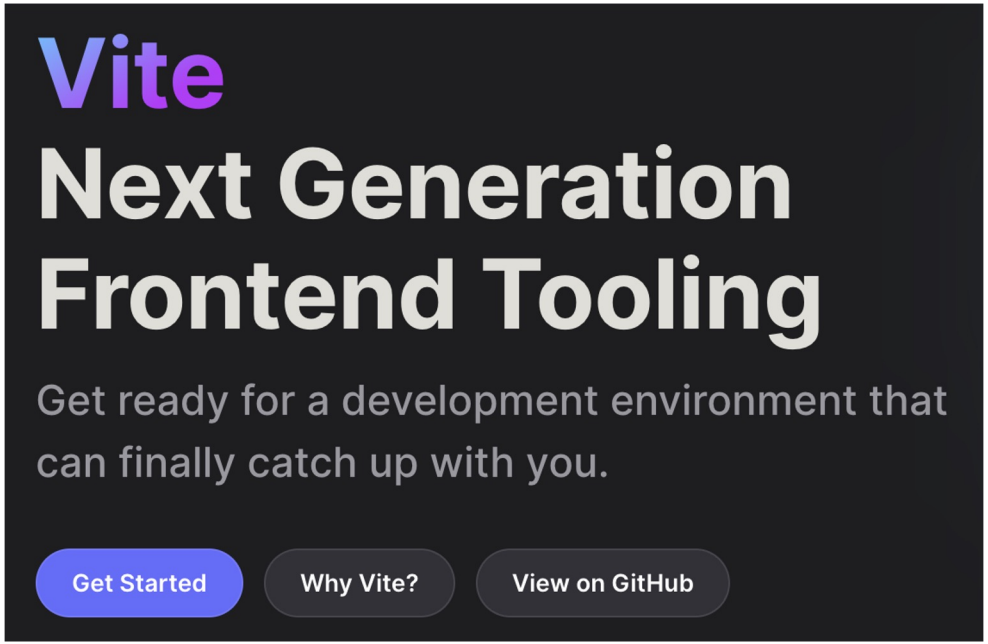
[OT]

1. 강사소개
2. 강의 주제 선정 이유
3. 커리큘럼 오버뷰

[OT]

1. React가 VirtualDOM을 활용하는 방법
    a. VirtualDOM의 역할
    b. Reconciliation 이란?
2. 상태(state)관리가 rendering에 미치는 영향
    a. 불필요한 rendering을 방지하려면?
    b. 다양한 hook들의 차이점과 효율적인 사용법
3. 현업에서 React를 최적화하는 방법
    a. Performance 최적화
    b. 신입들은 최적화를 어디서 경험하나요…
4. 프론트엔드 기본개념
    a. 면접대비
    b. 요즘 우대사항에서 많이 언급되는 용어들

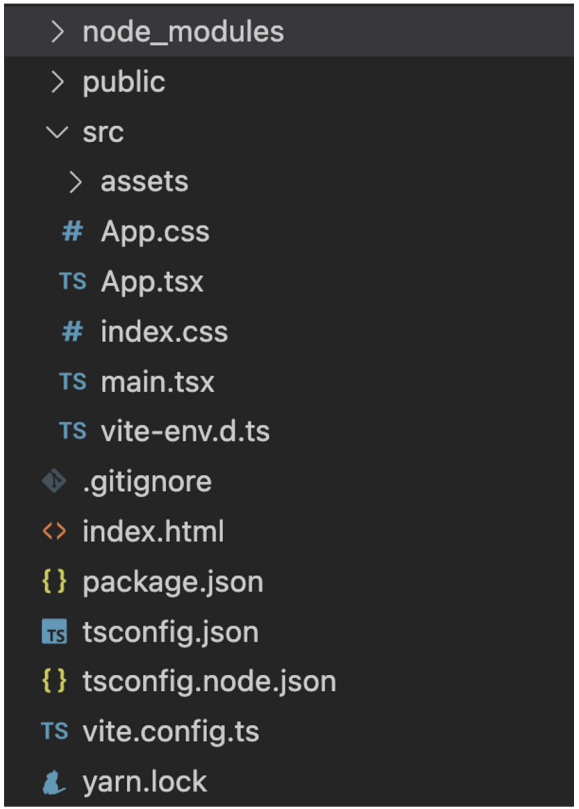
vite로 init한 React 프로젝트



vite로 init한 React 프로젝트

1. 번들링이 매우 빨라짐
    a. https://vitejs.dev/guide/dep-pre-bundling.html
2. Webpack, Rollup, Parcel등을 ESBuild라는 것으로 대체

vite로 init한 React 프로젝트



`main.tsx`부터 따라가보자

src > TS main.tsx

```tsx
import {StrictMode} from 'react'
import {createRoot} from 'react-dom/client'
import App from './App'
import './index.css'

createRoot(document.getElementById('root') as HTMLElement).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

createRoot()

```
createRoot(document.getElementById('root') as HTMLElement).render(
```

```
(alias) createRoot(container: Element | DocumentFragment, options?: RootOptions
| undefined): Root
import createRoot
```

Replaces `ReactDOM.render` when the `.render` method is called and enables Concurrent Mode.

