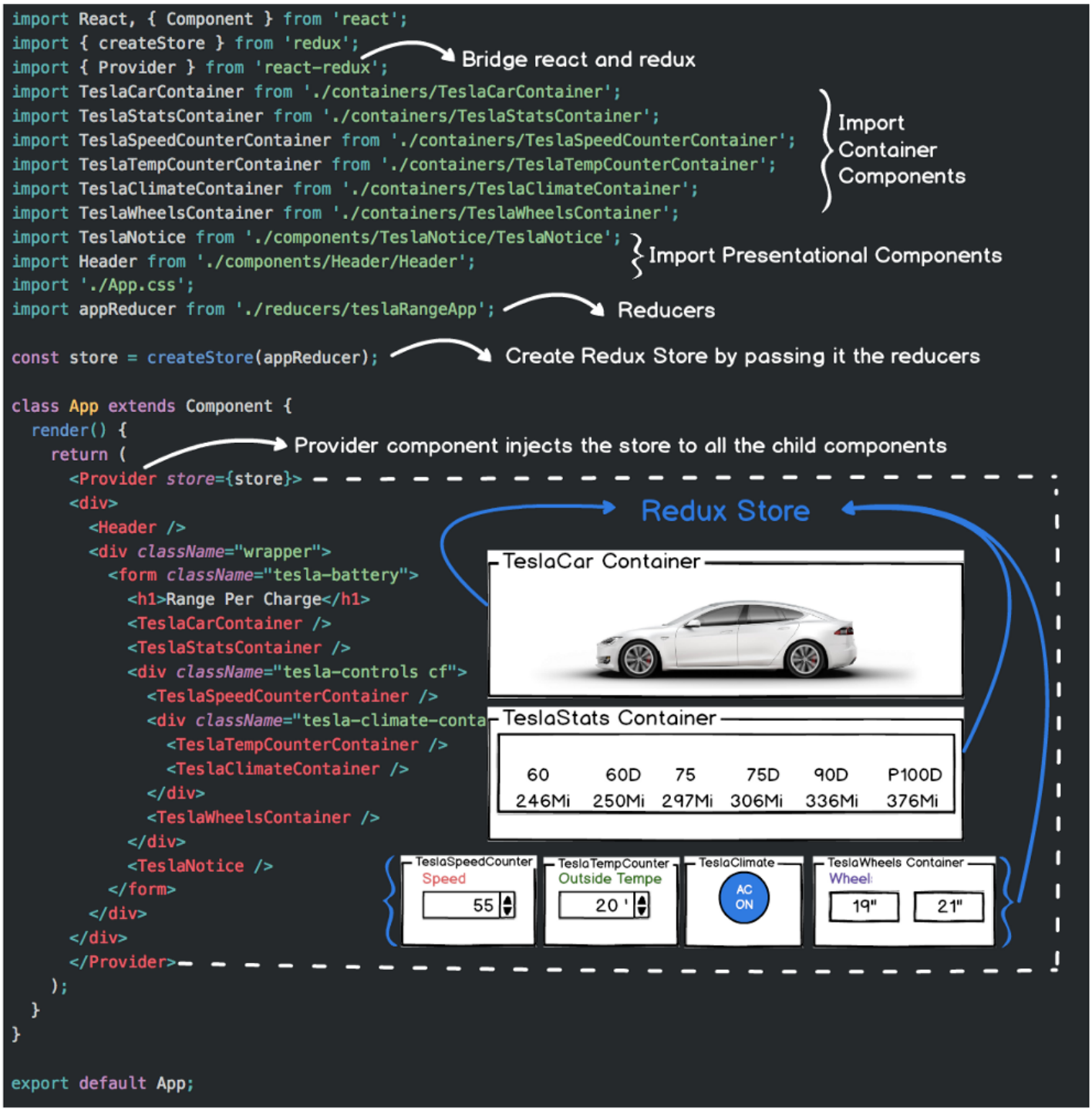
2.12 Provider

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 { 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;
```