

Section 7 : Debugging React Apps

Saturday, October 12, 2024 11:53 PM

- ▶ Making Sense of **React Error Messages**
- ▶ Finding **Logical Errors** via the Browser **DevTools & Debugger**
- ▶ Enabling React's **Strict Mode**
- ▶ Using the **React DevTools** for Application Analysis & Manipulation

Understanding React Error Message

Initial Investment: 10000
Annual Investment: 1200
Expected Return: 6
Duration: 10

Year	Investment Value	Interest (Year)	Total Interest	Invested Capital
1	\$11,800	\$600	\$600	\$11,200
2	\$13,708	\$708	\$1,308	\$12,400
3	\$15,730	\$822	\$2,130	\$13,600
4	\$17,874	\$944	\$3,074	\$14,800
5	\$20,147	\$1,072	\$4,147	\$16,000
6	\$22,556	\$1,209	\$5,356	\$17,200

So far our application works fine. Our console has not error at all.

now press 0 to the duration and our application crashes! Go to console and understand the error.

```
✖ Uncaught TypeError: Cannot read properties of undefined (reading 'valueEndOfYear')
  at Results (Results.jsx:8:16)
  at renderWithHooks (react-dom-client.js?v=87f26b8d:11566:26)
  at updateFunctionComponent (react-dom-client.js?v=87f26b8d:14600:28)
  at beginWork (react-dom-client.js?v=87f26b8d:15942:22)
  at HTMLUnknownElement.callCallback2 (react-dom-client.js?v=87f26b8d:3672:22)
  at Object.invokeGuardedCallbackDev (react-dom-client.js?v=87f26b8d:3697:24)
  at invokeGuardedCallback (react-dom-client.js?v=87f26b8d:3731:39)
  at beginWork$1 (react-dom-client.js?v=87f26b8d:19791:15)
  at performUnitOfWork (react-dom-client.js?v=87f26b8d:19224:20)
  at workLoopSync (react-dom-client.js?v=87f26b8d:19163:13)

✖ Uncaught TypeError: Cannot read properties of undefined (reading 'valueEndOfYear')
  at Results (Results.jsx:8:16)
```

```

at beginWork$1 (react-dom_client.js?v=87f26b8d:19791:15)
at performUnitOfWork (react-dom_client.js?v=87f26b8d:19224:20)
at workLoopSync (react-dom_client.js?v=87f26b8d:19163:13)

✖️ ▶️ Uncaught TypeError: Cannot read properties of undefined (reading 'valueEndOfYear')
    at Results (Results.jsx:8:16)
    at renderWithHooks (react-dom_client.js?v=87f26b8d:11566:26)
    at updateFunctionComponent (react-dom_client.js?v=87f26b8d:14600:28)
    at beginWork (react-dom_client.js?v=87f26b8d:15942:22)
    at HTMLUnknownElement.callCallback2 (react-dom_client.js?v=87f26b8d:3672:22)
    at Object.invokeGuardedCallbackDev (react-dom_client.js?v=87f26b8d:3697:24)
    at invokeGuardedCallback (react-dom_client.js?v=87f26b8d:3731:39)
    at beginWork$1 (react-dom_client.js?v=87f26b8d:19791:15)
    at performUnitOfWork (react-dom_client.js?v=87f26b8d:19224:20)
    at workLoopSync (react-dom_client.js?v=87f26b8d:19163:13)

✖️ ▶️ The above error occurred in the <Results> component:          console.js:288

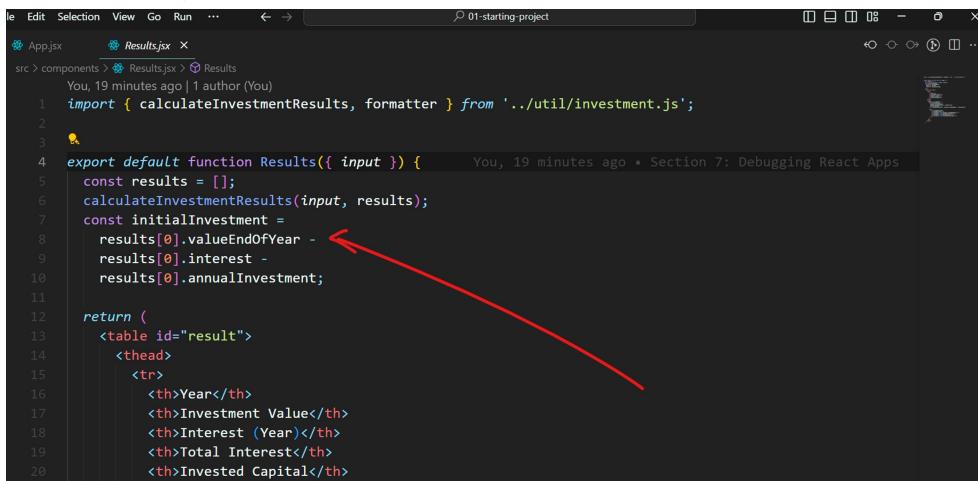
    at Results (http://localhost:5173/src/components/Results.jsx:18:35)
    at App (http://localhost:5173/src/App.jsx:24:37)

Consider adding an error boundary to your tree to customize error handling behavior.
Visit https://reactjs.org/link/error-boundaries to learn more about error boundaries.

✖️ ▶️ Uncaught TypeError: Cannot read properties of undefined (reading 'valueEndOfYear')
    at Results (Results.jsx:8:16)          react-dom_client.js?v=87f26b8d:9143
    at renderWithHooks (react-dom_client.js?v=87f26b8d:11566:26)

