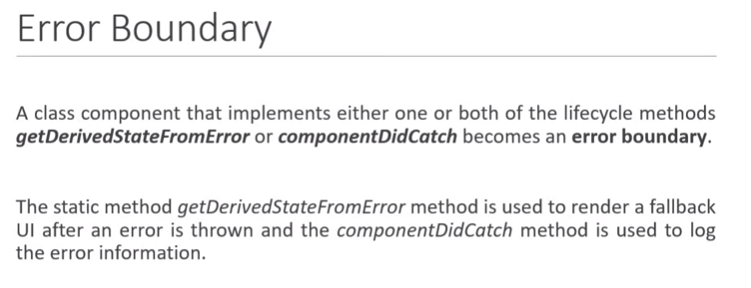
Error Handling Phase Methods:

Static getDerivedStateFromError(error)

componentDidCatch(error,info)

Error App:

Runtime errors during rendering could put our application in broken state, React basically unmounts the whole component tree. What would be great is if we can catch these errors anywhere in the tree and display a Fall-back UI for these errors.



Hero.js

import React from 'react'

function Hero({heroName}) {

if(heroName === 'Joker') {

throw new Error('Not a hero!')

}

return (

<div>

{heroName}

</div>

)

}

export default Hero

ErrorBoundary.js

import React, { Component } from 'react'

class ErrorBoundary extends Component {

constructor(props) {

super(props)

this.state = {

hasError:false

}

}

static getDerivedStateFromError(error){

return {

hasError:true

}

}

componentDidCatch(error,info){

console.log(error);

console.log(info);

}

render() {

if(this.state.hasError) {

return <h5>Something went wrong</h5>

}

else {

return this.props.children

}

}

}

export default ErrorBoundary

TestErrorBoundary.js

import React, { Component } from 'react'

import Hero from './Hero'

import ErrorBoundary from './Errors/ErrorBoundary'

class TestErrorBoundary extends Component {

render() {

return (

<div style={{textAlign:"center"}}>

{/\* <Hero heroName={'Batman'}/>

<Hero heroName={'Superman'}/>

<Hero heroName={'Joker'}/> \*/}

{/\* <ErrorBoundary>

<Hero heroName={'Batman'}/>

<Hero heroName={'Superman'}/>

<Hero heroName={'Joker'}/>

</ErrorBoundary> \*/}

<ErrorBoundary>

<Hero heroName={'Batman'}/>

</ErrorBoundary>

<ErrorBoundary>

<Hero heroName={'Superman'}/>

</ErrorBoundary>

<ErrorBoundary>

<Hero heroName={'Joker'}/>

</ErrorBoundary>

</div>

)

}

}

export default TestErrorBoundary