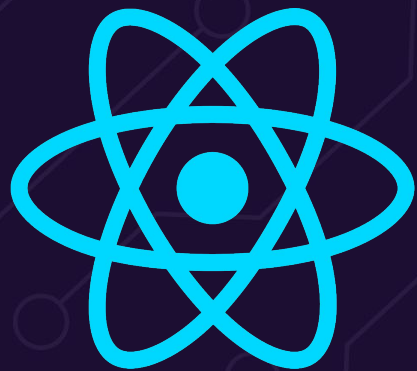


# Introducción a



# React

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

# ¿Qué es React?

¿Cuál es su origen?

React fue liberado en Github en el año 2013.

Dos años antes inició indirectamente su desarrollo cuando Facebook estaba trabajando en una tecnología llamada XHP una especie de evolución de PHP que permitía protegerse mejor de los ataques cross site scripting.



# ¿Qué es React?

¿Cuál es su origen?

Esta tecnología permitía crear elementos html reusables lo que ahora llamamos como **componentes**

¿Cuál fué el problema?

**XHP** no daba respuesta a las necesidades de las **SPA(Single Page Application)** ya que se renderizaba completamente en el servidor y fue entonces cuando decidieron convertir el proyecto en una librería que se pudiera ejecutar del lado del cliente

# ¿Qué es React

¿En qué se basa React?

## Declarativo

Nuestro código siempre refleja que debe renderizar nuestra interfaz pero no programamos el cómo y cuándo se debe renderizar. De eso se encarga **React**, esto hará que nuestro código sea más predecible y fácil de controlar gracias a él **state** de cada componente

# ¿Qué es React?

¿En qué se basa React?

## Basado en Componentes

- Nuestra interfaz estará dividida en partes más pequeñas
- Puedes crear componentes nuevos con componentes más pequeños
- Cada componente encapsula su estado
- Código más reusable pequeño y mantenible

# ¿Qué es React?

¿En qué se basa React?

## Componentes

Contenedor App

Caja de Búsqueda

Contenedor de Productos

Título de Categoría Productos

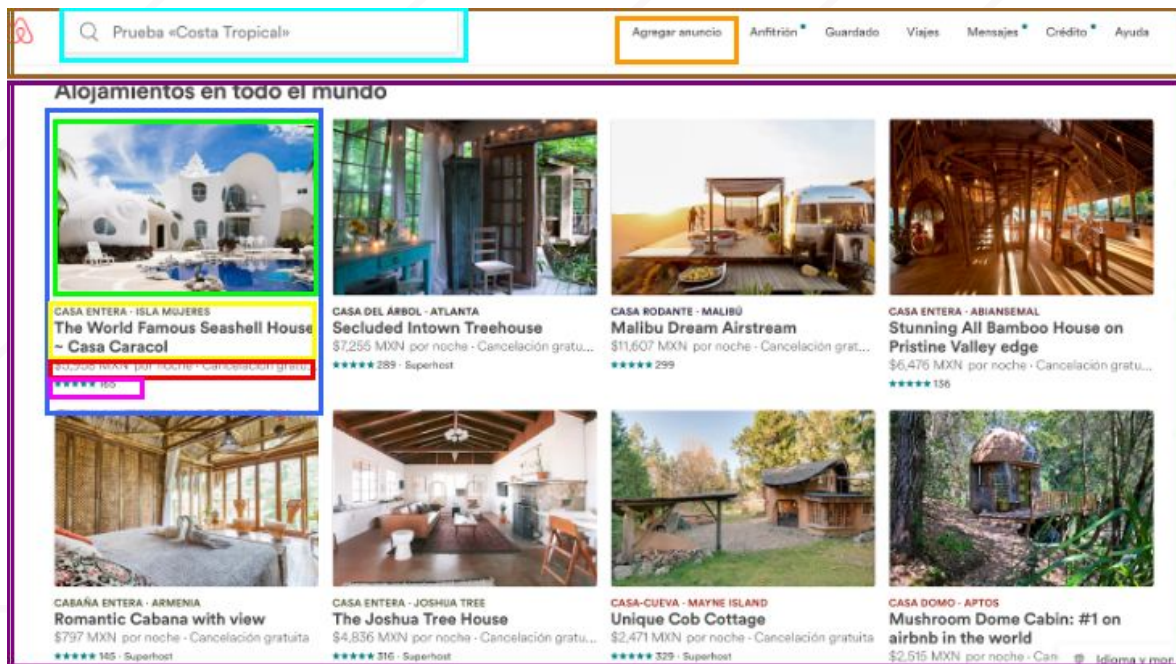
Producto

☐ 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

# ¿Qué es React

Piensa en componentes



# ¿Qué es React?

¿En qué se basa React?