```

Go to Result Component line no 8



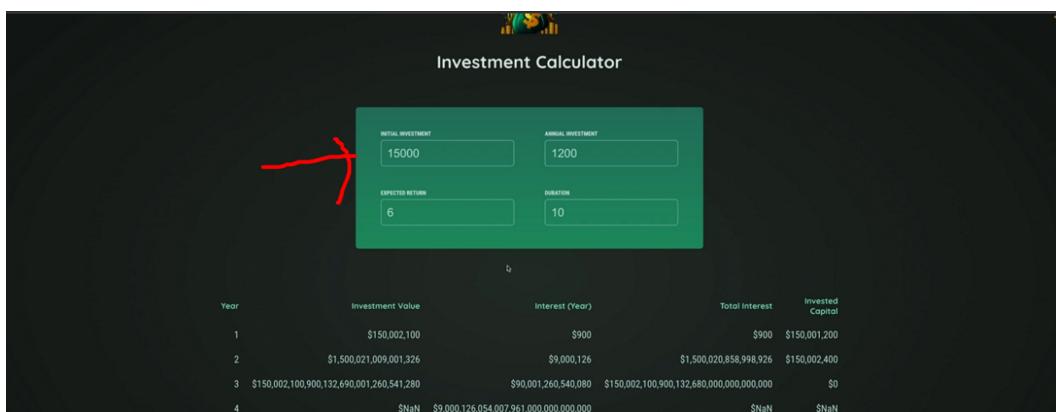
```

File Edit Selection View Go Run ... ← → 01-starting-project
App.jsx Results.jsx ✘
src > components > Results.jsx ⓘ Results
You, 19 minutes ago | 1 author (You)
1 import { calculateInvestmentResults, formatter } from '../util/investment.js';
2
3
4 export default function Results({ input }) {
5   const results = [];
6   calculateInvestmentResults(input, results);
7   const initialInvestment =
8     results[0].valueEndOfYear - ↗
9     results[0].interest -
10    results[0].annualInvestment;
11
12   return (
13     <table id="result">
14       <thead>
15         <tr>
16           <th>Year</th>
17           <th>Investment Value</th>
18           <th>Interest (Year)</th>
19           <th>Total Interest</th>
20           <th>Invested Capital</th>

```

Using the Browser Debugger & Breakpoints

Not all error shows to the console. Some are logical error.
 Write 15000 in the investment input and you can see everything meesed up.
 If you encounter this kinda error, you should ask yourself which part of the code
 Could be responsible for this error?



Investment Calculator

Year	Investment Value	Interest (Year)	Total Interest	Invested Capital
1	\$150,002,100	\$900	\$900	\$150,001,200
2	\$1,500,021,009,001,526	\$9,000,126	\$1,500,020,858,998,926	\$150,002,400
3	\$150,002,100,900,132,690,001,260,541,280	\$90,001,260,540,080	\$150,002,100,900,132,680,000,000,000,000	\$0
4	\$NaN	\$9,000,126,054,007,961,000,000,000,000	\$NaN	\$NaN
5	\$NaN	\$NaN	\$NaN	\$NaN
6	\$NaN	\$NaN	\$NaN	\$NaN
7	\$NaN	\$NaN	\$NaN	\$NaN
8	\$NaN	\$NaN	\$NaN	\$NaN
9	\$NaN	\$NaN	\$NaN	\$NaN

Which part of your code could be responsible for this error?

Go to Browser Sources tab and add a break point to line 16. and investigate.

```

    Investment Calculator

    INITIAL INVESTMENT: 1000 ANNUAL INVESTMENT: 1200
    EXPECTED RETURN: 6 DURATION: 10

    Year Investment Value Interest (Year) Total Interest Invested Capital
    1 $10,001,260 $60 $60
    2 $10,001,260,601,276 $600,076 $10,001,260,601,276
    3 $NaN $600,075,636,077 $NaN
    4 $NaN $NaN $NaN
    5 $NaN $NaN $NaN
    6 $NaN $NaN $NaN
    7 $NaN $NaN $NaN
    8 $NaN $NaN $NaN
    9 $NaN $NaN $NaN
    10 $NaN $NaN $NaN

    function handleChange(inputIdentifier) {
      setUserInput((prevUserInput) => {
        return {
          ...prevUserInput,
          [inputIdentifier]: newValue,
        };
      });
    }

    return (
      <>
      <Header />
      <UserInput userInput={userInput}>
    
```

