

Tema 3 - React a fondo: composición vs herencia

Composición

- Nos permite definir una UI compleja a partir de componentes sencillos
- Un componente que incluye otro(s) en su método **render** se convierte en el padre y "dueño" del ciclo de vida de éstos
- Podemos "configurar" o "cablear" los componentes hijo mediante **props**

Comunicación hijo -> padre

- ¿Y si queremos comunicarnos de hijo de padre?
- Hacemos que el componente tenga una API definida, usando funciones
- El padre pasará al hijo funciones como props
- El componente hijo llamará a esos “callbacks” con datos específicos de su dominio

Comunicación hijo -> padre

```
class Buttons extends Component {  
  render() {  
    return (  
      <div className="actions">  
        <button onClick={ this.props.onStop }>STOP</button>  
        <button onClick={ this.props.onStart }>START</button>  
      </div>  
    );  
  }  
}
```

```
Buttons.propTypes = {  
  onStop: React.PropTypes.func,  
  onStart: React.PropTypes.func  
}
```

Comunicación hijo -> padre

```
class Parent extends Component {
  constructor() {
    super();
    this.handleStart = this.handleStart.bind(this);
    this.handleStop = this.handleStop.bind(this);
  }
  handleStart(e) {
    console.log("Start en Buttons!");
  }
  handleStop(e) {
    console.log("Stop en Buttons!");
  }
  render() {
    return (
      <div className="parent">
        <Buttons onStart={this.handleStart} onStop={this.handleStop} />
      </div>
    );
  }
}
```

Ejercicio: cronómetro

- Ya podemos crear por fin nuestro primer componente interactivo usando props, estado interno y eventos
- Vamos a implementar un cronómetro como éste: el botón START inicia el temporizador, y el botón STOP lo detiene en el primer clic y lo reinicia a 0 en el segundo.



Ejercicio: cronómetro

- Disponéis de la plantilla para este ejercicio en `/ejercicios/tema3/src/templates/cronometro.html`
- Estilos en `/ejercicios/tema3/dist/index.css`: añadir la hoja de estilos a `index.html`.
- Tenéis funciones auxiliares para manipular el tiempo con fechas en `/ejercicios/tema3/src/lib/utils.js`

Cronómetro: primeros pasos

- Crear un componente Cronometro
- Pegar en su render la maqueta/plantilla entera, y cargar éste componente en el DOM
- Una vez que funcione correctamente, separar en otros componentes
- Al final el Cronometro será simplemente un contenedor de otros componentes, y albergará el estado
- Implementar interactividad en Cronometro y definir la interfaz padre <-> hijos

Cronómetro: pistas

- Utilizar **composición**: el cronómetro completo debe contener un componente Header, un componente Screen y un componente Buttons.
- Se pueden pasar funciones como **props** de modo que un evento sea “atendido” por el componente padre de quien lo registra y recibe.
- Intentar basar el paso de datos padre-hijo en **props**
- No almacenar información **derivada** en el estado (que pueda ser calculada a partir de props o estado)

Refs

- React gestiona el DOM por nosotros, nosotros generamos VirtualDOM y la librería hace el diff automáticamente
- Si necesitamos acceder a un nodo montado en el DOM, tenemos que marcarlo en JSX con una referencia:

```
<button ref="miboton">Click me</button>
```

- Después podemos obtener la referencia en código con **this.refs.miboton**. Si React elimina o sustituye ese nodo, actualizará la referencia para nosotros (o será *undefined*)
- En **this.refs.miboton** tenemos el nodo del DOM real

Refs

- Si ponemos una **ref** a un componente React (no HTML simple), la ref apuntará a la instancia de ese componente.

```
<MyComponent ref="comp" />
```

- Después podremos acceder desde fuera a los métodos públicos de la clase, su estado, etc.

```
this.refs.comp.doSomething()  
this.refs.comp.state
```

Formularios

- Los controles de formulario HTML son problemáticos para React
- Son inherentemente mutables mediante interacciones de usuario (comportamiento definido por el navegador)

Formularios

```
class TextInput extends Component {  
  render() {  
    return (  
      <input type="text" value="Introduce tu nombre" />  
    );  
  }  
}
```

Si intentamos escribir en esa caja de texto, no pasará nada
¿Por qué?