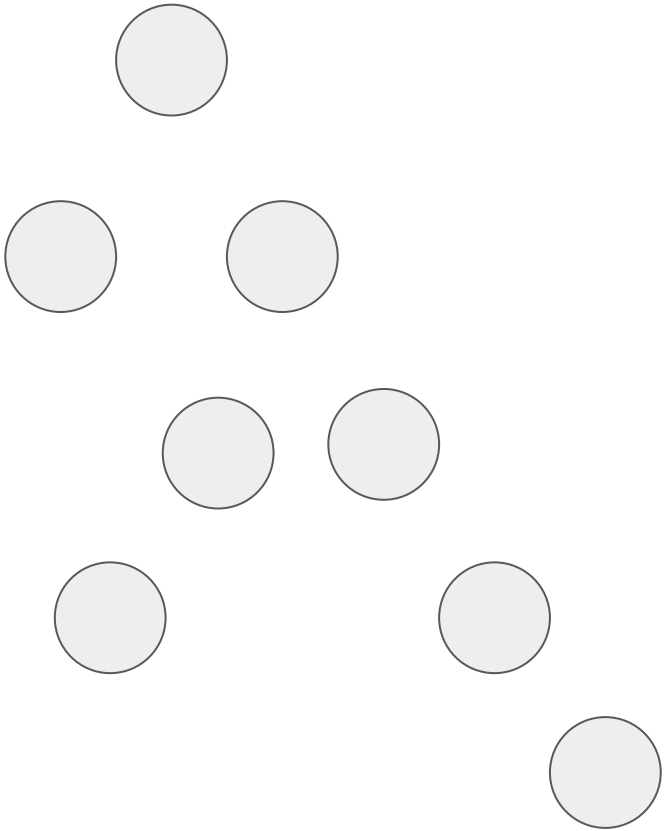
@see — https://reactjs.org/docs/concurrent-mode-reference.html#createroot

1. Element | DocumentFragment
2. Concurrent Mode
3. RootOptions

createRoot()

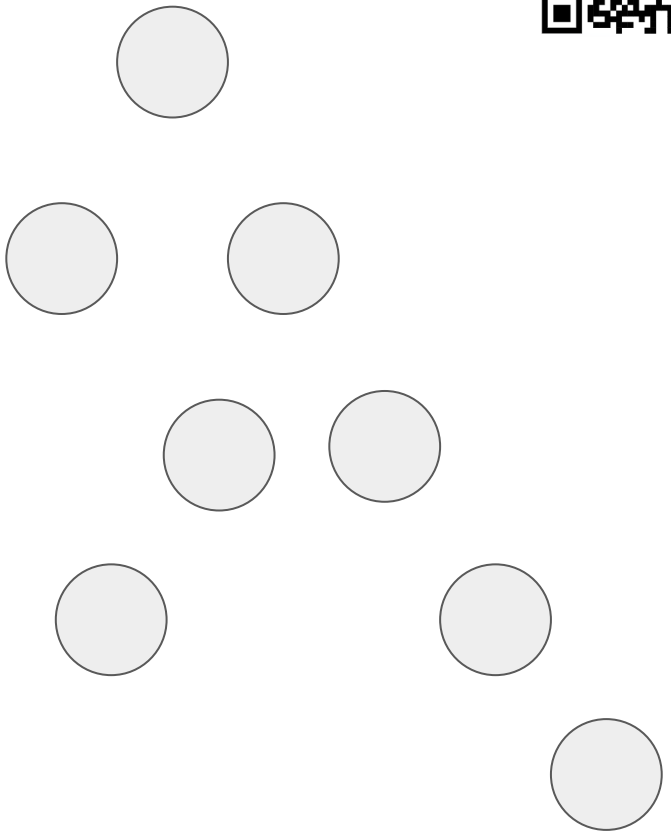
1. DOM Tree vs VirtualDOM Tree
2. VirtualDOM Tree에서 `<div id=”root” />`를 나타내는 node를 생성함

DOM Tree



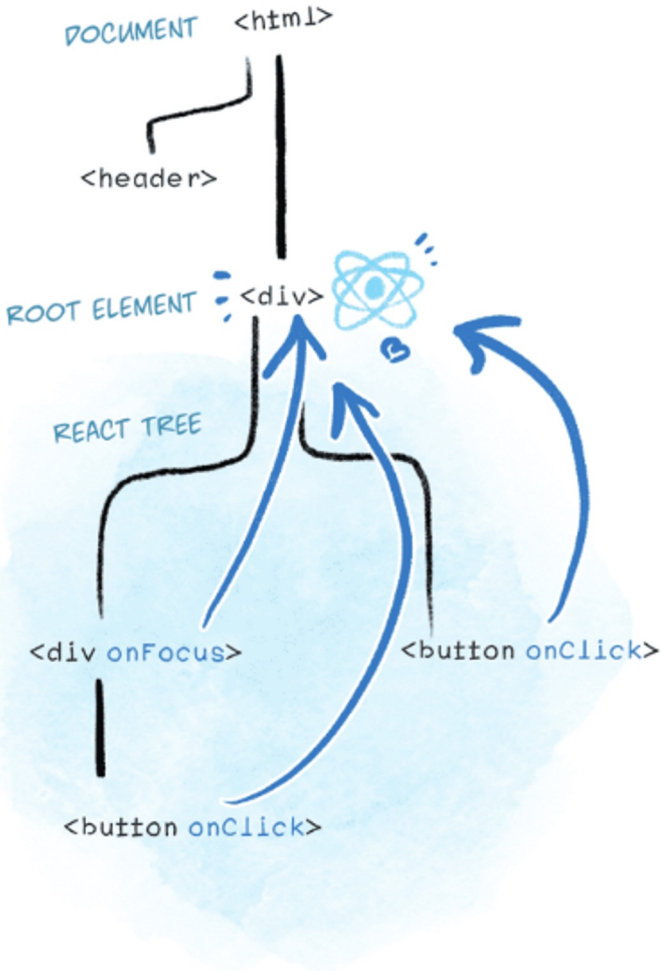
Virtual DOM Tree

HostRoot

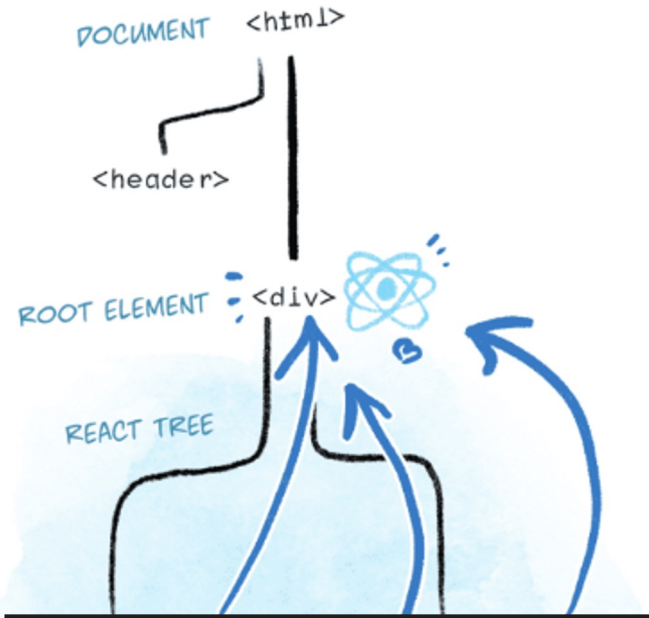


VirtualDOM의 역할

1. 최적화된 업데이트로 사용자 경험 개선
    a. 브라우저에 반영하기 전에 사전작업을 하는 것
2. 업데이트에 우선순위를 부여
    a. Animation vs Text


DOCUMENT
<html>
<header>
ROOT ELEMENT

REACT TREE
<div onFocus>
<button onClick>
<button onClick>




```
markContainerAsRoot(root.current, container);
var rootContainerElement = container.nodeType ===
listenToAllSupportedEvents(rootContainerElement);
return new ReactDOMRoot(root);
```

Fiber

1. 리액트 렌더링/업데이트의 가장 작은 단위
2. `work`라고도 한다
3. 효율적인 업데이트를 위해
    a. work를 중지하고, 필요 시 다시 시작할 수 있어야 한다.
    b. 다른 종류의 work들에게 우선순위를 부여할 수 있어야 한다.
    c. 이미 완료된 work를 재사용 할 수 있어야 한다.
    d. work가 더이상 필요 없게 되면 버릴 수 있어야 한다.