Line 16, Column 5 (From App.jsx) Coverage: n/a

Breakpoints

Understanding React's Strict Mode



Investment Calculator

INITIAL INVESTMENT	ANNUAL INVESTMENT
<input type="text" value="100020"/>	<input type="text" value="1200"/>
EXPECTED RETURN	DURATION
<input type="text" value="6"/>	<input type="text" value="10"/>

Year	Investment Value	Interest (Year)	Total Interest	Invested Capital
1	\$11,800	\$600	\$600	\$11,200
2	\$13,708	\$708	\$1,308	\$12,400
3	\$15,730	\$822	\$2,130	\$13,600
4	\$17,874	\$944	\$3,074	\$14,800
5	\$20,147	\$1,072	\$4,147	\$16,000
6	\$22,556	\$1,209	\$5,356	\$17,200
7	\$25,109	\$1,353	\$6,709	\$18,400

Our component renders perfectly. But as soon as we start typing to the Initial Investment, We get this error.

The list keeps adding and adding.

We could get this error initially we if use React Strict mode.

```

Results.jsx    index.jsx  ●
1 import ReactDOM from 'react-dom/client';
2
3 import App from './App.jsx';
4 import './index.css';
5
6
7 ReactDOM.createRoot(document.getElementById('root')).render(
8   <StrictMode>
9     <App />
10   </StrictMode>
11 );
12

```

Strict Mode does couple of things behind the scene to help us catch the certain errors. It executes every component function twice. It only does that for development. If every component gets executed twice , it can help you identify error like this

Year	Investment Value	Interest (Year)	Total Interest	Invested Capital
1	\$11,800	\$600	\$600	\$11,200
2	\$13,708	\$708	\$1,308	\$12,400
3	\$15,730	\$822	\$2,130	\$13,600
4	\$17,874	\$944	\$3,074	\$14,800
5	\$20,147	\$1,072	\$4,147	\$16,000
6	\$22,556	\$1,209	\$5,356	\$17,200
7	\$25,109	\$1,353	\$6,709	\$18,400
8	\$27,815	\$1,507	\$8,215	\$19,600
9	\$30,684	\$1,669	\$9,884	\$20,800
10	\$33,725	\$1,841	\$11,725	\$22,000
1	\$11,800	\$600	\$600	\$11,200
2	\$13,708	\$708	\$1,308	\$12,400
3	\$15,730	\$822	\$2,130	\$13,600
4	\$17,874	\$944	\$3,074	\$14,800
5	\$20,147	\$1,072	\$4,147	\$16,000
6	\$22,556	\$1,209	\$5,356	\$17,200
7	\$25,109	\$1,353	\$6,709	\$18,400
8	\$27,815	\$1,507	\$8,215	\$19,600
9	\$30,684	\$1,669	\$9,884	\$20,800

Using the React DevTools (Browser Extension)

The screenshot displays the Investment Calculator application interface alongside the React DevTools extension. The application has a dark theme with a green header bar containing a dollar sign icon and the text "Investment Calculator". Below this is a light blue input form with fields for "INITIAL INVESTMENT" (10000), "ANNUAL INVESTMENT" (1200), "EXPECTED RETURN" (6), and "DURATION" (10). To the right, a sidebar titled "Components" shows the component hierarchy: "App" (selected) → "Header" → "UserInput" (highlighted in blue) → "Results". A search bar at the top of the sidebar contains the placeholder "Search (text or /regex/)". At the bottom of the sidebar, there is a preview of the "UserInput" component's state, which is identical to the table shown above. The preview includes the table header and the first six rows of data.

Year	Investment Value	Interest (Year)	Total Interest	Invested Capital
1	\$11,800	\$600	\$600	\$11,200
2	\$13,708	\$708	\$1,308	\$12,400
3	\$15,730	\$822	\$2,130	\$13,600
4	\$17,874	\$944	\$3,074	\$14,800
5	\$20,147	\$1,072	\$4,147	\$16,000
6	\$22,556	\$1,209	\$5,356	\$17,200

The screenshot shows the Investment Calculator application running in a browser. On the left is the UI, which includes a header with a logo and the title "Investment Calculator". Below the header are four input fields: "INITIAL INVESTMENT" (10000), "ANNUAL INVESTMENT" (1200), "EXPECTED RETURN" (6), and "DURATION" (10). To the right is the developer tools' component inspector. The "UserInput" component is selected. The "props" section shows the following code:

```

props
onchange: f handleChange() {}
- userInput: {annualInvestment: 1200, duration: 10, e...
  annualInvestment: 1200
  duration: '10'
  expectedReturn: 6
  initialInvestment: 10000
  new entry: ""

```

This screenshot is similar to the one above, but the "duration" prop has been changed to "12" in the developer tools. A red arrow points from the "duration: 12" line in the props section to the "DURATION" input field in the UI. The UI now displays "12" in the "DURATION" field. The developer tools still show "10" as the current value of the "duration" prop.

You can also edit those prop value and show the UI reflection.