Formularios

```
class TextInput extends Component {  
  render() {  
    return (  
      <input type="text" value="Introduce tu nombre" />  
    );  
  }  
}
```

Porque **render** dice que, invariablemente, el valor de ese INPUT es “Introduce tu nombre”

Formularios

```
class TextInput extends Component {  
  render() {  
    return (  
      <input type="text" value="Introduce tu nombre" />  
    );  
  }  
}
```

Si fuera HTML y no React, podríamos borrar ese texto y escribir otro...

Formularios

- Hay dos formas de trabajar con formularios en React: con componentes **controlados** o **no controlados**
- Cuando ponemos **value="xxx"** o **value={xxx}** lo convertimos en **controlado**: su valor lo controla React
- Si no ponemos value, o ponemos **defaultValue** (valor inicial), funcionará como un INPUT corriente, estará **no controlado**

Ejemplo no controlado

```
class LoginForm extends Component {
  constructor() {
    super()
    this.handleSubmit = this.handleSubmit.bind(this);
  }
  handleSubmit(e) {
    let username, password;
    e.preventDefault();
    username = this.refs.usuario.value;
    password = this.refs.pwd.value;
    console.log('Iniciando sesión para', username, password);
  }
  render() {
    return (
      <div>
        <form onSubmit={ this.handleSubmit }>
          <p>
            Usuario: <br />
            <input type='text' ref='usuario' />
          </p>
          <p>
            Password:<br />
            <input type='password' name='pwd' ref='pwd' />
          </p>
          <p><button type='submit'>Iniciar sesión</button></p>
        </form>
      </div>
    )
  }
}
```

Ejemplo no controlado

```
class LoginForm extends Component {
  constructor() {
    super()
    this.handleSubmit = this.handleSubmit.bind(this);
  }
  handleSubmit(e) {
    let username, password;
    e.preventDefault();
    username = this.refs.usuario.value;
    password = this.refs.pwd.value;
    console.log('Iniciando sesión para', username, password);
  }
  render() {
    return (
      <div>
        <form onSubmit={ this.handleSubmit }>
          <p>
            Usuario: <br />
            <input type='text' ref='usuario' />
          </p>
          <p>
            Password:<br />
            <input type='password' name='pwd' ref='pwd' />
          </p>
          <p><button type='submit'>Iniciar sesión</button></p>
        </form>
      </div>
    )
  }
}
```

Formularios

- ¿Y si queremos control total con **value**?
- Es un “choque” conceptual con el Virtual DOM de React, que gestiona por nosotros todo el HTML producido
- Tenemos props **específicas** para controles de formularios
- Y un evento muy útil: **onChange**

Formularios

- **value** - establece el valor en:
 - `<input type="text" .../>`
 - `<input type="password" .. />`
 - `<textarea .. />`
 - `<select />` (valor del elemento seleccionado!!)

Formularios

- **checked** - (Boolean) recupera/establece si están activos:
 - `<input type="checkbox" .../>`
 - `<input type="radio" .. />`

Formularios

- **selected** - (Boolean) recupera/establece si están seleccionados los elementos **option** de un desplegable:
- `<select>`
 - `<option value="1">Uno</option>`
 - `<option value="2">Dos</option>`
 - `</select>`

Formularios: componentes controlados

- La salida del método **render** define el estado de la UI en cualquier momento determinado
- Si escribimos


```
<textarea value="Introduce tu comentario"></textarea>
```
- El usuario **no puede modificar** el contenido. Está “hard-coded” en el código Javascript generado a partir de JSX

Formularios: componentes controlados

- Una solución es utilizar el estado interno del componente como fuente para el control del formulario
- Implica que tenemos que modificar “manualmente” el estado cada vez que el usuario modifique el control
- **onChange** funciona en todos los controles, afortunadamente