render()

```
(method) Root.render(children: React.ReactNode): void
etElementById('root') as HTMLElement).render(
```

render()

1. Render phase
    a. 업데이트를 수행해서 화면을 변경하는 것
2. Commit phase
    a. 변경된 화면을 실제 브라우저에 나타내는 것

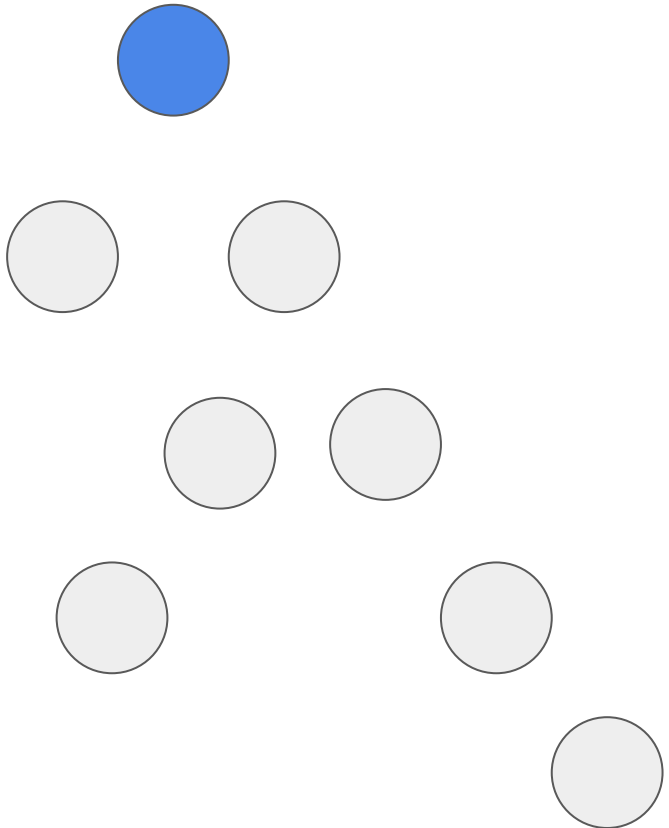
일상 용어로 설명하면?

1. **Triggering** a render (delivering the guest's order to the kitchen)
2. **Rendering** the component (preparing the order in the kitchen)
3. **Committing** to the DOM (placing the order on the table)

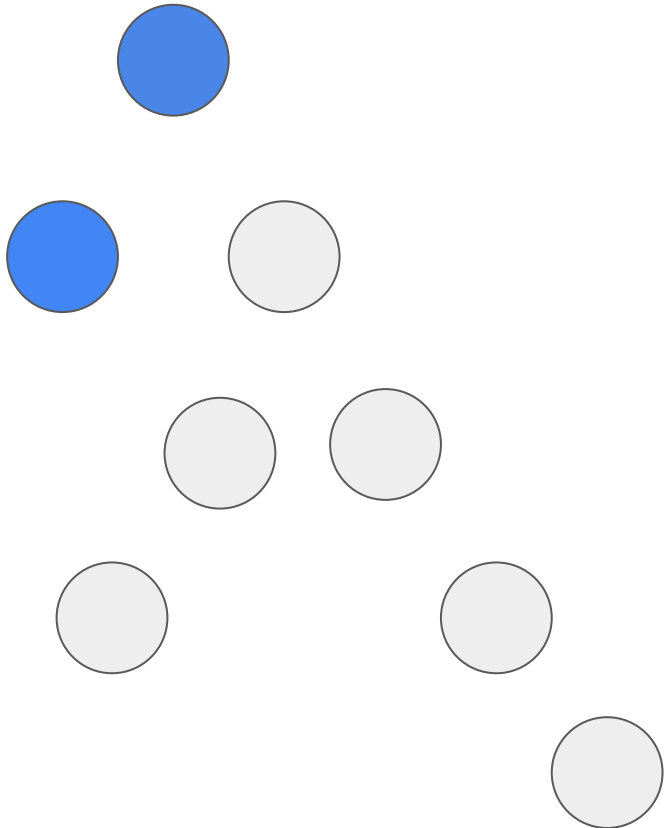
Reconciliation

