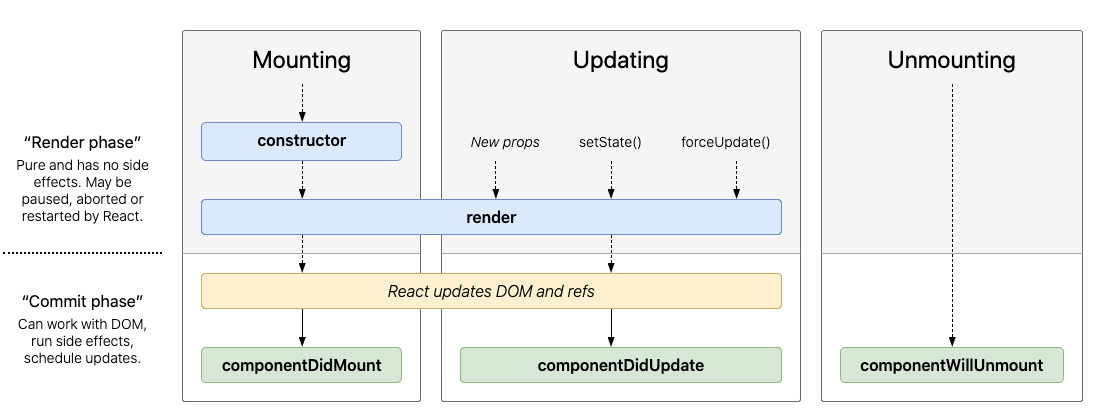
※컴포넌트의 라이프사이클 이해(주요 메서드 호출 순서)

1. 컴포넌트 라이프사이클이란?

컴포넌트 라이프사이클은 컴포넌트의 생성부터 소멸에 이르는 일련의 이벤트로 생각할 수 있습니다.

모든 리액트 컴포넌트는 Lifecycle을 갖습니다. Lifecycle은 세 가지 카테고리로 나누어집니다.

1. Mounting: 컴포넌트가 화면에 나타남
2. Updating: 컴포넌트가 업데이트됨
3. Unmounting: 컴포넌트가 화면에서 사라짐



[참고 : https://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/]

[중요 : index.js 파일 소스 수정 코딩]

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(<App />);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

2. 컴포넌트는 Lifecyle마다 메서드를 가지고 있습니다. 이 메서드를 이용해서 특정 시점에 원하는 동작을 하도록

만들 수 있습니다.

\* Lifecycle 메서드 : 에러 처리 메서드를 제외하면 Lifecycle 메서드는 총 8가지입니다.

1) Mounting : 컴포넌트의 인스턴스가 생성되어 DOM 상에 삽입되는 단계입니다. Mounting은 Lifecyle이

종료될 때까지 한 번만 일어납니다. 아래 메서드들이 이 단계에서 순서대로 호출됩니다.

· constructor: 컴포넌트의 인스턴스를 새로 만들 때마다 생성자 메서드가 호출됩니다.

· static getDerivedStateFromProps

· render: 화면에 표현될 JSX를 반환하고 화면에 그립니다.

· componentDidMount: 컴포넌트가 화면에 모두 그려진 이후 호출됩니다.

componentDidMount 메서드는 첫 렌더링 이후 실행됩니다. End-Point에서 클라이언트로 데이터를 불러와야 하는 경우라면, API 호출을 하기 좋은 위치입니다. 데이터를 받아 올 때 setState 메서드를 이용하여 컴포넌트를 업데이트할 수 있습니다.

2) Updating : props 또는 state가 변경되어 컴포넌트가 업데이트되는 단계입니다. 아래 메서드들이

이 단계에서 순서대로 호출됩니다.

· static getDerivedStateFromProps

· shouldComponentUpdate

· render: 데이터가 변경되면 자동으로 호출됩니다. 화면을 다시 그립니다.

· getSnapshotBeforeUpdate

· componentDidUpdate: 화면이 다시 그려진 이후 호출됩니다.

3) Unmounting : 컴포넌트가 DOM 상에서 제거되는 단계입니다.

·componentWillUnmount: 컴포넌트가 화면에서 제거되기 전에 호출됩니다.

=========================================================================

\* Lifecycle 메서드 실행 과정 : 라이프사이클 메서드는 리액트에서 지정된 시점에 실행됩니다. 각각의 Lifecyle

단계에서 메서드들이 실행되는 순서를 살펴보겠습니다. 부모컴포넌트와 자식컴포넌트를 하나씩 만들어서

실행했습니다.