Formularios: componentes controlados

```
class LoginFormControlled extends Component {
  constructor() {
    super()
    this.state = {
      usuario: '',
      password: ''
    }
    this.handleSubmit = this.handleSubmit.bind(this);
    this.handleChange = this.handleChange.bind(this);
  }
  handleSubmit(e) {
    e.preventDefault();
    const { usuario, password } = this.state;
    console.log('Iniciando sesión para', usuario, password);
  }
  handleChange(e) {
    this.setState({
      usuario: e.target.value
    });
  }
  render() {
    const { usuario, password } = this.state;
    return (
      <div>
        <form onSubmit={ this.handleSubmit }>
          <p>
            Usuario: <br />
            <input type='text' id='usuario' value={ usuario } onChange={ this.handleChange } />
          </p>
          <p>
            Password:<br />
            <input type='password' id='password' value={ password } />
          </p>
          <p><button type='submit'>Iniciar sesión</button></p>
        </form>
      </div>
    )
  }
}
```



Formularios: componentes controlados

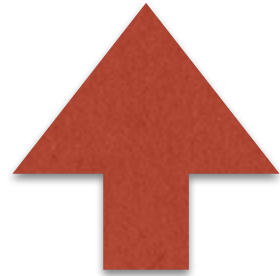
- ¿Y hay que hacer un método **handleXXX** por cada control del formulario?
- Depende, podemos aprovechar Babel y la magia de ES6 para reescribir **handleChange** del ejemplo anterior así:

Formularios: componentes controlados

```
handleChange(e) {  
  this.setState({  
    [e.target.id]: e.target.value  
  });  
}
```

Formularios: componentes controlados

```
handleChange (e) {  
  this.setState ({  
    [e.target.id]: e.target.value  
  });  
}
```



"Computed property names"

Thug Life



Ejercicio: formularios

- Buscador de personajes de Juego de Tronos

Buscador Juego de Tronos

Actor / personaje

Familia

Todas

Sólo personajes vivos☐

Aparece en temporada

1☐ 2☐ 3☐ 4☐ 5☐

Personaje	Actor	Nº Ep	Vivo
Eddard Stark	Calvin Hobbs	45	Sí
Eddard Stark	Calvin Hobbs	45	Sí
Eddard Stark	Calvin Hobbs	45	Sí

Encontrados 25 personajes

Ejercicio: formularios

- Queremos un buscador que actualice los resultados en vivo, según se modifican los parámetros de búsqueda (al estilo **onChange**)
- Los datos en JSON:
/ejercicios/tema3/src/data/got.js
- Esqueleto en el repositorio:
/ejercicios/tema3/src/components/buscador/

Composición

- Una aplicación entera de React se pinta **a partir de un componente raíz**, que a su vez incluye componentes hijos y así sucesivamente
- El componente que incluye otro en su método render es el **dueño** de ese nodo hijo
- El padre puede pasarle props al hijo, configurándolo, y será el responsable del ciclo de vida del hijo
- Cuando no aparezca en su **render**, React desmontará el componente por nosotros

Composición

- La “manera React” es intentar hacer componentes específicos con el mínimo estado posible
- Recuerda: $UI = f(\text{datos})$
- Es decir: **render** depende sólo de los **props** y **state** actuales del componente
- Separación de Responsabilidades a nivel de UI
- Cada componente hace una cosa

Composición

- De esta forma los componentes son cajas negras que “cableamos” mediante sus props.
- Le damos datos via props
- Atendemos sus *notificaciones* pasando una función vía props (ej: onChange en el buscador)

Composición

centralLog - Events							
<div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> </div> <div> <div>Go to</div> <div>MODE LIVE</div> </div>							
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
ⓘ	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
⚠	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
⚠	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
ⓘ	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
⚠	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
ⓘ	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
ⓘ	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
⚠	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
ⓘ	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
⚠	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
⚠	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