1. 업데이트가 발생할 때 기존의 tree와 차이점을 비교하는 방법
2. DFS(Depth First Search)로 tree를 Traverse

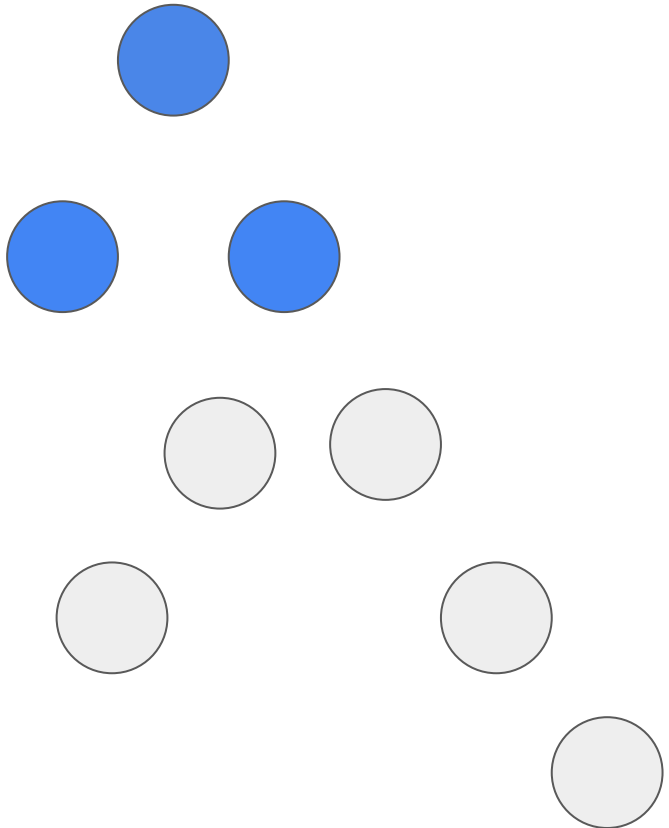
Render Phase



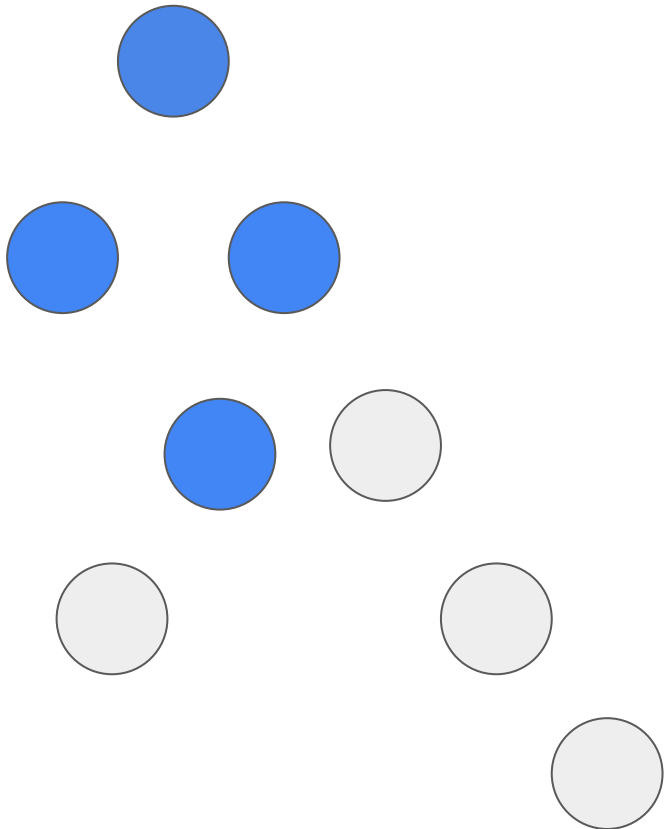
Render Phase



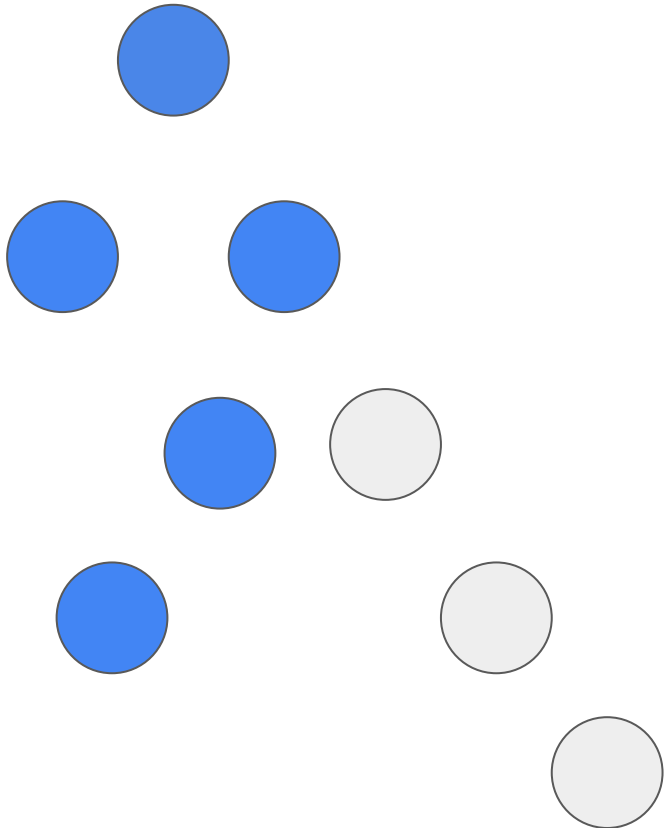
Render Phase



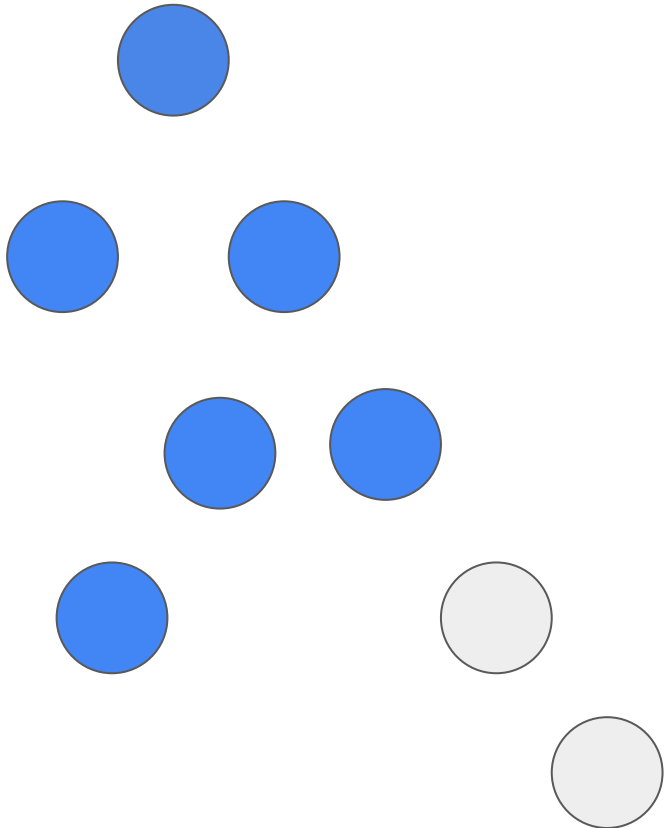
Render Phase



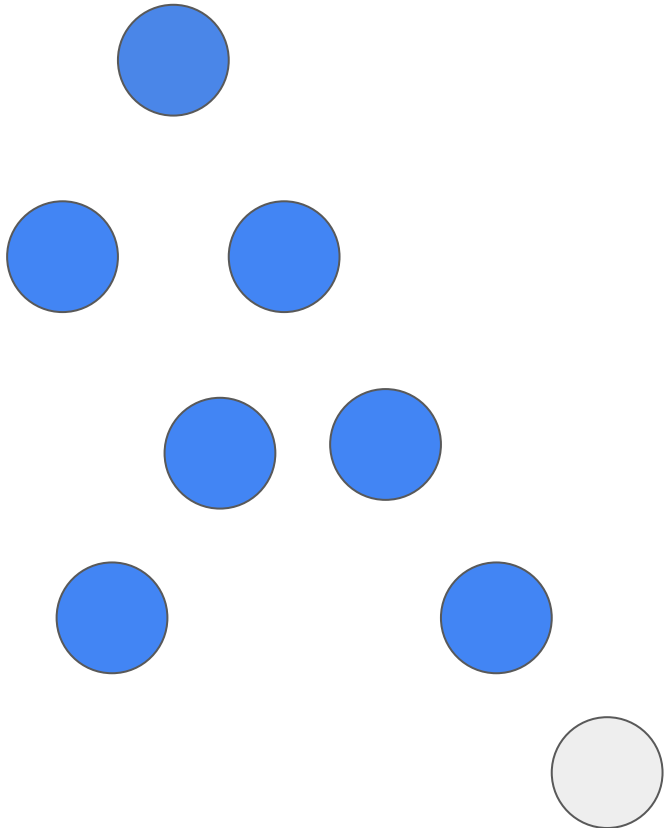
Render Phase



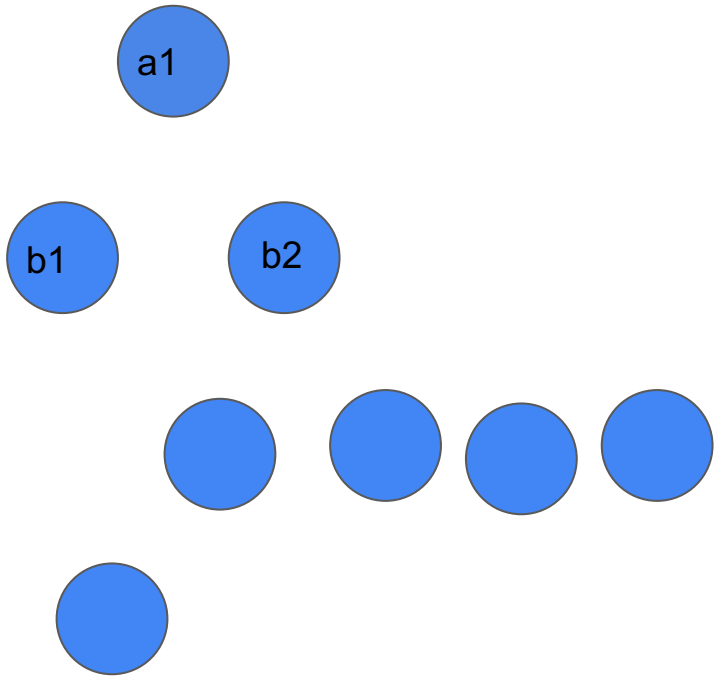
Render Phase



Render Phase

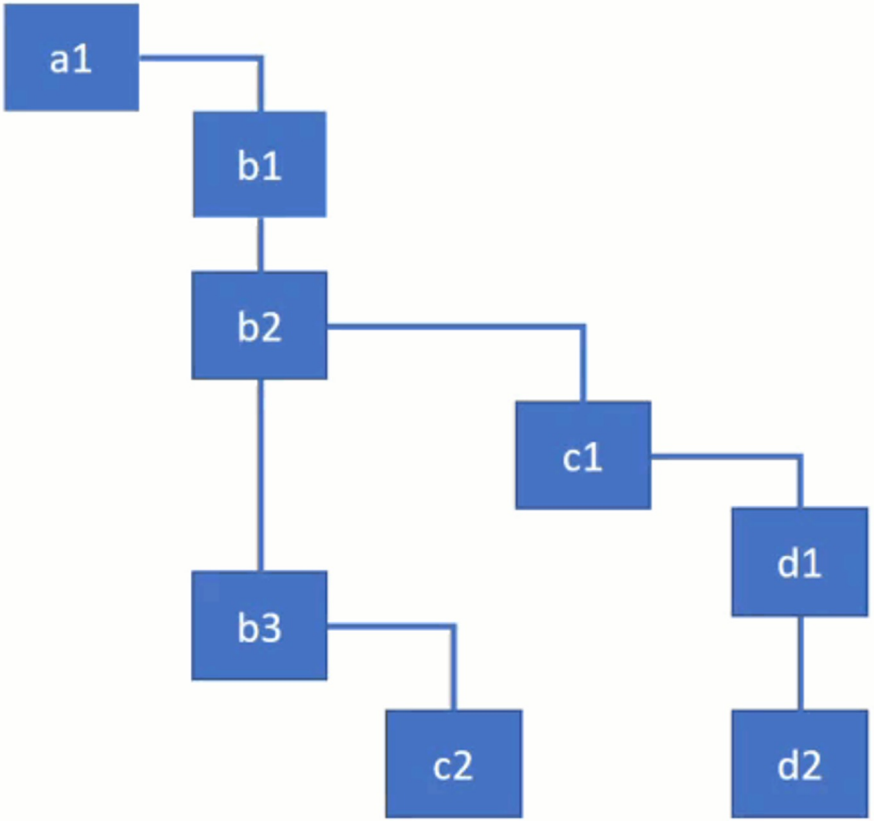


Render Phase



Render Phase

1. `performUnitOfWork`
2. `beginWork`
