

## 2.12 Provider

Let's put together all the things we've done so far and make our apps work. So far we have defined **action objects** and created **action creators** that create action objects. And when an action occurs, we have created **reducers** that actually treat and return a new state. We then created a **container component** that connects each of the presentation components to the Redux store.

Now every container component needs a way to access the store, which is what the **Provider** does. The Provider component **wraps the entire application and allows subcomponents to communicate with the store via connect()**.

The top-level component of our app, **App.js**, looks like this:

```

import React, { Component } from 'react';
import { createStore } from 'redux';
import { Provider } from 'react-redux';
import TeslaCarContainer from './containers/TeslaCarContainer';
import TeslaStatsContainer from './containers/TeslaStatsContainer';
import TeslaSpeedCounterContainer from './containers/TeslaSpeedCounterContainer';
import TeslaTempCounterContainer from './containers/TeslaTempCounterContainer';
import TeslaClimateContainer from './containers/TeslaClimateContainer';
import TeslaWheelsContainer from './containers/TeslaWheelsContainer';
import TeslaNotice from './components/TeslaNotice/TeslaNotice';
import Header from './components/Header/Header';
import './App.css';
import appReducer from './reducers/teslaRangeApp';

const store = createStore(appReducer);

class App extends Component {
  render() {
    return (
      <Provider store={store}>
        <div>
          <Header />
          <div className="wrapper">
            <form className="tesla-battery">
              <h1>Range Per Charge</h1>
              <TeslaCarContainer />
              <TeslaStatsContainer />
              <div className="tesla-controls cf">
                <TeslaSpeedCounterContainer />
                <div className="tesla-climate-conta">
                  <TeslaTempCounterContainer />
                  <TeslaClimateContainer />
                </div>
                <TeslaWheelsContainer />
              </div>
            </form>
          </div>
          <TeslaNotice />
        </div>
      </Provider>
    );
  }
}

export default App;

```

Bridge react and redux

Import Container Components

Import Presentational Components

Reducers

Create Redux Store by passing it the reducers

Provider component injects the store to all the child components

Redux Store

TeslaCar Container

TeslaStats Container

60	60D	75	75D	90D	P100D
246Mi	250Mi	297Mi	306Mi	336Mi	376Mi

TeslaSpeedCounter  
Speed  
55

TeslaTempCounter  
Outside Tempe  
20

TeslaClimate  
AC  
ON

TeslaWheels Container  
Wheel:  
19" 21"

```
import { getModelData } from '../services/BatteryService';
```

```

const initialState = {
  carstats: [
    { miles: 246, model: "60" },
    { miles: 250, model: "60D" },
    { miles: 297, model: "75" },
    { miles: 306, model: "75D" },
    { miles: 336, model: "90D" },
    { miles: 376, model: "P100D" }
  ],
  config: {
    speed: 55,
    temperature: 20,
    climate: true,
    wheels: 19
  }
};

```

```

    }
  }

function updateStats(state, newState) {
  return {
    ...state,
    config: newState.config,
    carstats: calculateStats(newState)
  }
}

function calculateStats(state) {
  const models = ['60', '60D', '75', '75D', '90D', 'P100D'];
  const dataModels = getModelData();
  return models.map(model => {
    const { speed, temperature, climate, wheels } = state.config;
    const miles = dataModels[model][wheels][climate ? 'on' : 'off'].speed[speed][temperature];
    return {
      model,
      miles
    };
  });
}

function appReducer(state = initialState, action) {
  switch (action.type) {
    case 'CHANGE_CLIMATE': {
      const newState = {
        ...state,
        config: {
          climate: !state.config.climate,
          speed: state.config.speed,
          temperature: state.config.temperature,
          wheels: state.config.wheels
        }
      };
      return updateStats(state, newState);
    }
    case 'SPEED_UP': {
      const newState = {
        ...state,
        config: {
          climate: state.config.climate,
          speed: action.value + action.step,
          temperature: state.config.temperature,
          wheels: state.config.wheels
        }
      };
      return updateStats(state, newState);
    }
    case 'SPEED_DOWN': {
      const newState = {
        ...state,
        config: {
          climate: state.config.climate,
          speed: action.value - action.step,
          temperature: state.config.temperature,
          wheels: state.config.wheels
        }
      };
      return updateStats(state, newState);
    }
  }
}

```

```

    case 'TEMPERATURE_UP': {
      const newState = {
        ...state,
        config: {
          climate: state.config.climate,
          speed: state.config.speed,
          temperature: action.value + action.step,
          wheels: state.config.wheels
        }
      };
      return updateStats(state, newState);
    }
    case 'TEMPERATURE_DOWN': {
      const newState = {
        ...state,
        config: {
          climate: state.config.climate,
          speed: state.config.speed,
          temperature: action.value - action.step,
          wheels: state.config.wheels
        }
      };
      return updateStats(state, newState);
    }
    case 'CHANGE_WHEEL': {
      const newState = {
        ...state,
        config: {
          climate: state.config.climate,
          speed: state.config.speed,
          temperature: state.config.temperature,
          wheels: action.value
        }
      };
      return updateStats(state, newState);
    }
    default:
      return state
  }
}

export default appReducer;

```