using React Dev Tools

Debugging React

2024. 3. 24.

두선아

Agenda

01. What is React Dev Tools?
02. Component
03. Profiler
04. Conclusions

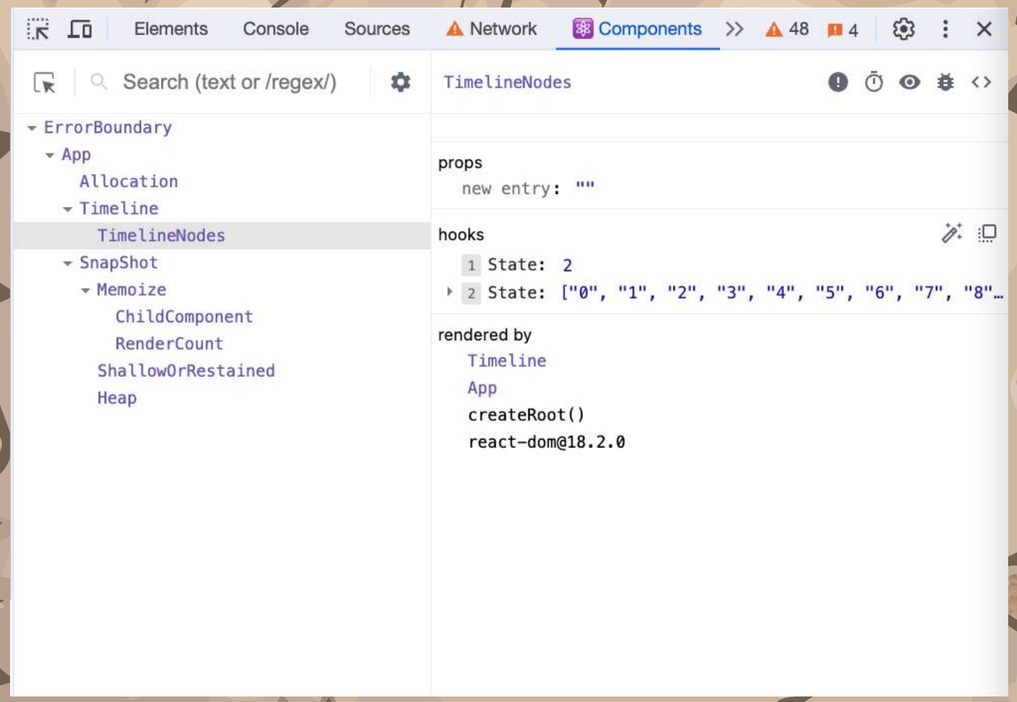
01 React Dev Tools이란?

리액트 앱의 개발과 디버깅을 도와주는
브라우저 확장 프로그램
Chrome, Firefox, Edge

리액트 애플리케이션의 전체 컴포넌트 트리를 탐색하고,
상태와 속성을 검사하며, 성능 문제를 진단

리액트 앱의 모든 측면을 이해하는 데 필수적

02 Component 컴포넌트 탭



탐색

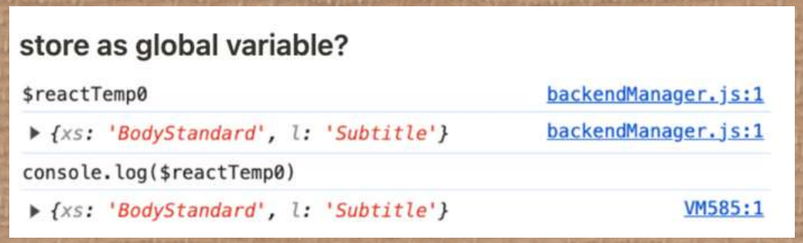
앱의 컴포넌트 트리를 확인

- 앱의 구조를 이해
- 특정 컴포넌트의 상태와 데이터 흐름을 디버깅

검사

선택한 컴포넌트의 상세 정보

- props, state, hooks (현재 값과 변경 내역을 검사)
  - copy value to clipboard
  - store as global variable



학습 사항

❓익명함수를 default로 export한 경우?