Ser declarativo y orientado a componentes da lugar a otra de las grandes ventajas de React y es de hecho la que le otorga el nombre.

## PROGRAMACIÓN **REACTIVA**

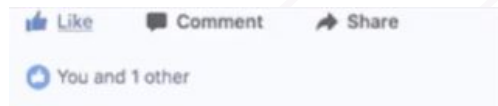
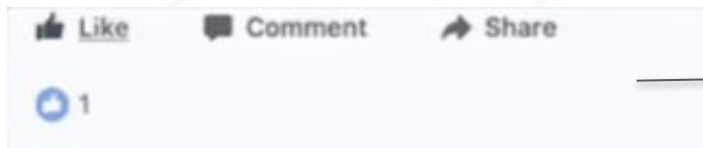


# ¿Qué es React?

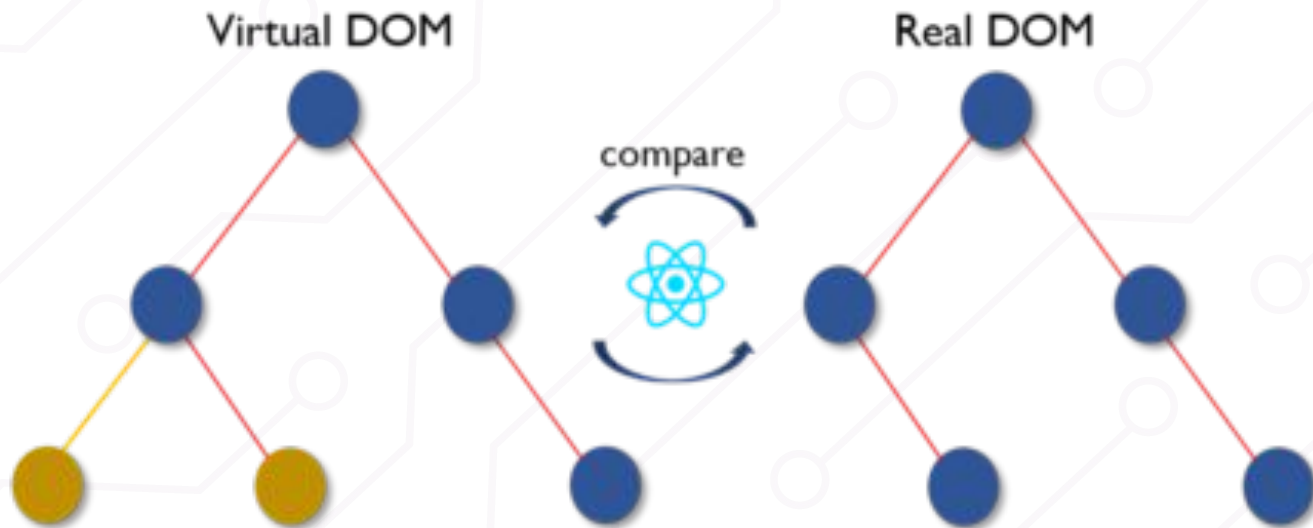
¿En qué se basa React?

## PROGRAMACIÓN REACTIVA

Cambios de estado y de propiedades de nuestros componentes generan un nuevo Renderizado en donde se realiza el cambio.



# VIRTUAL DOM



# ¿Qué es React?

**JSX** es una extensión de JavaScript creada por Facebook para el uso con su librería React. Sirve de preprocesador (como Sass o Stylus a CSS) y transforma el código a JavaScript.

```
const element = <h1>Hello, world!</h1>;
```

# ¿Qué es React

JSX under the hood

## JSX bajo el capó

```
// JSX
<div prop="miProp">
  <h1>Soy un Header!</h1>
</div>
```

Babel transformará el código a lo siguiente:

```
// JS
React.createElement( "div", // tag o componente
  { prop: "miProp" }, // propiedades
  React.createElement( // elemento hijo
    "h1", null, // null cuando el elemento no tiene propiedades
    "Soy un Header!"
  )
);
```

