# REACT

### Props vs. State

La principal diferencia entre el **state** y las **props** es que las **props** son inmutables.

Esta es la razón por la cual el componente del contenedor debe definir el **state** que se puede actualizar y cambiar, mientras que los componentes secundarios solo deben pasar datos del estado usando **props**.

### Uso de props

#### App.js

#### index.js

### Default Props

#### App.js

#### index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App.jsx';
ReactDOM.render(<App/>, document.getElementById('app'));
```

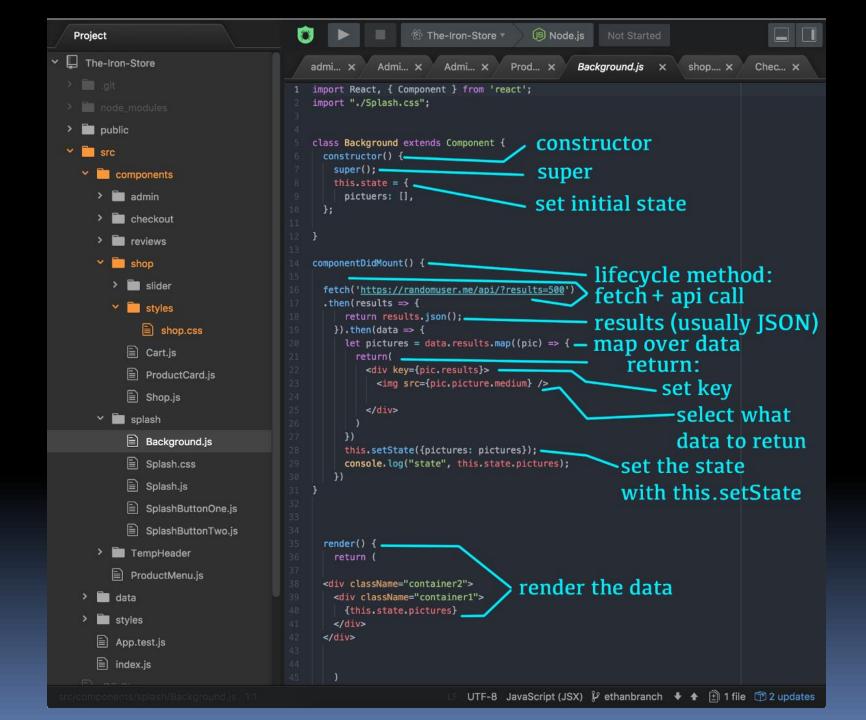
## Combinación (child)

```
import React from 'react';
class App extends React.Component {
 constructor(props) {
            super(props);
            this.state = { header: "Header from props...",
                            content: "Content from props..."
render() { return ( <div> <Header headerProp = {this.state.header}/>
<Content contentProp = {this.state.content}/> </div> ); } }
class Header extends React.Component {
            render() { return ( <div> <h1>{this.props.headerProp}</h1> </div> );
class Content extends React.Component {
            render() { return ( <div> <h2>{this.props.contentProp}</h2> </div> );
export default App;
```



### Métodos (parte 1)

- componentWillMount: es ejecutado antes del render, tanto en el cliente como en el servidor.
- componentDidMount: es ejecutado después del primer render solo del lado del cliente. (se utiliza para actualizar stαte.
- componentWillReceiveProps: se invoca tan pronto como se actualizan las props antes de que se llame a otro render.



### Peticiones fetch

El uso más simple de fetch() toma un argumento — la ruta del recurso que se quiera traer — y devuelve una promesa que contiene la respuesta (un objeto Response).

```
var miImagen = document.querySelector('img');
fetch('flores.jpg')
.then(function(response) {
  return response.blob(); })
.then(function(myBlob) {
  var objectURL = URL.createObjectURL(myBlob);
  miImagen.src = objectURL;
});
```