_defalut

- ErrorBoundary
  - App
    - _default
    - StateManagement
    - HOC
    - HooksC

❓(Anonymous) Memo, ForwardRef 컴포넌트

- HooksA
  - Anonymous `Memo`
    - RenderCount

- HooksB
  - Anonymous `ForwardRef`
    - RenderCount

❓(Anonymous) HOC, 고차 컴포넌트로 감싼 컴포넌트

- HOC
  - Anonymous

❓무기명 함수에 displayName 속성을 추가

- HooksA
  - hi?💁
    - RenderCount

```
ChildComponent.displayName = "hi?💁";
```

- 리액트를 빌드하는 경우, 난수화되어 사라질 것
- 개발 모드에서 제한적으로 확인

❓컴포넌트 상세 정보

- 변수
  - copy value to clipboard
  - store as global variable: window.$r // 전역변수로 보관
- 함수
  - go to definition
- hooks
  - use가 생략된 모습임
  - 커스텀 훅도 나옴
  - 훅에 넘겨주는 함수 기명함수면 확인o
- rendered by
  - 랜더링 주체

학습 사항

📌 컴포넌트 도구

inspect the matching DOM element
- Element 탭으로 이동 + 랜더링된 HTML 요소 선택

log this component data to the console
- Props, Hooks, Nodes

view source for this element
- 소스 코드 보기

📌 console.log - Props, Hooks, Nodes

console	내용
Props	card, onRenderComplete, setFirstCardImage
Hooks	hookSource, id, isStateEditable, name, subHooks, value
Nodes	노드 element 배열

```
▼<div> == $0
    "non useCallback"
    ":"
  </div>
```

```
▼ [Click to expand] <Memo(Anonymous) />                                                    backendManager.js:1
  Props: ▶ {name: 'non useCallback', value: false, onChange: f}                            backendManager.js:1
  Hooks: ▶ (2) [{…}, {…}]                                                                  backendManager.js:1
  Nodes: ▶ (3) [div, div.flex.gap-2.items-center.justify-center, p.text-xs.text-right.opacity-30]  backendManager.js:1
  Right-click any value to save it as a global variable for further inspection.            backendManager.js:1
```

```tsx
const ChildComponent = memo(({ name, value, onChange }: MyProps) => {
  const renderCount = MyUseRef<number>(0);

  useEffect(() => {
    console.log("rendering", name);
    renderCount.current += 1;
  });

  return (
    <>
      <div>{name}:</div>
      <div className="flex gap-2 items-center justify-center">
        <button onClick={onChange}>{value ? "on" : "off"}</button>
      </div>
      <RenderCount count={renderCount.current} />
    </>
  );
});
```

Component 탭 사용 예

📌 사용자 인터페이스의 특정 부분이 예상대로 동작하지 않을 때,
- 'Components' 탭을 사용하여 해당 UI 부분을 구성하는 컴포넌트를 찾기
- 컴포넌트를 선택해서, props와 state를 실시간으로 검사
    - props나 state가 예상치 못한 방식으로 변경되는 경우를 추적
    - 데이터 흐름이나 상태 변화의 문제를 진단

📌 리랜더링 원인 찾기
- 불필요한 리렌더링을 줄이는 데 도움

Component

단순히 구조와 정보를 확인하는 것을 넘어
애플리케이션의 데이터 흐름과
컴포넌트 간의 상호작용을 이해하고
더 신속하고 효과적으로
디버깅할 수 있다.

문제 컴포넌트를 빠르게 식별, 원인 파악

03

Profiler 프로파일러 탭

성능 진단

- 불필요한 렌더링과 성능 병목 현상을 정밀하게 분석
- 랜더링 과정에 개입해 디버깅에 필요한 내용을 기록(개발 phase)

