

Sunday, 3 September 2023

Context API

Problem: Let us suppose we have 4 nested components and we want to pass one state from 1 component to last component. Then we have to pass it as a prop from 1st -> 2nd component then 2nd -> 3rd then 3rd -> 4th. So this is known as props drilling passing props from parent to grandchildren. This can be very complex when you have a lot of nesting components. To solve this problem, Context API and Redux are created.

What is the Context API?

- The Context API is a feature in React that provides a way to share data between components without having to pass props explicitly through each level of the component tree.

Motivation for Using Context:

- Context is particularly useful when you have data that is needed by many components at different levels of your application, such as themes, user authentication, or language preferences.

Creating a Context:

- To create a context, you use the `createContext` function provided by React. This function returns an object with `Provider` and `Consumer` components.

javascript

 Copy code

```
const MyContext = React.createContext();
```

Provider Component:

- The `Provider` component is responsible for making the context data available to all components within its subtree. It takes a `value` prop to specify the data you want to share.

javascript


 Copy code

```
<MyContext.Provider value={/* your data */}>
  /* Components that can access the context data */
</MyContext.Provider>
```

Consumer Component (Functional Component):

- In functional components, you can use the `useContext` hook to access the context data.

javascript

 Copy code

```
import React, { useContext } from 'react';

const value = useContext(MyContext);
// Use the context data
```

Context API can be implemented in number of ways. Watch class recording to get one of the best and simple way in which we create context in separate file and just export it. So it can be used as Provider in other files.