3. `completeUnitOfWork`
4. `completeWork`



nextUnitOfWork

performUnitOfWork

beginWork

completeUnitOfWork

completeWork

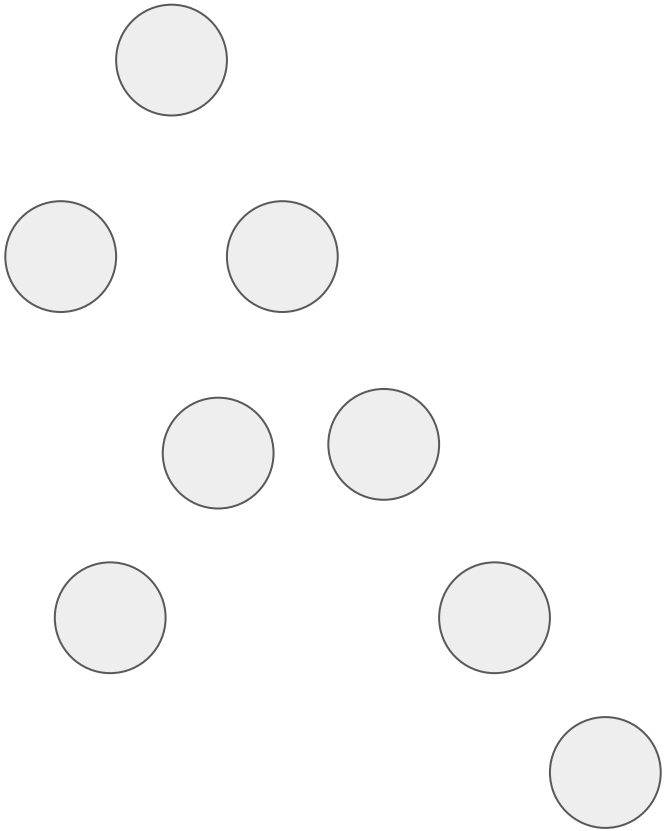
Render Phase

1. console.log()로 확인

Current and work-in-progress

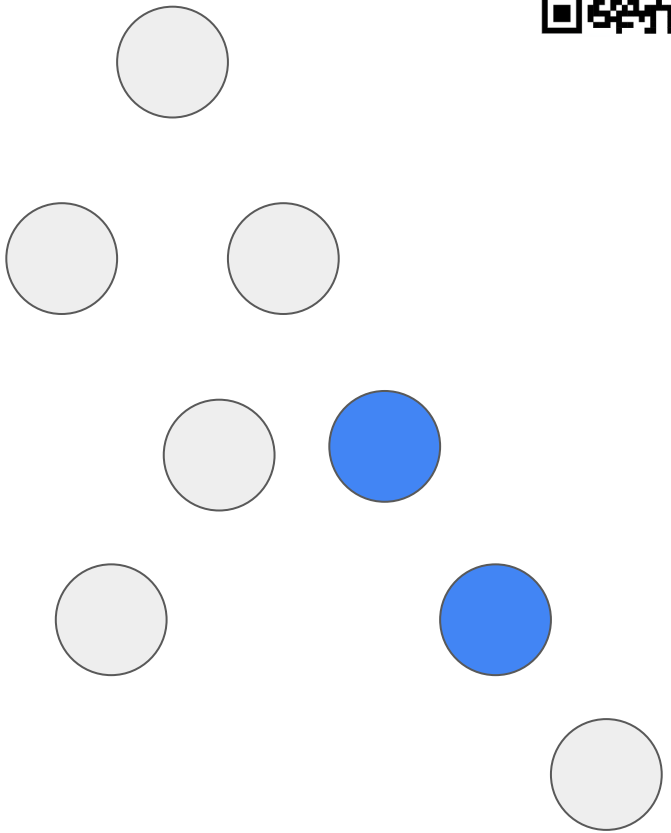
1. Current : 현재 브라우저에 보이는 DOM Tree
2. Work-in-progress: 업데이트 사항이 반영되는 중인 DOM Tree

DOM Tree

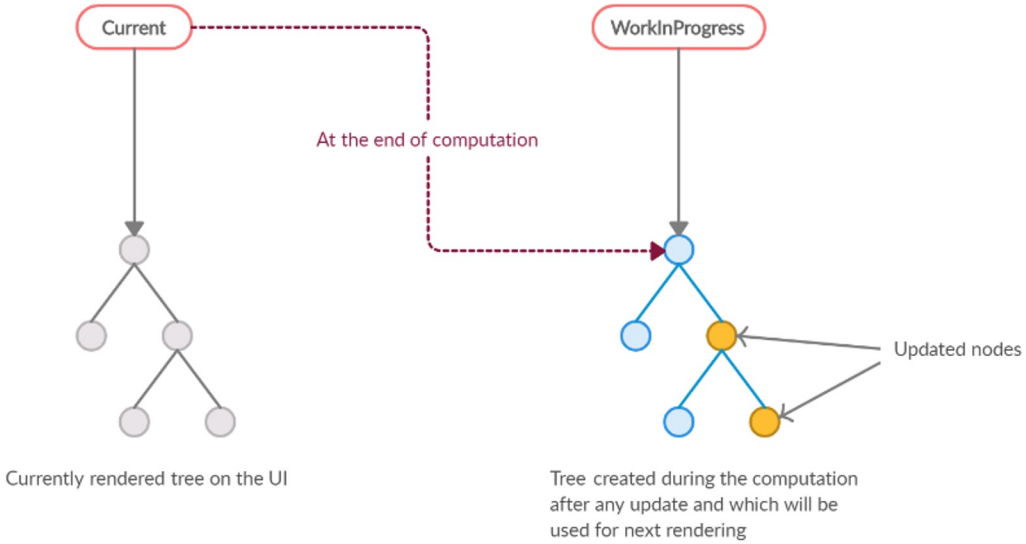


Virtual DOM Tree

HostRoot



Commit Phase



Render 효율을 위해

1. 불필요한 jsx element는 없는 것이 좋음
2. key를 잘 지정해주어야 한다

key

