

REACT & REDUX

---

# DESIGN PATTERNS

REACT

---

**BREAK UI INTO COMPONENT**

# DRAW BOXES AND NAMED IT

☐ Only show products in stock

Name	Price
Sporting Goods	
Football	\$49.99
Baseball	\$9.99
Basketball	\$29.99
Electronics	
iPod Touch	\$99.99
iPhone 5	\$399.99
Nexus 7	\$199.99

# WHAT SHOULD BE IT OWN COMPONENT ?

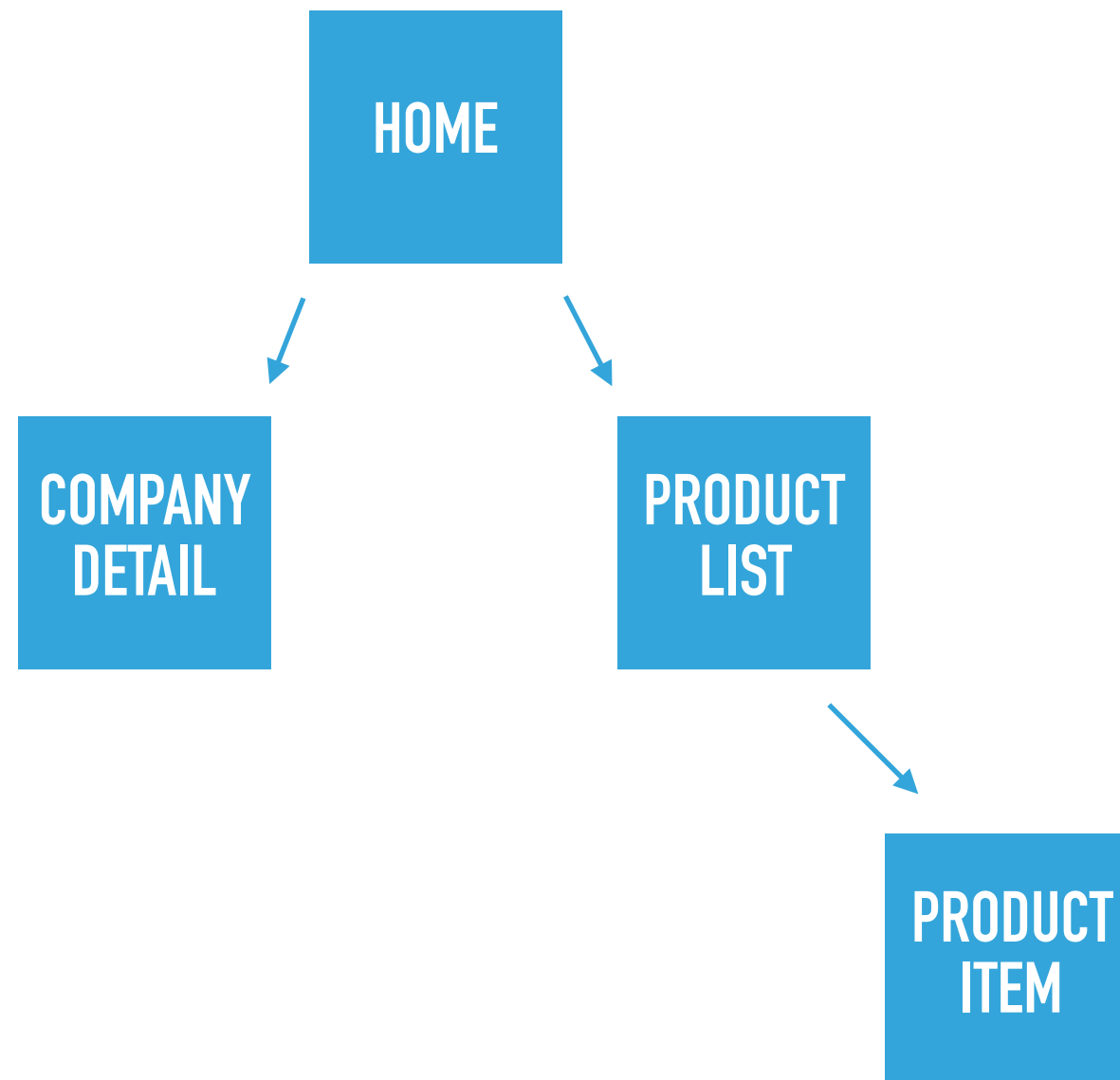
- ▶ **Single responsibility principle** (same as function or class)
- ▶ If component do more than one thing break it into **SubComponent**
- ▶ Map component to **data model**