Composición

<body>

Root

EventsLayout

FilterBox

Toolbar

Menu

EventList

EventListHeader

Event

Event

Event

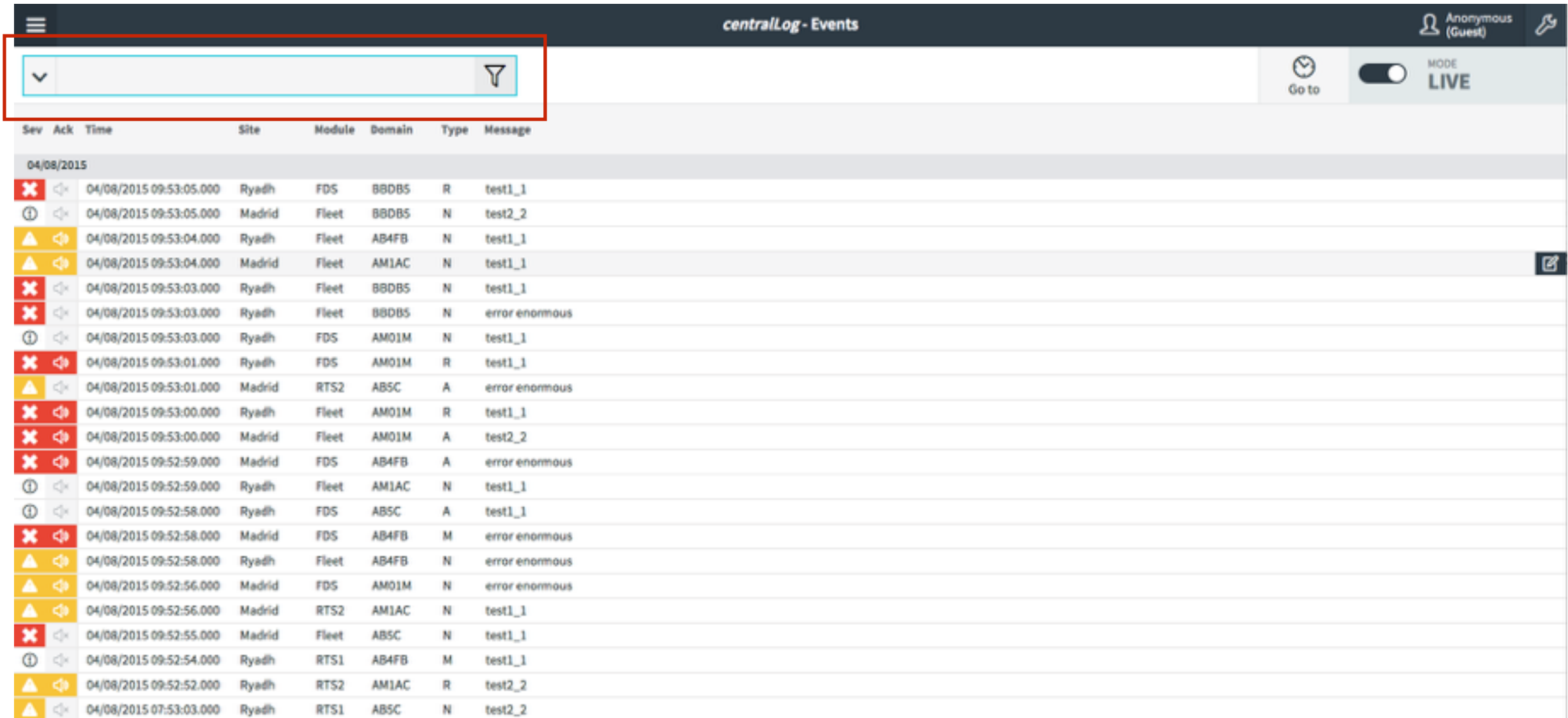
</body>

Composición

centralLog - Events							
Anonymous (Guest)							
Go to		MODE LIVE					
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<EventsLayout />

Composición



Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<FilterBox />

Composición

centralLog - Events							
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
ⓘ	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
⚠	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
⚠	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
ⓘ	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
⚠	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
ⓘ	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
ⓘ	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
⚠	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
ⓘ	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
⚠	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
⚠	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<Toolbar />

Composición

centralLog - Events							
Anonymous (Guest)		LIVE					
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<Menu visible={false} />

