Understanding Components

Roland Guijt

MVP | CONSULTANT | TRAINER | AUTHOR

@rolandguijt rolandguijt.com

Module Overview

Component basics

Kinds of components

Props

State

Lifecycle methods

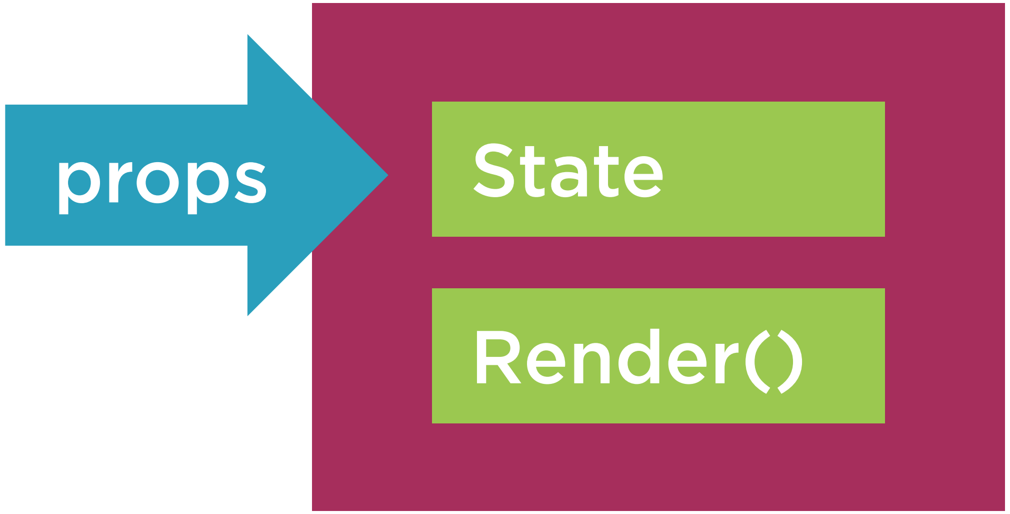
Building components

Components

Function

Class

Component



props

State

Render()

<Component prop="value"/>

State

Private data for the component

setState triggers re-render

Smart, async, optimized

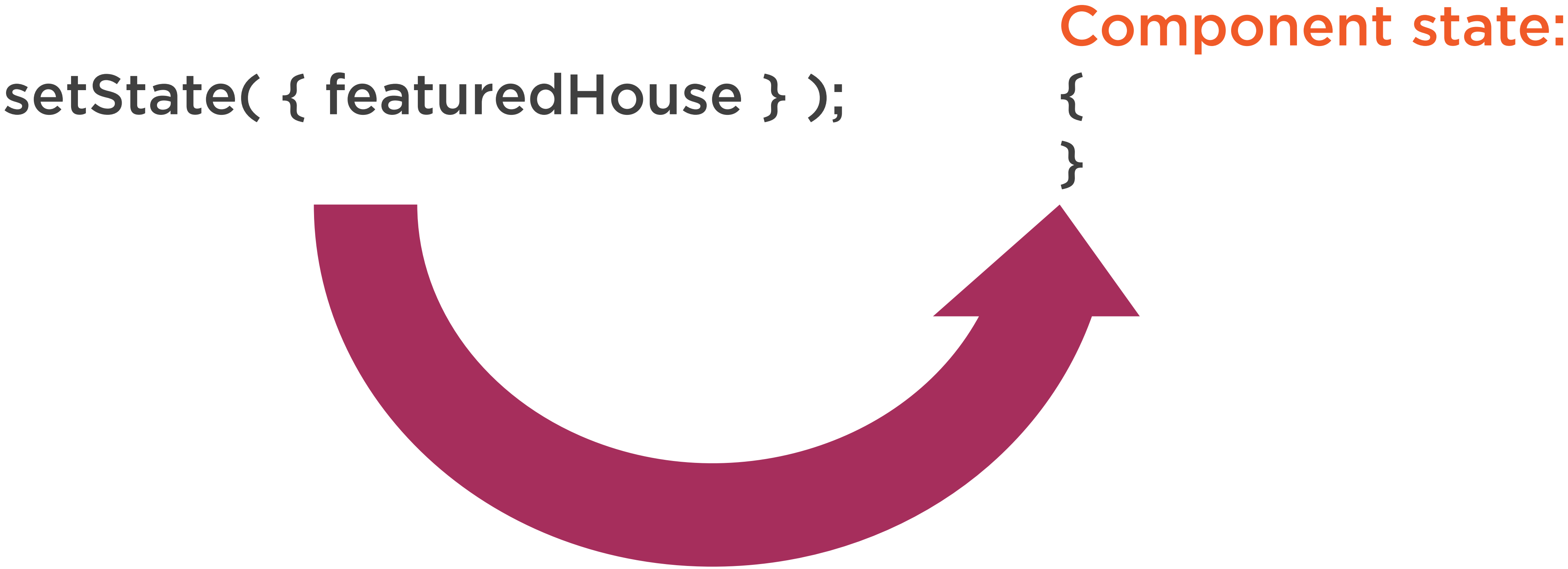
State is used in render()