REACT

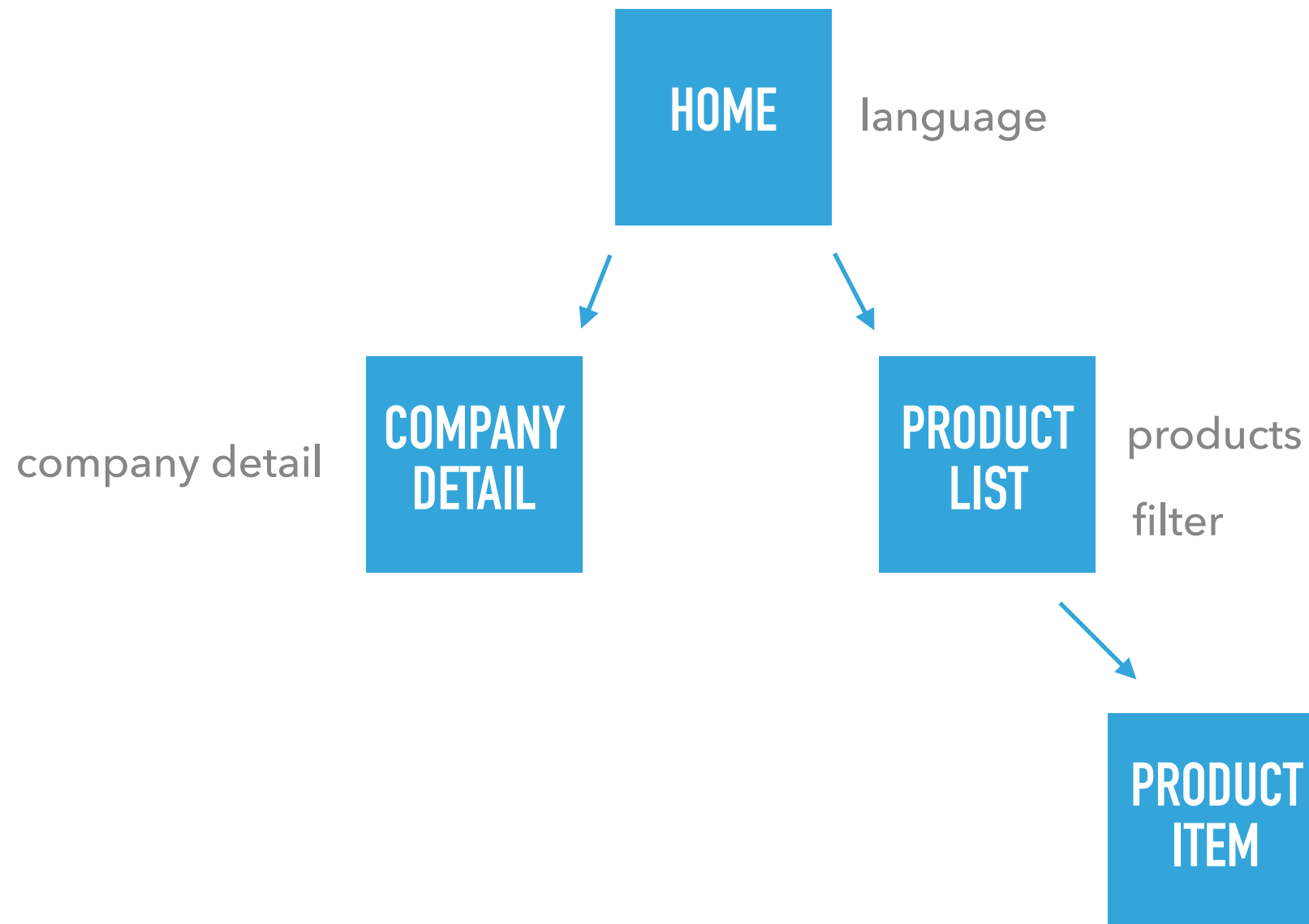
---

**DATA FLOW**

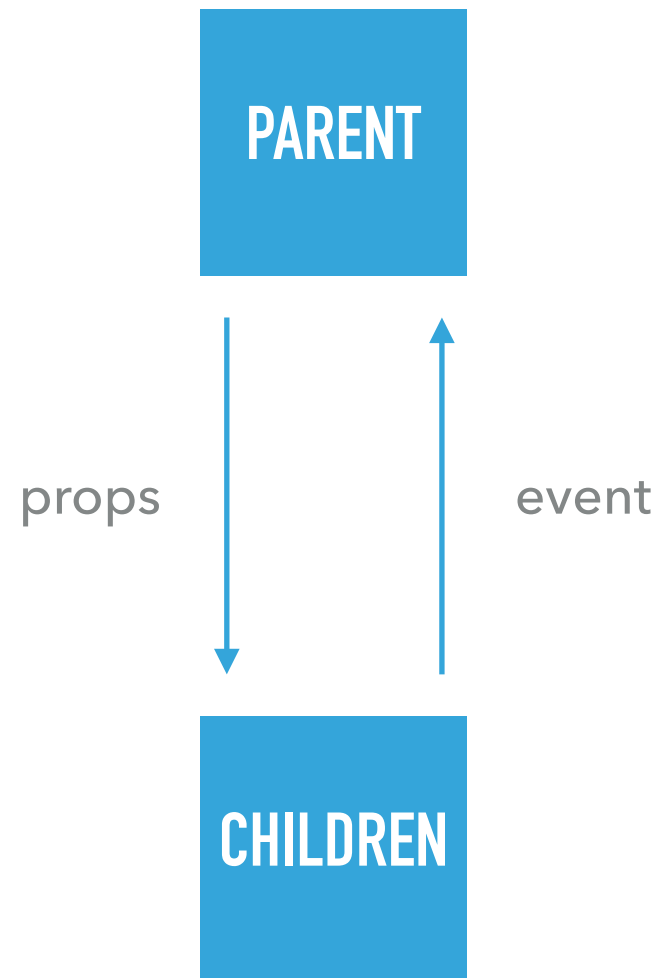
## IDENTIFY WHERE STATE SHOULD LIVE



# IDENTIFY WHERE STATE