Chapter 11: Communicate Between Components

Section 11.1: Communication between Stateless Functional Components

In this example we will make use of `Redux` and `React Redux` modules to handle our application state and for auto re-render of our functional components., And ofcourse `React` and `React Dom`

You can checkout the completed demo here

In the example below we have three different components and one connected component

- **UserInputForm**: This component display an input field And when the field value changes, it calls `inputChange` method on `props` (which is provided by the parent component) and if the data is provided as well, it displays that in the input field.
- **UserDashboard**: This component displays a simple message and also nests `UserInputForm` component, It also passes `inputChange` method to `UserInputForm` component, `UserInputForm` component inturn makes use of this method to communicate with the parent component.
    - **UserDashboardConnected**: This component just wraps the `UserDashboard` component using `ReactRedux connect` method., This makes it easier for us to manage the component state and update the component when the state changes.
- **App**: This component just renders the `UserDashboardConnected` component.

```
const UserInputForm = (props) => {

  let handleSubmit = (e) => {
    e.preventDefault();
  }

  return(
    <form action="" onSubmit={handleSubmit}>
      <label htmlFor="name">Please enter your name</label>
      <br />
      <input type="text" id="name" defaultValue={props.data.name || ''} onChange={
props.inputChange } />
    </form>
  )

}


const UserDashboard = (props) => {

  let inputChangeHandler = (event) => {
    props.updateName(event.target.value);
  }

  return(
    <div>
      <h1>Hi { props.user.name || 'User' }</h1>
      <UserInputForm data={props.user} inputChange={inputChangeHandler} />
    </div>
```

```
  )

}

const mapStateToProps = (state) => {
  return {
    user: state
  };
}
const mapDispatchToProps = (dispatch) => {
  return {
    updateName: (data) => dispatch( Action.updateName(data) ),
  };
};

const { connect, Provider } = ReactRedux;
const UserDashboardConnected = connect(
  mapStateToProps,
  mapDispatchToProps
)(UserDashboard);



const App = (props) => {
  return(
    <div>
      <h1>Communication between Stateless Functional Components</h1>
      <UserDashboardConnected />
    </div>
  )
}



const user = (state={name: 'John'}, action) => {
  switch (action.type) {
    case 'UPDATE_NAME':
      return Object.assign( {}, state, {name: action.payload}  );

    default:
      return state;
  }
};

const { createStore } = Redux;
const store = createStore(user);
const Action = {
  updateName: (data) => {
    return { type : 'UPDATE_NAME', payload: data }
  },
}


ReactDOM.render(
  <Provider store={ store }>
    <App />
  </Provider>,
  document.getElementById('application')
);
```

JS Bin URL