# ¿Qué es React

Self-Closing Tags


Cabe recalcar, que también podemos expresar componentes con self-closing tags, si estos no tienen componentes hijos:

```
<ComponenteSinHijos className="barra-lateral" />
```

```
<ComponenteSinHijos className="barra-lateral" />
```

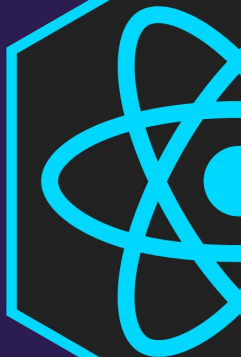
A close-up photograph of a human hand, palm up, holding a single, translucent red pill. The background is dark and out of focus.

Functional  
Components

A close-up photograph of a human hand, palm up, holding a single, translucent green pill. The background is dark and out of focus.

Class-Based  
Components

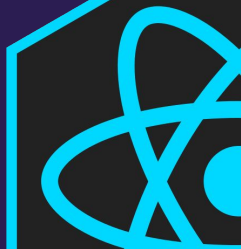
# Functional Component



También conocidos como  
componentes de Presentación,  
componentes tontos/dumb , o  
stateless components

```
const SoyUnComponenteFuncional = () => {  
  return <div>Tonto dice...</div>  
}
```

# Class-Based Component



También se les llama contenedores  
componentes tontos/smart o  
stateful components



```
class App extends React.Component{  
  
  render(){  
    return (  
      <div>  
        Tengo clase  
      </div>  
    )  
  }  
}
```