1. Mouting : App.jsx(또는 App.js)

import React, { Component } from 'react';

import Child from './components/Child';

class App extends Component {

constructor(props) {

super(props);

this.state = {};

console.log('부모컴포넌트 => constructor 호출');

}

render() {

console.log('부모컴포넌트 => render 호출');

return <Child />;

}

}

export default App;

1) Mouting : Child.jsx

import { Component } from 'react';

class Child extends Component {

constructor(props) {

super(props);

this.state = {};

console.log('자식컴포넌트 => constructor 호출');

}

render() {

console.log('자식컴포넌트 => render 호출');

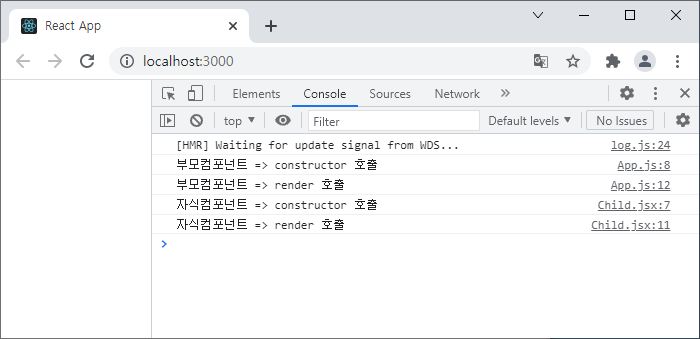
return null;

}

}

export default Child;

1) Mounting 종합 정리 : 실행 결과 확인 바랍니다.



Mounting 단계는 부모컴포넌트의 render가 호출된 다음 자식 컴포넌트의 Mounting 단계가 실행되는 것을

알 수 있습니다. 만약 부모컴포넌트와 자식컴포넌트 모두 componentDidMount 메서드를 호출한다면

자식컴포넌트에서 먼저 호출됩니다. 이는 아래 업데이트 단계를 살펴보면서 실행시켜 보겠습니다.

==========================================================================

1. Updating : App.jsx (또는 App.js)

import React, { Component } from 'react';

import Child from './components/Child';

class App extends Component {

constructor(props) {

super(props);

this.state = {};

console.log('부모컴포넌트 => constructor 호출');

}

render() {

console.log('부모컴포넌트 => render 호출');

return <Child />;

}

componentDidMount() {

console.log('부모컴포넌트 => componentDidMount 호출');

this.setState({ updated: true });

}

componentDidUpdate() {

console.log('부모컴포넌트 => componentDidUpdate 호출');

}

}

export default App;

2) Updating : Child.jsx

import { Component } from 'react';

class Child extends Component {

constructor(props) {

super(props);

this.state = {};

console.log('자식컴포넌트 => constructor 호출');

}

render() {

console.log('자식컴포넌트 => render 호출');

return null;

}

componentDidMount() {

console.log('자식컴포넌트 => componentDidMount 호출');

this.setState({ updated: true });

}

componentDidUpdate() {

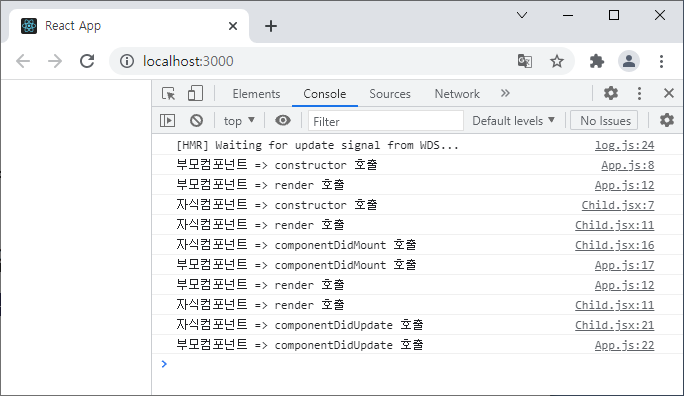
console.log('자식컴포넌트 => componentDidUpdate 호출');

}

}

export default Child;

2) Updating 종합 정리 : 실행 결과 확인 바랍니다.



마운트 단계에서 componentDidMount 메서드 호출 시 setState를 통해 컴포넌트를 업데이트시켰습니다.

==========================================================================

1. Unmounting : 언마운팅은 컴포넌트가 돔에서(화면에서) 제거되는 것입니다. componentWillUnmount 메서드를 실행해 보기 위해서 다음과 같은 코드를 실행하겠습니다. App 컴포넌트에서 componentDidMount 메서드를 이용해서 자식컴포넌트를 화면에서 제거합니다.