Read by accessing state property

Written to by using setState

State updates are merged

State Updates



```
setState( { featuredHouse } );
```

Component state:

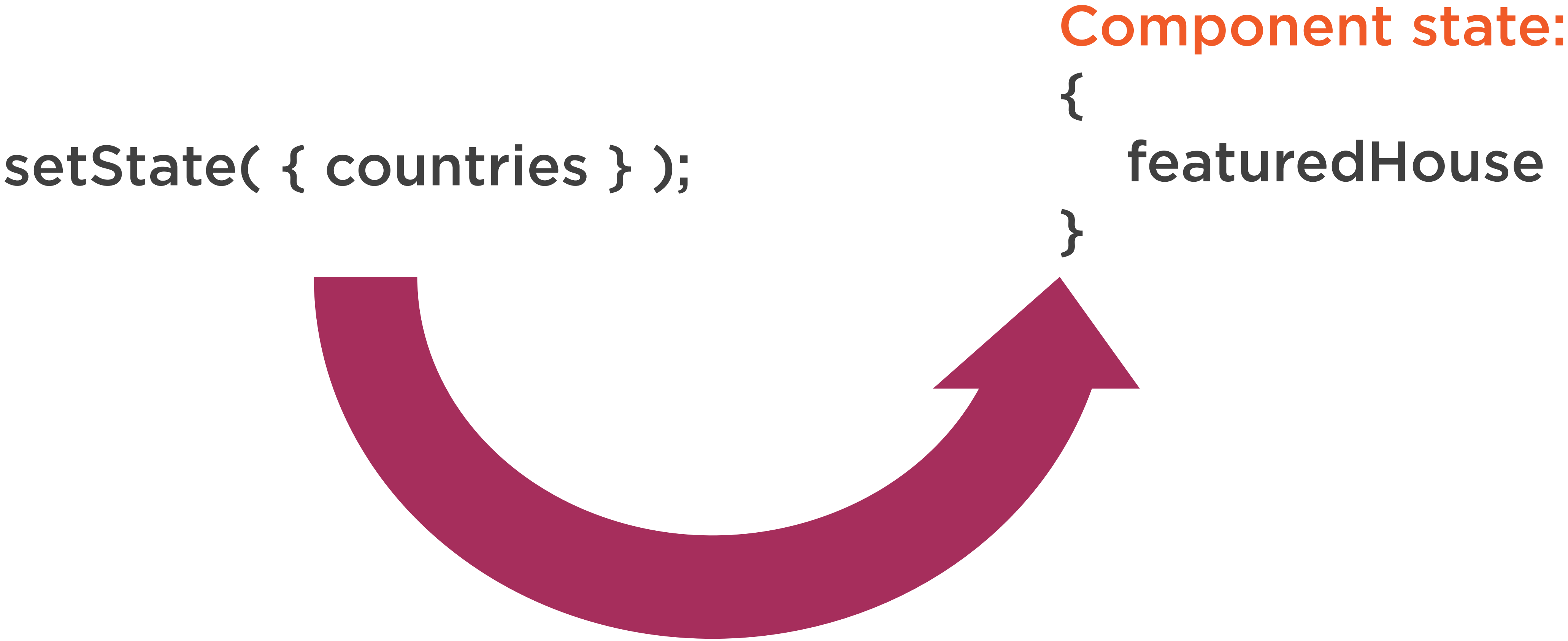
```
{
}
```

State Updates

Component state:

```
{
   featuredHouse
}
```

State Updates



```
setState( { countries } );
```

Component state:

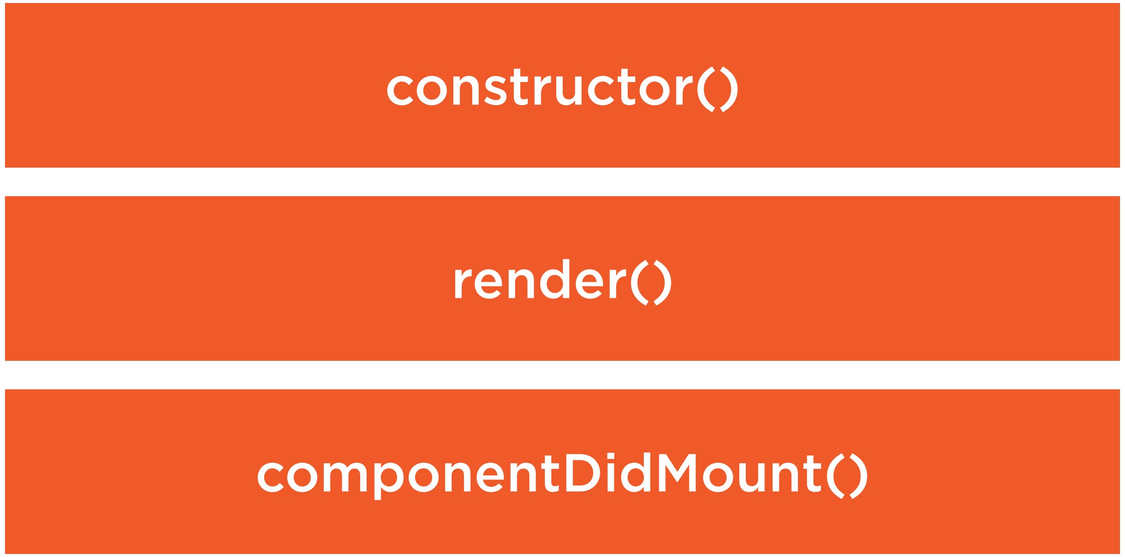
```
{
   featuredHouse
}
```

State Updates

Component state:

```
{
    featuredHouse,
    countries
}
```

Lifecycle Methods: Mounting



constructor()

render()

componentDidMount()

Lifecycle Methods: Updating

getDerivedStateFromProps()

shouldComponentUpdate()

render()

componentDidUpdate()