Disallow usage of Array index in keys (`react/no-array-index-key`)

Warn if an element uses an Array index in its `key`.

key

Rules of keys

- **Keys must be unique among siblings.** However, it’s okay to use the same keys for JSX nodes in *different* arrays.
- **Keys must not change** or that defeats their purpose! Don’t generate them while rendering.

[아하!모먼트] 면접에 대비하는 주니어 프론트엔드 개발자를 위해

1. 면접대비 방법
    a. 스터디
    b. 모의면접

[아하!모먼트] 면접에 대비하는 주니어 프론트엔드 개발자를 위해

1. 면접대비 방법
    a. 스터디
    b. 모의면접
2. CS 지식
    a. 알고리즘 - 시간복잡도
    b. 자료구조
    c. 그냥많이

[아하!모먼트] 면접에 대비하는 주니어 프론트엔드 개발자를 위해

1. 면접대비 방법
    a. 스터디
    b. 모의면접
2. CS 지식
    a. 알고리즘 - 시간복잡도
    b. 자료구조
    c. 그냥많이
3. 이력서, 포트폴리오

[아하!모먼트] 면접에 대비하는 주니어 프론트엔드 개발자를 위해

1. 면접대비 방법
    a. 스터디
    b. 모의면접
2. CS 지식
    a. 알고리즘 - 시간복잡도
    b. 자료구조
    c. 그냥많이
3. 이력서, 포트폴리오

[아하!모먼트] 면접에 대비하는 주니어 프론트엔드 개발자를 위해

1. 면접대비 방법
    a. 스터디
    b. 모의면접
2. CS 지식
    a. 알고리즘 - 시간복잡도
    b. 자료구조
    c. 그냥많이
3. 이력서, 포트폴리오
4. 마인드컨트롤

참고링크

https://blog.ull.im/engineering/2019/03/10/logs-on-git.html

https://tailwindcss.com/docs/installation