

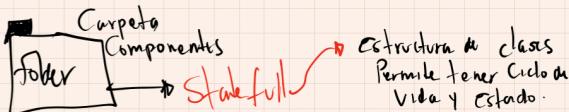
# REACT

Public → Están los archivos públicos q' se van a producción

Src → La carpeta mas importante

↳ app.js ← Componente visualizandose

Service Worker = Escucha si la aplicación Cambia o no



IDS componentes inician con mayuscula

↳ import React {Component} from 'react'

## Componente Presentacional

`import React from 'react';  
const Component = () => <h1>Hola Mundo!</h1>;  
export default Component;`

## Componente tipo clase

## Tipos de Componentes

Stateful y Stateless

```
class Stateful extends Component {  
  constructor(props){  
    super(props)  
    this.state = {  
      hello: 'Hola'  
    }  
  }  
  render(){  
    return(  
      <h1>Hola mundo</h1>  
      {this.state.hello}  
    )  
  }  
  export default stateful
```

Stateful  
Component

This.state ->  
hello: 'Hola'  
↳ objeto  
state

Componente Stateless → Sin Estado.  
no dependen de Estado ni Ciclo de Vida, solo presenta lógica

Código  
↳ import React from 'react'  
const stateless = () => {  
 return(  
 <h1>Hola mundo</h1>  
 )  
}

## Componentes

Wednesday, 30 September 2020 10:29 PM

Las etiquetas en los Componentes deben cerrarse  
<img src='url'> /> Si no se cierran las etiquetas sale un error.

0 J0

## folder Containers

Es el Container que va a recibir los Componentes

Componentes nuevos  
Componentes utilizados

```
import React from 'react'  
import Header from './components/Header'  
import Footer from './components/Footer'  
import App from './components/App'  
  
const App = () => {  
  <div>  
    <Header />  
    <Carousel />  
    <Footer />  
  </div>  
}  
  
export default App
```

## Index.js

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

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title>  
      </title>  
  </head>  
  <body>  
    <div id='app'>  
    </div>  
  </body>  
</html>
```

```
1 import React from 'react'  
2 import '../assets/styles/components/Categories.scss'  
3  
4 const Categories = ({children, title}) => (  
5   <div className="categories">  
6     <h3 className="categories__title">{title}</h3>  
7     {children}  
8   </div>  
9 )  
10  
11  
12 export default Categories
```

```
<Categories title='My List'>  
  <Carousel>  
    <CarouselItem/>  
    <CarouselItem/>  
    <CarouselItem/>  
    <CarouselItem/>  
  </Carousel>  
</Categories>
```

## folder Assets

Va a guardar todos los estilos

## Folder Componentes

Va a guardar los Estilos de Cada Componente

```
assets / styles  
  components  
    Header.scss  
    Search.scss  
    App.scss
```

Container Principal y se Cargan los Componentes.

Div principal de la aplicación

aquí es donde Se Carga la aplicación

el children me permite meter un Componente

Aquí Carga un Componente pero no lo hace por los propiedades.

El Componente Se abre y Se Cierra y Por eso permite un children

Este es el children de Categories y también tiene children CarouselItem

# Hooks

fue Presentado en Octubre de 2018  
En la react Conf.

Con react hook se Pueden hacer llamadas para mantener estados o Ciclo de vida desde un Componente Igeno. Sin transmitir informacion entre Componentes.

Ofrece estado y Ciclo de Vida a los Componentes de tipo función o Stateless.

Se Crearonx que la gente no Entendia las Clases y Dificultaba Usar los Constructores

Era mas Complicado transmitir las Propiedades de los elementos y se formaban Cascadas.

Const App = () => (

Return Explicito  
)

No Estavando las llaves.

useState → Para mantener el Estado

useEffect → Para hacer Peticiones o transmitir Eventos entre Componentes

○ Escuchar Cambios.

containers > App.jsx > ...

```
| import React, { useState, useEffect } from 'react';
```

useEffect recibe dos Parámetros. que se encarga de Escuchar cuando una propiedad Cambie y no se genere loop infinito.

## Custom Hooks

Thursday, 1 October 2020 11:38 PM

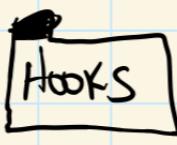
Puedo crear Custom Hooks para separar la lógica de los Componentes a una función que se puede usar en cualquier Componente

```
const [ videos, setVideos ] = useState([]);
useEffect(() => {
  fetch('http://localhost:3000/initialState')
    .then(response => response.json())
    .then(data => setVideos(data));
}, []);
```

Este es un hook

↳ voy a agregar a una carpeta

en



useInitialState.js

Importo los estados.  
La función va a manejar  
el Estado y Ciclo de Vida  
de los Componentes.

```
1 import {useState, useEffect} from 'react'
2
3 const useInitialState = () => {
4   const [ videos, setVideos ] = useState([])
5   useEffect(() => {
6     fetch('http://localhost:3000/initialState')
7       .then(response => response.json())
8       .then(data => setVideos(data))
9   }, [])
10  return videos
11}
12
13 export default useInitialState
```

# JSX - validaciones

Thursday, 1 October 2020 10:48 PM

## Validaciones JSX Condicionales.

### Validación.

```
{videos.mylist.length > 0 &&  
  <Categories title='My List'>  
    <Carousel>  
      <CarouselItem/>  
    </Carousel>  
  </Categories>  
}
```

Condición si no hay videos  
El componente no se muestra!

```
const App = () => {  
  const [ videos, setVideos ] = useState([])  
  useEffect(() => {  
    fetch('http://localhost:3000/initialState')  
      .then(response => response.json())  
      .then(data => setVideos(data))  
  }, [])
```

aquí se está alimentando  
la variable videos.  
Con setVideos

## Iterar en un Componente

```
<Categories title='Favorites'>  
  <Carousel>  
    {videos.trends.map(item =>  
      <CarouselItem key={item.id} {...item} />  
    )}  
  </Carousel>  
</Categories>
```

Con map heredan el  
Componente y se van  
a generar tantos Carousel  
Como hayan en la API.

```
{videos.trends?.map(item =>
```



Condicionar que trends exista  
If (videos.trends)? map.

```
<Categories title='Tendencias'>  
  <Carousel>  
    {videos.trends === undefined &&  
     videos.trends.map(item =>  
       <CarouselItem key=  
       {item.id} {...item}></CarouselItem>  
     ))}
```

También se puede condicionar  
en App.jsx

## PropTypes

Friday, 2 October 2020 12:08 AM

PropTypes ofrece una manera dinámica de verificar las propiedades que se le pasan a los componentes. Para verificar el tipo de dato.

↳ `npm install prop-types`

→ Instalación

Se importan

```
import React from 'react'
```

```
import PropTypes from 'prop-types'
```

```
CarouselItem.propTypes = {
  cover: PropTypes.string,
  title: PropTypes.string,
  year: PropTypes.number,
  contentRating: PropTypes.string,
  duration: PropTypes.number
}
```

→ Se definen como se reciben los tipos de datos de las propiedades del componente.

```
Component.propTypes = {
  name: PropTypes.string.isRequired, // obligatorio
  lastName: PropTypes.string.isRequired, // obligatorio
  age: PropTypes.number, // opcional,
  list: PropTypes.array, // opcional
};
```