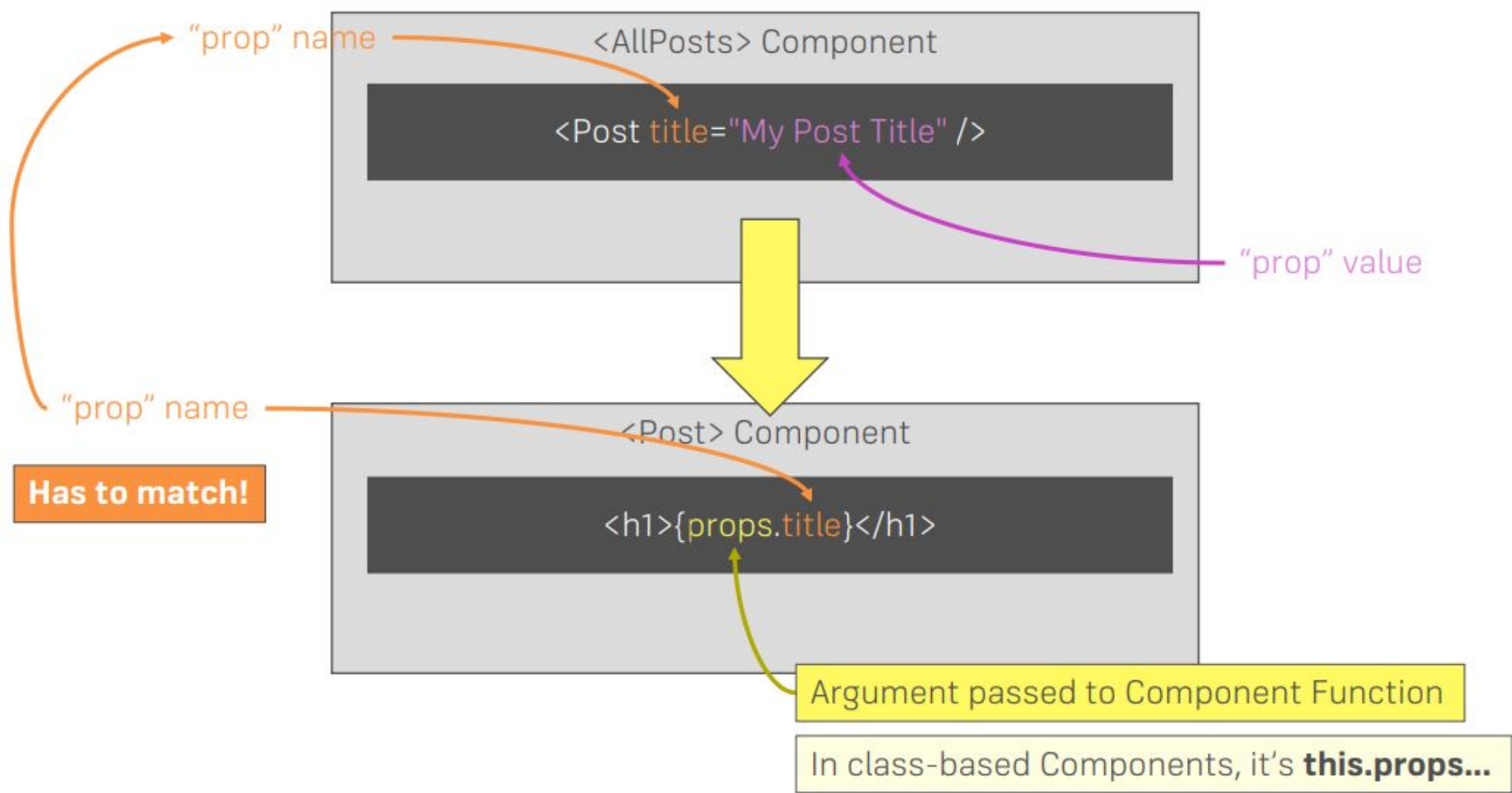


## Functional

- 1 Functional programming style
- 2 Minimal boilerplate  
Clean and simple

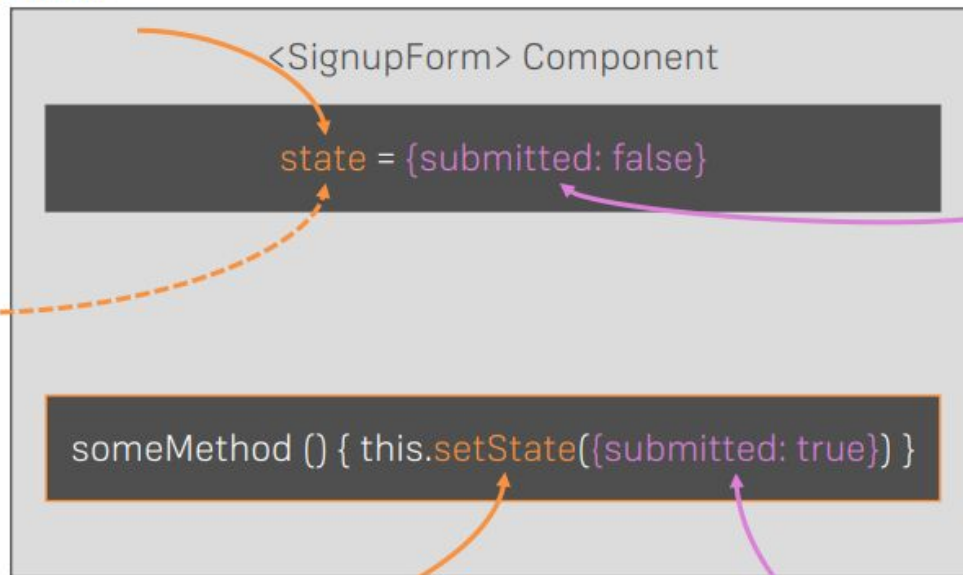
## Class-based

- 1 Object-oriented programming style
- 2 Can have state
- 3 Can have lifecycle methods  
perform actions when the component is mounted, unmounted, about to be updated, etc.
- 4 Can have refs  
Reference and manipulate underlying DOM elements
- 5 Performance optimisation  
With `shouldComponentUpdate` and `PureComponent`  
**Use with caution!**



## Changes from WITHIN a Component

"state" is a reserved property name  
(and can only be set in  
class-based components!)



Any data of your choice!

Mutate state &  
trigger re-render

Gets merged with  
original state

# State



Data in the State control what you see in the View

```
const data = [  
  {  
    "name": "AFC Bournemouth",  
    "logo": "",  
    "manager": "Eddie Howe",  
    "stadium": "Dean Court",  
    "capacity": 11360  
  }  
]
```

## EPL Teams

1. AFC Bournemouth
2. Arsenal
3. Brighton & Hove Albion
4. Burnley
5. Chelsea
6. Crystal Palace
7. Everton

# create- React-app name-of-app

Instalando create-react-app con npm

Para crear una aplicación en React se

```
npm install -g create-react-app  
create-react-app my-app
```

```
cd my-app  
npm start
```

feature

Buscar

issue 1

issue 2

issue 3

issue n