SHOULD LIVE

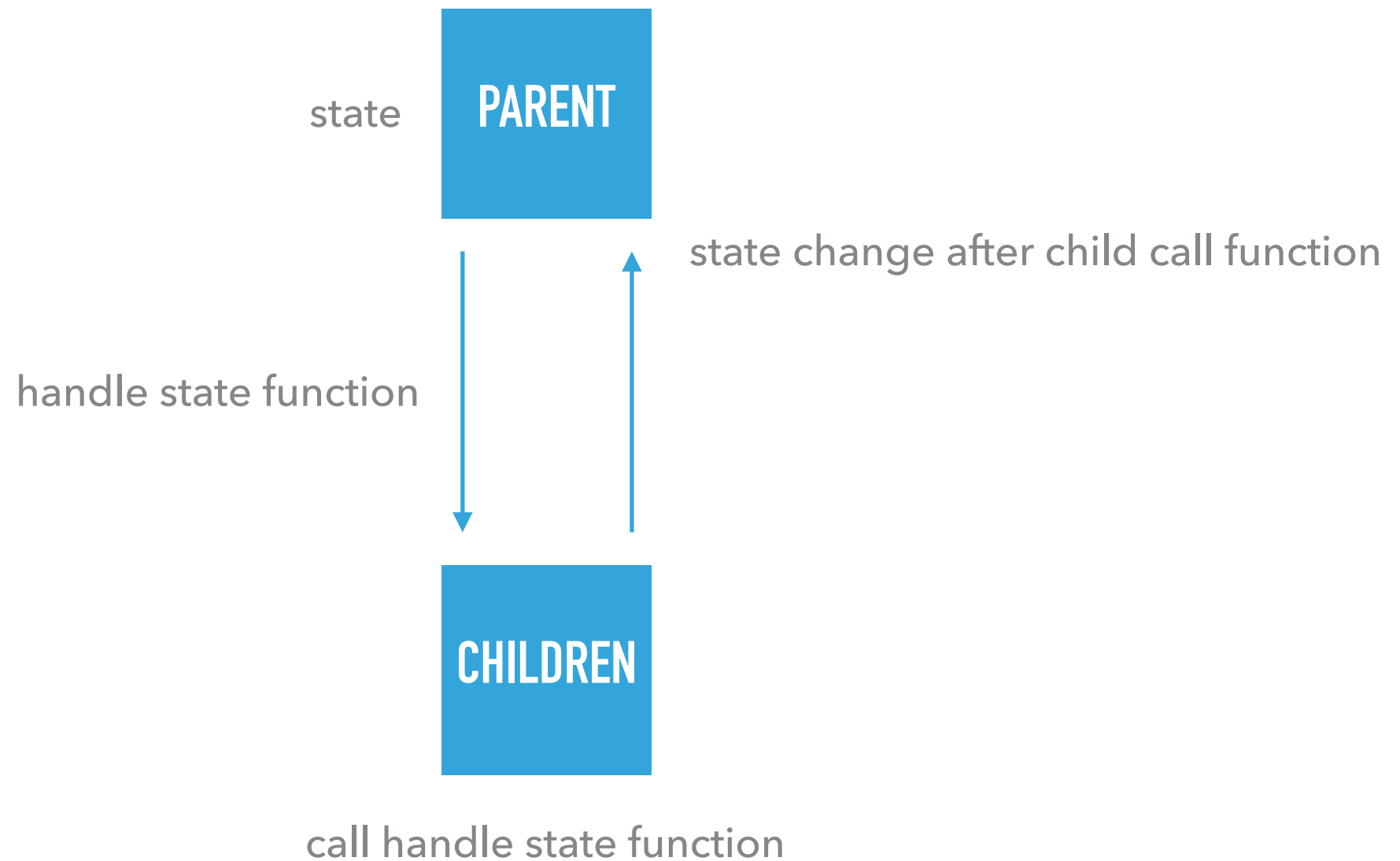


## ADD INVERSE DATA FLOW

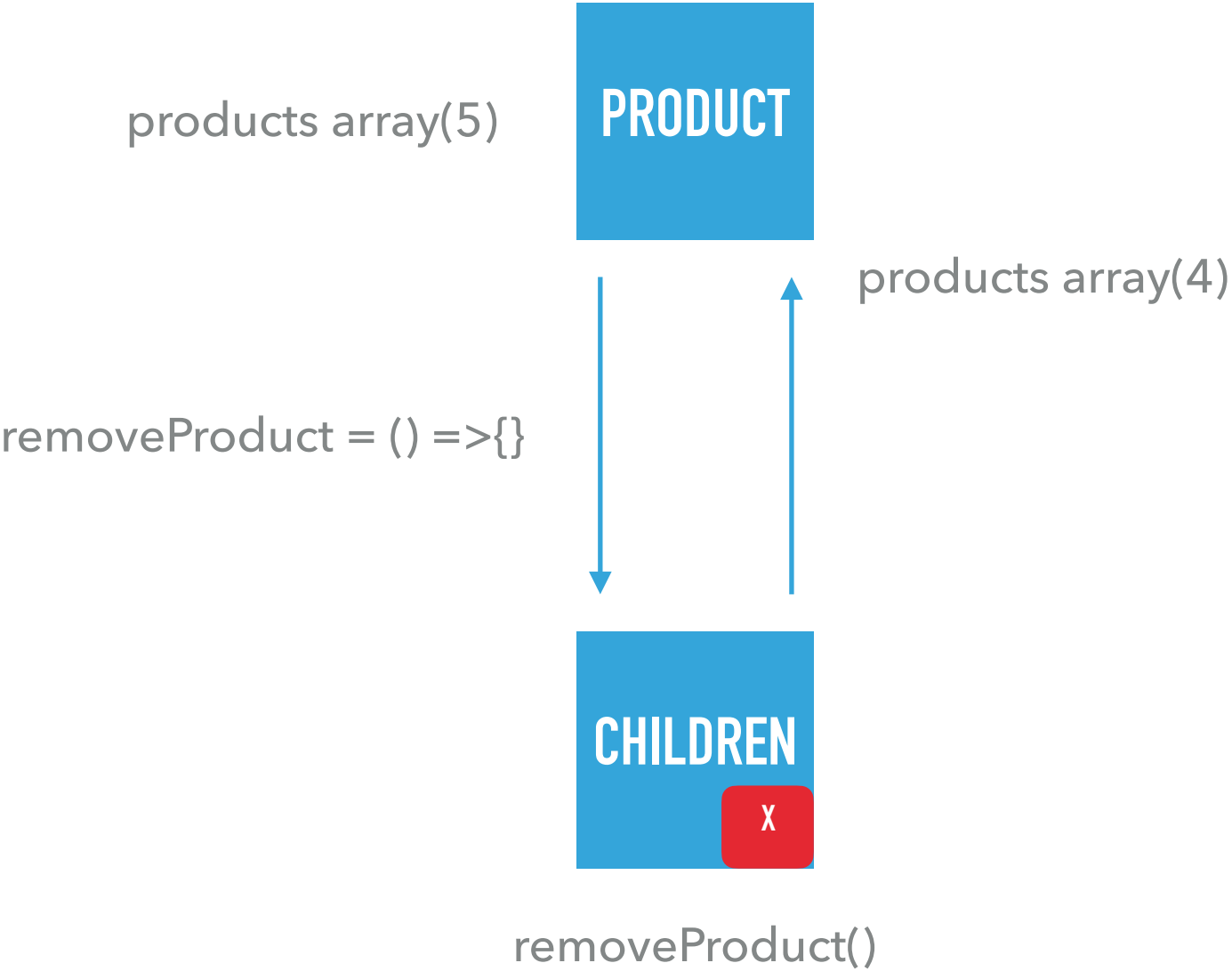




## ADD INVERSE DATA FLOW



# ADD INVERSE DATA FLOW



REACT