Flamegraph

랜더 커밋별
일어난 작업 확인

너비는 랜더링 시간

랜더링 관련 정보 확인

Ranked

랜더 커밋별
시간 오래걸린 순으로
정렬한 그래프

랜더링 발생한
컴포넌트만 보여줌

Timeline

시간의 흐름에 따라
컴포넌트 작동 확인

timestamp
동기/비동기
업데이트 발생 시간

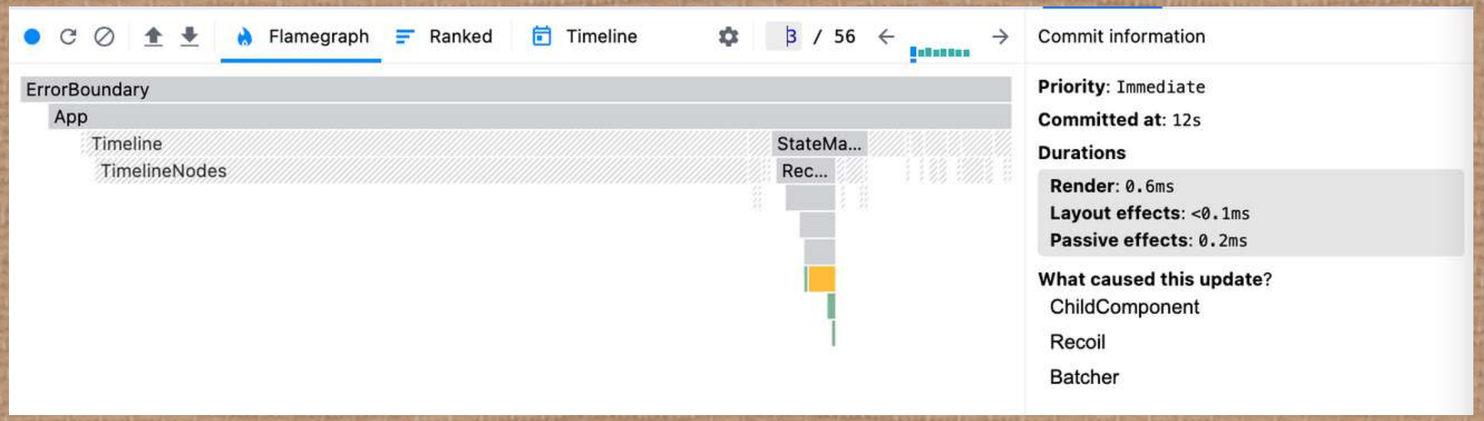
실습

CodeSandbox 샌드박스 링크

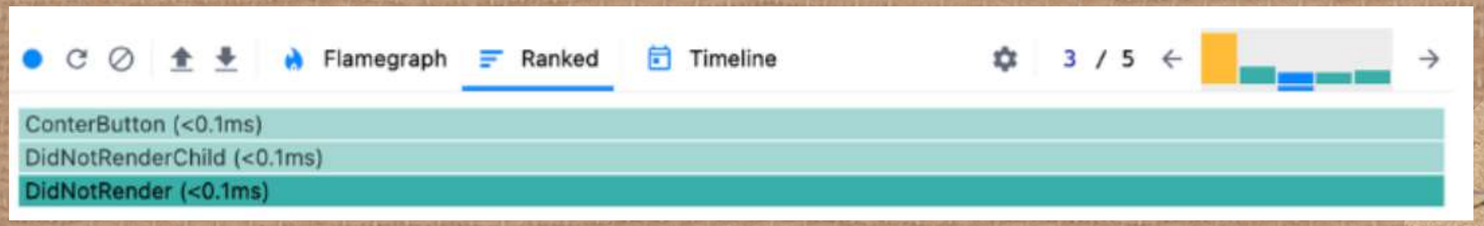
6장 예제 코드 실습

React Dev Tools 확인

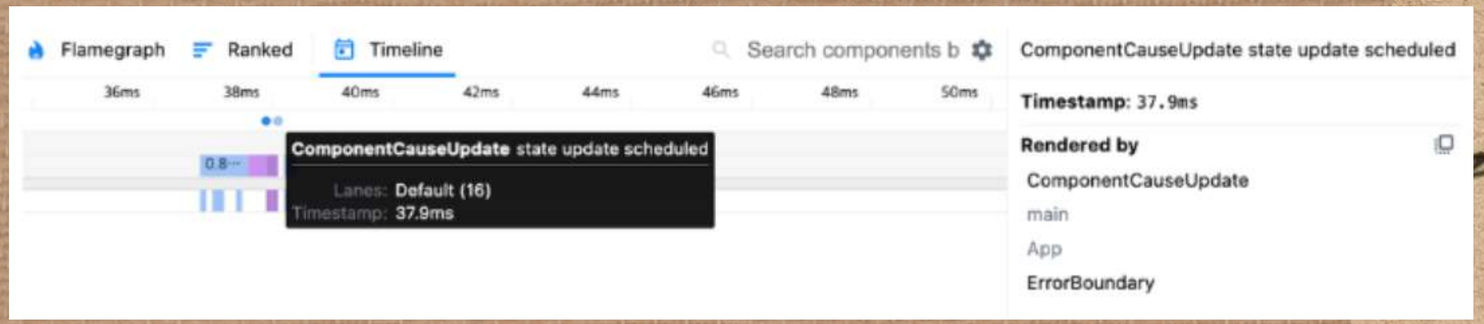
👉 Flamegraph



👉 Ranked



👉 Timeline



학습 사항

? 설정 변경하기

- Highlight updates when components render.
- Hide logs during second render in Strict Mode.
  - Strict Mode의 useEffect 두 번 찍히는 사항 비활성화
- Record why each component rendered while profiling
  - profiling 중, 컴포넌트 랜더링 원인을 기록

? Start / Reload / Stop Profiling

Start Profiling

- 시작/종료 토글

Reload and Start Profiling

- 새로고침 후 프로파일링 시작

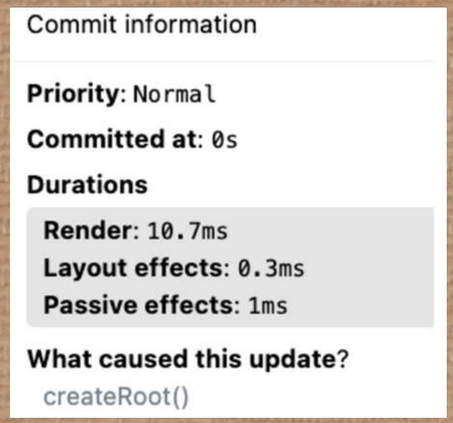
Stop Profiling

- 프로파일링된 현재 내용을 모두 삭제