3) Unmounting : App.jsx (또는 App.js)

import React, { Component } from 'react';

import Child from './components/Child';

class App extends Component {

constructor(props) {

super(props);

this.state = { hasDestoryed: false };

console.log('부모컴포넌트 => constructor 호출');

}

render() {

console.log('부모컴포넌트 => render 호출', this.state);

return (

<React.Fragment>

{this.state.hasDestoryed ? null : <Child />}

</React.Fragment>

);

}

componentDidMount() {

console.log('부모컴포넌트 => componentDidMount 호출');

this.setState({ hasDestoryed: true });

}

componentDidUpdate() {

console.log('부모컴포넌트 => componentDidUpdate 호출');

}

componentWillUnmount() {

console.log('부모컴포넌트 => componentWillUnmount 호출');

}

}

export default App;

3) Unmounting : Child.jsx

import React, { Component } from 'react';

import Descendant from './Descendant';

class Child extends Component {

constructor(props) {

super(props);

console.log('자식컴포넌트 => constructor 호출');

}

render() {

console.log('자식컴포넌트 => render 호출');

return <Descendant />;

}

componentDidMount() {

console.log('자식컴포넌트 => componentDidMount 호출');

}

componentDidUpdate() {

console.log('자식컴포넌트 => componentDidUpdate 호출');

}

componentWillUnmount() {

console.log('자식컴포넌트 => componentWillUnmount 호출');

}

}

export default Child;

3) Unmounting : Descendant.jsx

import { Component } from 'react';

class Descendant extends Component {

constructor(props) {

super(props);

console.log('자손컴포넌트 => constructor 호출');

}

render() {

console.log('자손컴포넌트 => render 호출');

return null;

}

componentDidMount() {

console.log('자손컴포넌트 => componentDidMount 호출');

}

componentDidUpdate() {

console.log('자손컴포넌트 => componentDidUpdate 호출');

}

componentWillUnmount() {

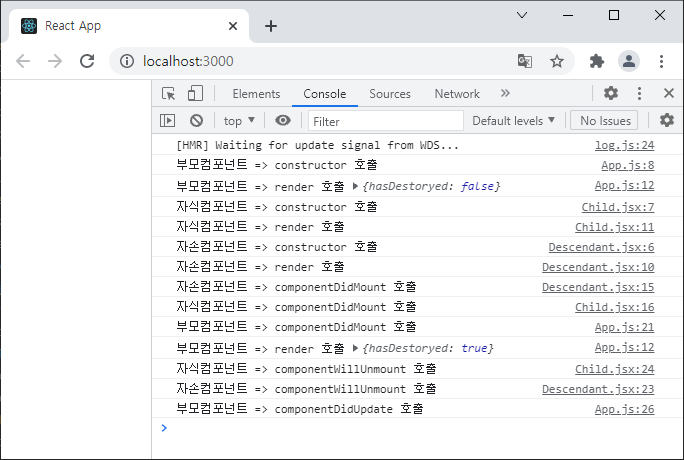
console.log('자손컴포넌트 => componentWillUnmount 호출');

}

}

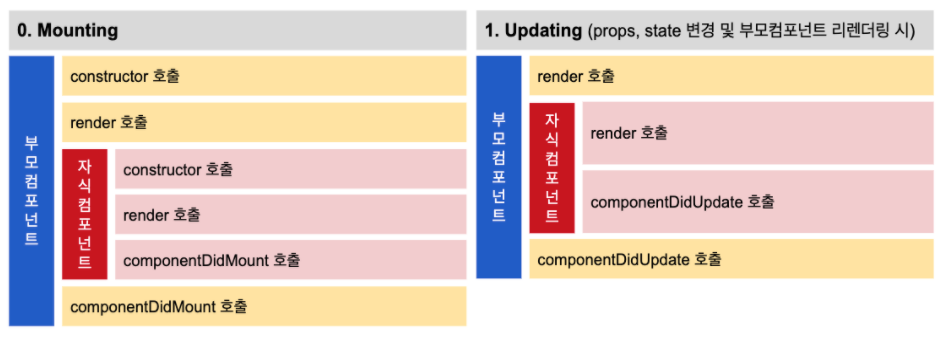
export default Descendant;

3) Unmounting 종합 정리 : 실행 결과 확인 바랍니다.

==========================================================================

[종합 정리] 라이프사이클 메서드 호출 순서

마운팅과 업데이팅 단계에서 메서드 호출 순서를 그림으로 정리하면 다음과 같습니다.



[참고 웹사이트]

1. https://ko.reactjs.org/docs/react-component.html
2. https://hogni.tistory.com/141
3. https://araikuma.tistory.com/486