---

# TECHNIQUE & PATTERNS

REACT

---

# COMPONENT COMMUNICATION

# COMPOSITION

```
<App>  
  <Header>  
    <Navigation> ... </Navigation>  
  </Header>  
</App>
```

## COMPONENT COMMUNICATION

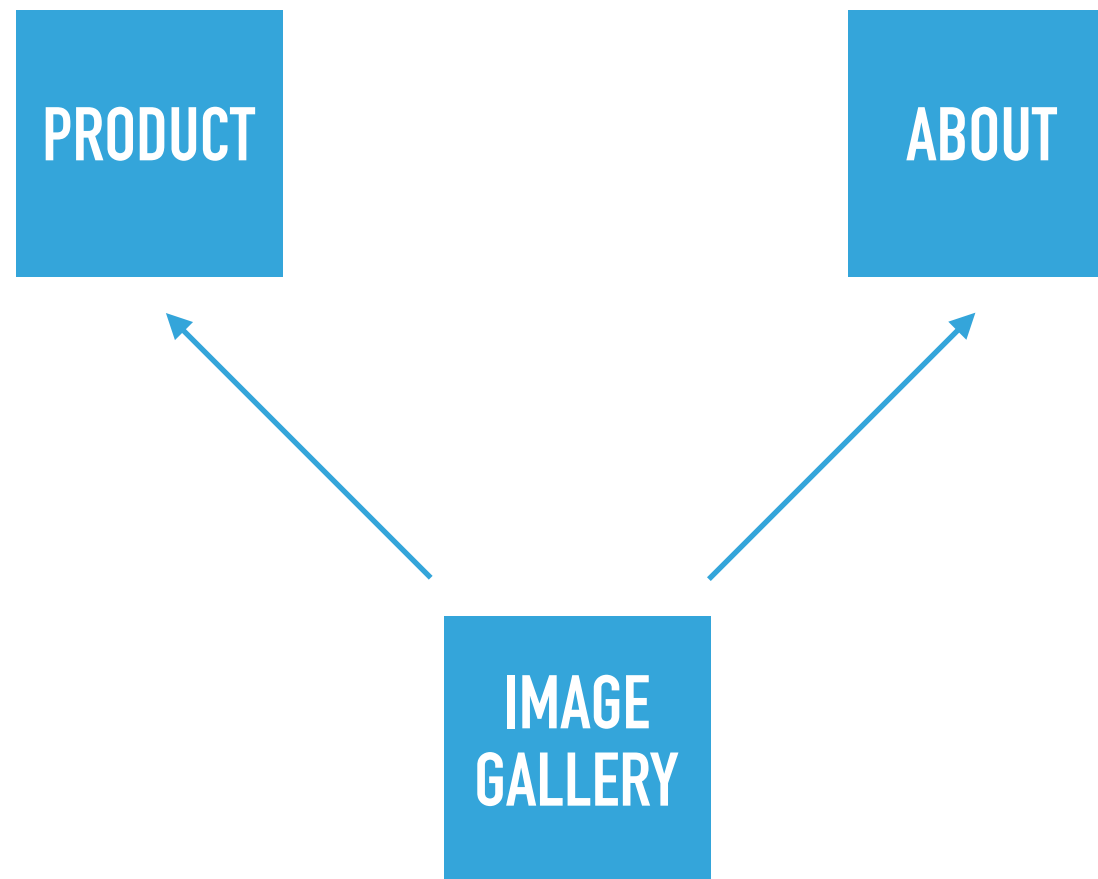
- ▶ No need to know how other component works
- ▶ Communicate by **props**