? Load Profile... / Save Profile...

- profiling 기록을 json 형태로 불러오기/저장

? Commit information



Profiler 탭 사용 예

📌 불필요하게 자주 리렌더링되는 경우
- 리렌더링되는 컴포넌트를 식별
- 예를 들어 목록 컴포넌트가 리랜더링 되는 경우,
  컴포넌트를 메모하고 props가 실제로 변경될 때만 리렌더링되도록 한다
  - 페이지의 로딩 시간 단축

📌 '커밋'된 시간과 각 컴포넌트의 '렌더링 시간'에 주목
- 어떤 컴포넌트가 성능 저하의 주범인지 식별한다.
- 최적화의 우선 순위를 정한다.
  - (Ranked) 랜더링 가장 오래 걸리는 컴포넌트 확인

Profiler

애플리케이션의 렌더링 성능에 대한
깊이 있는 분석

컴포넌트의 렌더링 시간 측정

리렌더링 원인 파악

최적화 우선 순위 선정를 위한 데이터

04

Conclusions

01

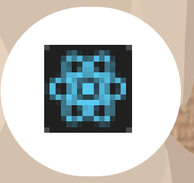
코드가 많아져서 손 대기 힘들어지기 전에
디버깅해서 버그를 방지하자.

- Dev Tools은 코드 이해/작성에도 도움이 된다.

02

디버깅 툴을 잘 활용하자.

- React Dev Tools, Chrome Dev Tools
- 웬만한 라이브러리는 Dev Tool이 있다.



Thank you

Reference & Links

- 모던 리액트 Deep Dive
  - 6장 리액트 개발 도구로 디버깅하기 (401~430p)
- CodeSandbox: mordern-react-deep-dive
  - 6장 예제 코드 실습
  - React Dev Tools 크롬 익스텐션 확인