Tutorial 12: ReactJS (4)

Objectives

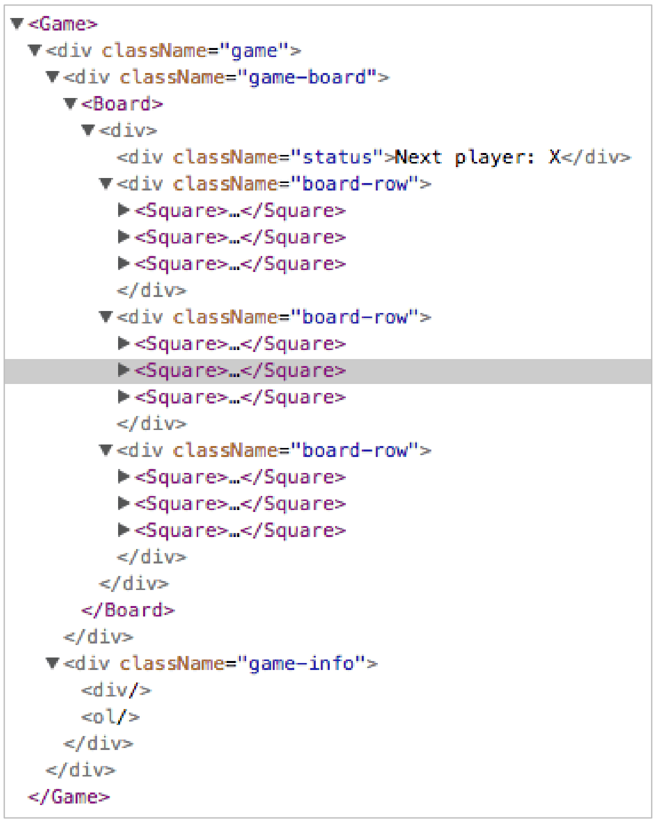
In this tutorial, we focus on practicing:

- Component lifecycle in use
- Refactor react components by **lifting state up** into a parent component
- Fetch data from APIs

IFY

IFY 1: Developer Tools

The React Devtools extension for Chrome and Firefox lets you inspect a React component tree with your browser's developer tools.

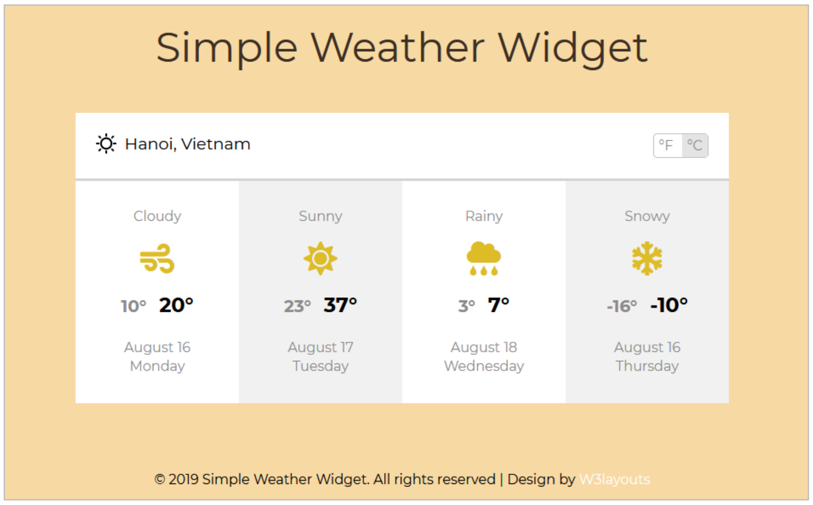


```
▼<Game>
  ▼<div className="game">
    ▼<div className="game-board">
      ▼<Board>
        ▼<div>
           <div className="status">Next player: X</div>
          ▼<div className="board-row">
            ▶<Square>…</Square>
            ▶<Square>…</Square>
            ▶<Square>…</Square>
           </div>
          ▼<div className="board-row">
            ▶<Square>…</Square>
            ▶<Square>…</Square>
            ▶<Square>…</Square>
           </div>
          ▼<div className="board-row">
            ▶<Square>…</Square>
            ▶<Square>…</Square>
            ▶<Square>…</Square>
           </div>
         </div>
       </Board>
     </div>
    ▼<div className="game-info">
       <div/>
       <ol/>
     </div>
   </div>
 </Game>
```

- The React DevTools let you check the props and the state of your React components.
- After installing React DevTools, you can right-click on any element on the page, click "Inspect" to open the developer tools, and the React tabs ("⚛ Components" and "⚛ Profiler") will appear as the last tabs to the right. Use "⚛ Components" to inspect the component tree.

Tutorial Exercises

Recall, in previous tutorial, we created a react app named Simple Weather Widget but **fixed** data. In this tutorial, we continue to fetch **dynamic** data from external APIs.



Exercise 1: Weather App – finishing up (30 mins)

- Refactor weather app (HTML version) into react with components.
- Decide what are the data (state & props) for each component & how data passed between components
- Handle event: user click to switch between °C and °F

Recall:

1. **Hint**: always use `setState()` to update the state. To understand more about *Why mutability is* important (read https://reactjs.org/tutorial/tutorial.html#why-immutability-is-important)
2. **Note: Lifting state up**
- *To collect data from multiple children, or*
- *To have two child components communicate with each other,*

→ *you need to declare the shared state in their parent component instead. The parent component can pass the state back down to the children by using props; this keeps the child components in sync with each other and with the parent component.*

Exercise 2: Weather App – fetching data (30 mins)

Let’s make our data dynamic by using data from an open weather API provided by openweathermap.org.

- Read the API docs: https://openweathermap.org/forecast5

Example: weather of Hanoi.

http://api.openweathermap.org/data/2.5/forecast?q=Hanoi&appId=92d7508ae3e2a43dc07cb1c28e5a3a7c&units=metric

- Populate the result returned by the API

Note: You can also use the icon provided by openweathermap. Example: icon 04n will have the url: http://openweathermap.org/img/w/04n.png

Homeworks: FlashCards

Recall: in some previous tutorials, we developed the server APIs for the FlashCards application but with server-rendering.

Similar to Weather app, continue with FlashCards react app to make things dynamic consuming the returned data from these server APIs.