REACT

---

# COMPONENT REUSABILITY

# COMPONENT REUSABILITY





REACT

---

# CONTAINER COMPONENT & PRESENTATIONAL COMPONENT

# CONTAINER COMPONENT & PRESENTATIONAL COMPONENT

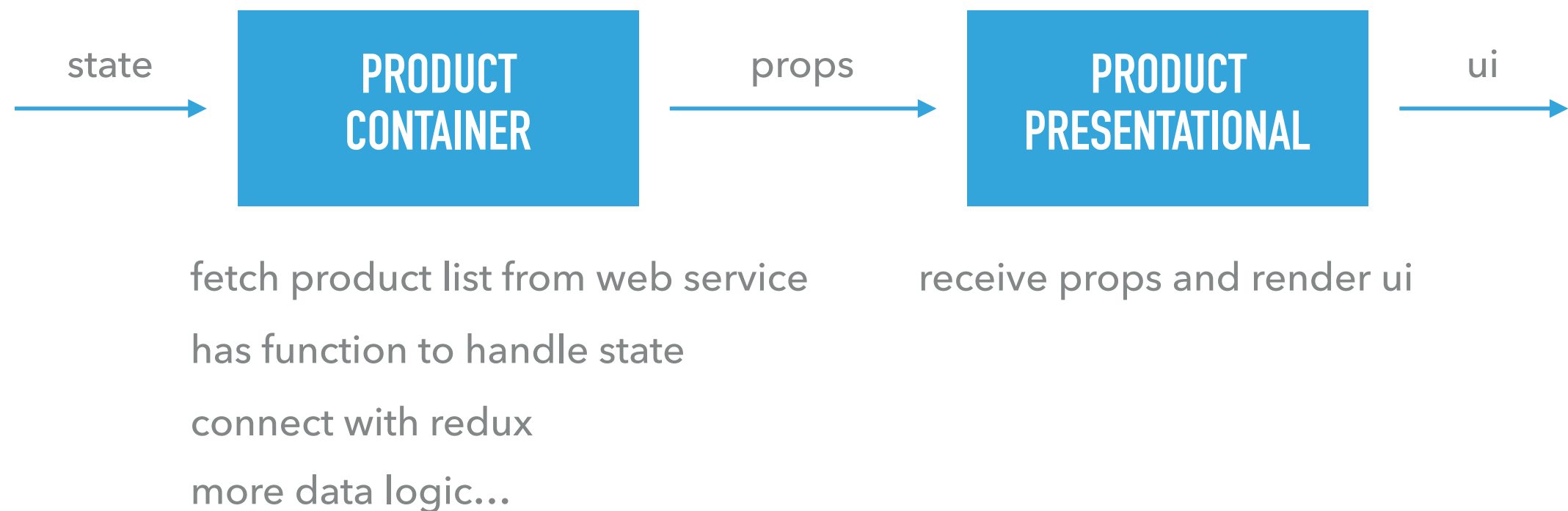
**CONTAINER**

define how things works

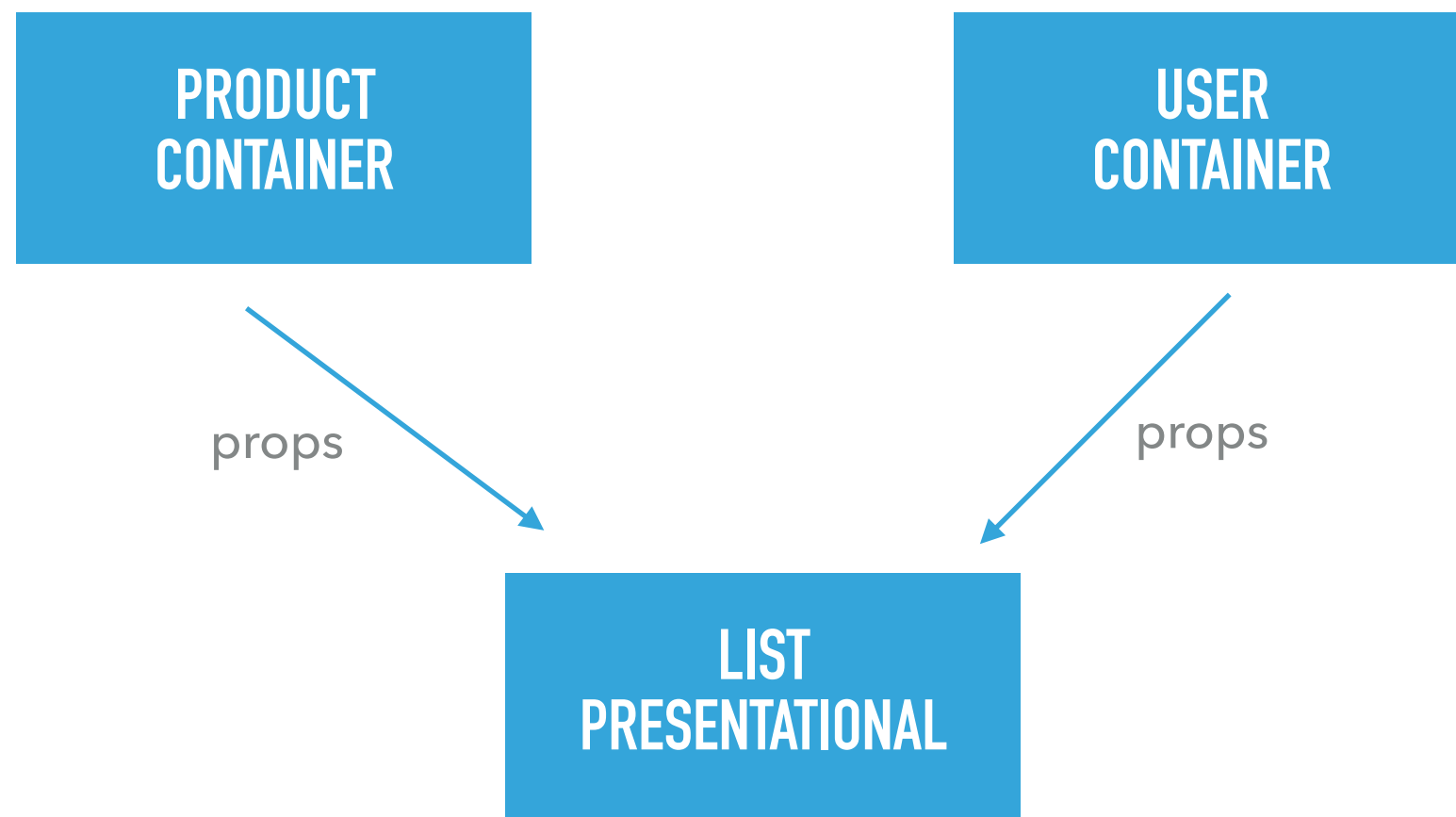