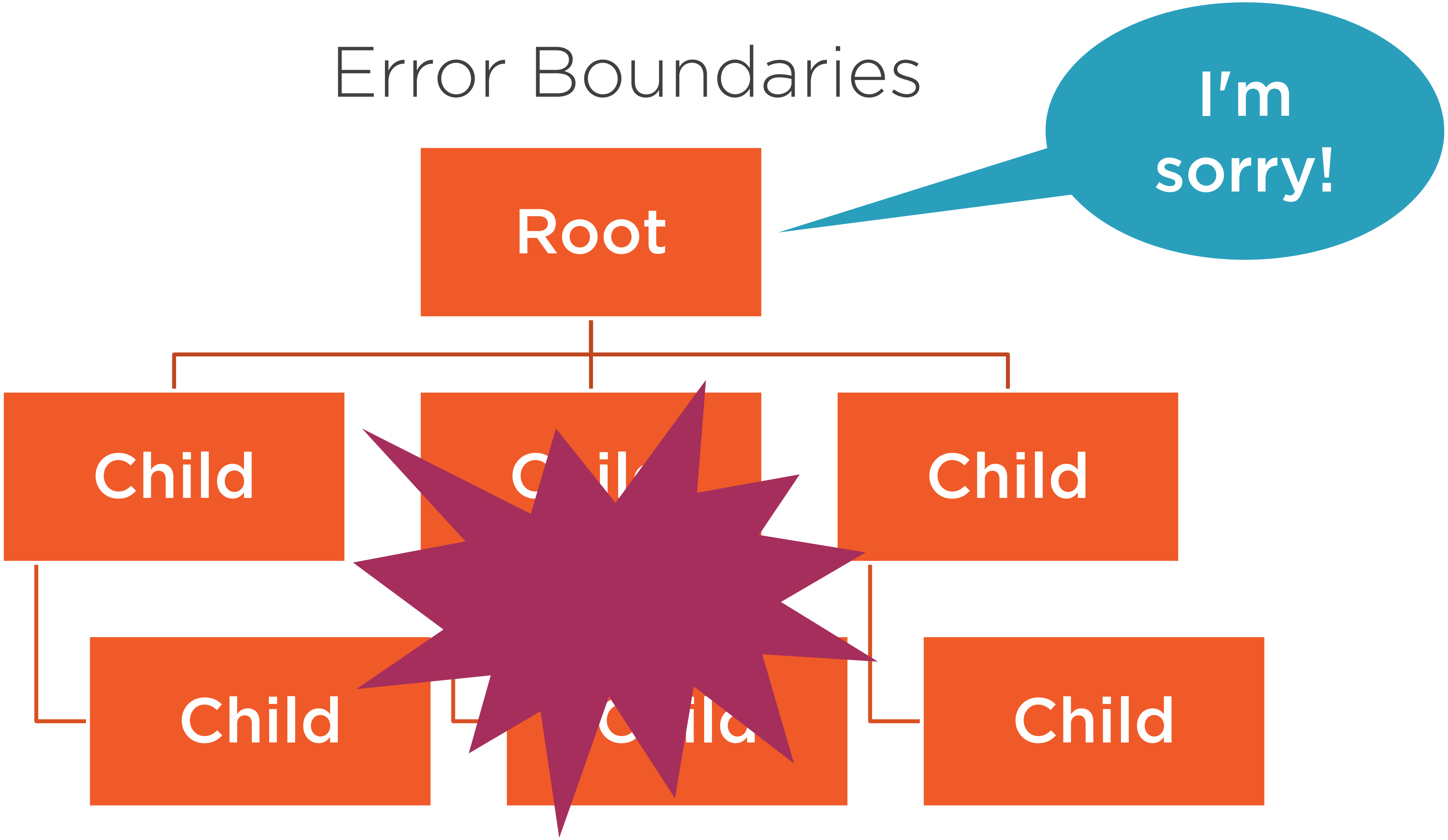
Lifecycle Methods: Unmounting

componentWillUnmount()

Lifecycle Methods: Error Boundary

componentDidCatch()

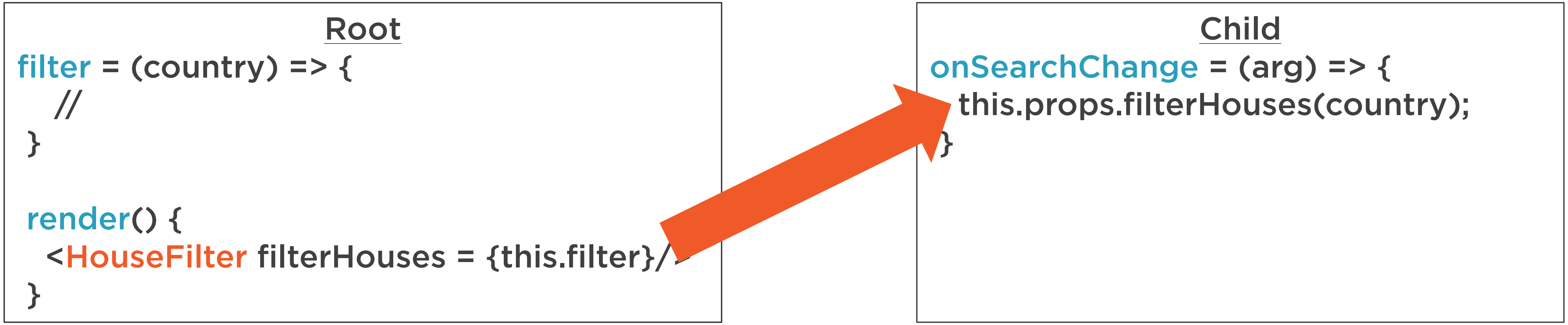
Error Boundaries



ComponentDidCatch

```
componentDidCatch(error, info) {
   this.setState({ hasError: true });
   log(error, info);
}

render() {
  if (this.state.hasError) {
      return <h1>Whoops! Sorry!</h1>;
  }
  return <div></div>; //normal UI
}
```

Functions as Props



Root

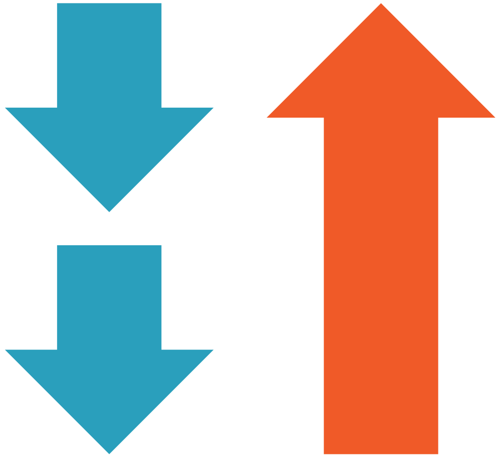
```
filter = (country) => {
   //
 }

 render() {
  <HouseFilter filterHouses = {this.filter}/>
 }
```

Child

```
onSearchChange = (arg) => {
  this.props.filterHouses(country);
}
```

Component Hierarchy

```
setActiveHouse = (house)
{}
```



App

SearchResults

SearchResultsRow

Summary

Component 101

Building components

Placing components