**PRESENTATIONAL**

define how things looks

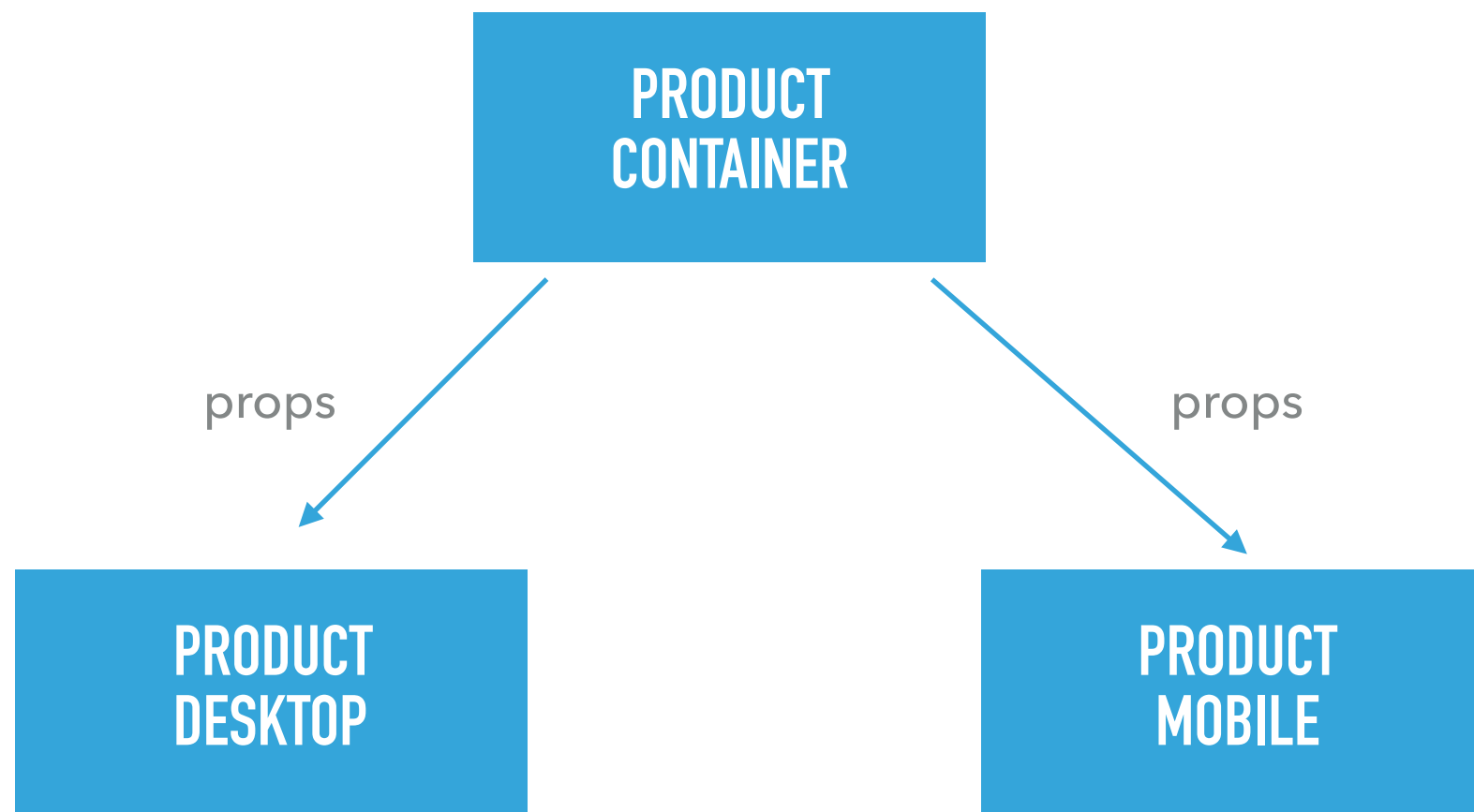
# CONTAINER COMPONENT & PRESENTATIONAL COMPONENT



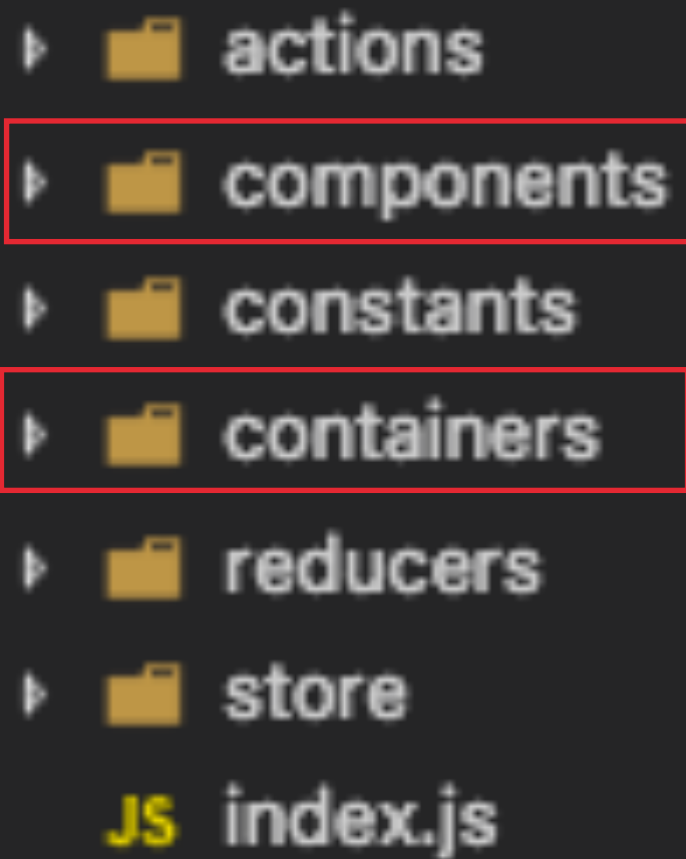
# CONTAINER COMPONENT & PRESENTATIONAL COMPONENT



# CONTAINER COMPONENT & PRESENTATIONAL COMPONENT



# FOLDER STRUCTURE



```
▶ actions
▶ components
▶ constants
▶ containers
▶ reducers
▶ store
JS index.js
```

A screenshot of a file explorer showing a project structure. The folders 'components' and 'containers' are highlighted with red rectangles. The structure includes folders for actions, components, constants, containers, reducers, and store, along with a file named index.js.