Composición

centralLog - Events							
<div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> </div> <div>Go to</div> <div> <div></div> <div>MODE LIVE</div> </div>							
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
ⓘ	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
⚠	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
⚠	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
ⓘ	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
⚠	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
ⓘ	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
ⓘ	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
⚠	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
ⓘ	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
⚠	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
⚠	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<EventList />

Composición

centralLog - Events							
Sev Ack Time Site Module Domain Type Message							
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<EventListHeader />

Composición

centralLog - Events							
Anonymous (Guest)							
Go to		MODE LIVE					
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<Event severity="error" state="acked" />

Composición

centralLog - Events							
Anonymous (Guest)							
Go to		MODE LIVE					
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
🔊	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
🔊	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
🔊	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
🔊	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
🔊	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
🔊	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<Event severity="info" state="acked" />

Composición

centralLog - Events							
Anonymous (Guest)							
Go to <input type="checkbox"/> MODE LIVE							
Sev	Ack	Time	Site	Module	Domain	Type	Message
04/08/2015							
✖	🔊	04/08/2015 09:53:05.000	Ryadh	FDS	BBDB5	R	test1_1
🔊	🔊	04/08/2015 09:53:05.000	Madrid	Fleet	BBDB5	N	test2_2
⚠	🔊	04/08/2015 09:53:04.000	Ryadh	Fleet	AB4FB	N	test1_1
⚠	🔊	04/08/2015 09:53:04.000	Madrid	Fleet	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	test1_1
✖	🔊	04/08/2015 09:53:03.000	Ryadh	Fleet	BBDB5	N	error enormous
🔊	🔊	04/08/2015 09:53:03.000	Ryadh	FDS	AM01M	N	test1_1
✖	🔊	04/08/2015 09:53:01.000	Ryadh	FDS	AM01M	R	test1_1
⚠	🔊	04/08/2015 09:53:01.000	Madrid	RTS2	AB5C	A	error enormous
✖	🔊	04/08/2015 09:53:00.000	Ryadh	Fleet	AM01M	R	test1_1
✖	🔊	04/08/2015 09:53:00.000	Madrid	Fleet	AM01M	A	test2_2
✖	🔊	04/08/2015 09:52:59.000	Madrid	FDS	AB4FB	A	error enormous
🔊	🔊	04/08/2015 09:52:59.000	Ryadh	Fleet	AM1AC	N	test1_1
🔊	🔊	04/08/2015 09:52:58.000	Ryadh	FDS	AB5C	A	test1_1
✖	🔊	04/08/2015 09:52:58.000	Madrid	FDS	AB4FB	M	error enormous
⚠	🔊	04/08/2015 09:52:58.000	Ryadh	Fleet	AB4FB	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	FDS	AM01M	N	error enormous
⚠	🔊	04/08/2015 09:52:56.000	Madrid	RTS2	AM1AC	N	test1_1
✖	🔊	04/08/2015 09:52:55.000	Madrid	Fleet	AB5C	N	test1_1
🔊	🔊	04/08/2015 09:52:54.000	Ryadh	RTS1	AB4FB	M	test1_1
⚠	🔊	04/08/2015 09:52:52.000	Ryadh	RTS2	AM1AC	R	test2_2
⚠	🔊	04/08/2015 07:53:03.000	Ryadh	RTS1	AB5C	N	test2_2

<Event severity="warning" state="active" />

Composición: contenedores y componentes

- Distinguimos dos tipos de componentes React
- **Componentes** / presentacionales / Dumb components
- **Contenedores** / Controladores / Smart components

Composición: componentes presentacionales

- Son abstracciones sobre HTML
- Muy poca o ninguna lógica propia, dependen de **props** para casi todo
- Muy reutilizables, muy configurables
- Ejemplos:
 - Screen y Buttons del cronómetro
 - Lista personajes en el buscador de Juego de Tronos

Composición: contenedores



- Prácticamente no tienen HTML en su **render**, sólo la composición de otros componentes presentacionales
- Se encargan de gestionar los datos en su estado interno, y la orquestación entre hijos, así como el "cableado" de props, etc.
- ¿Ejemplos hasta ahora?

Ciclo de vida de un componente

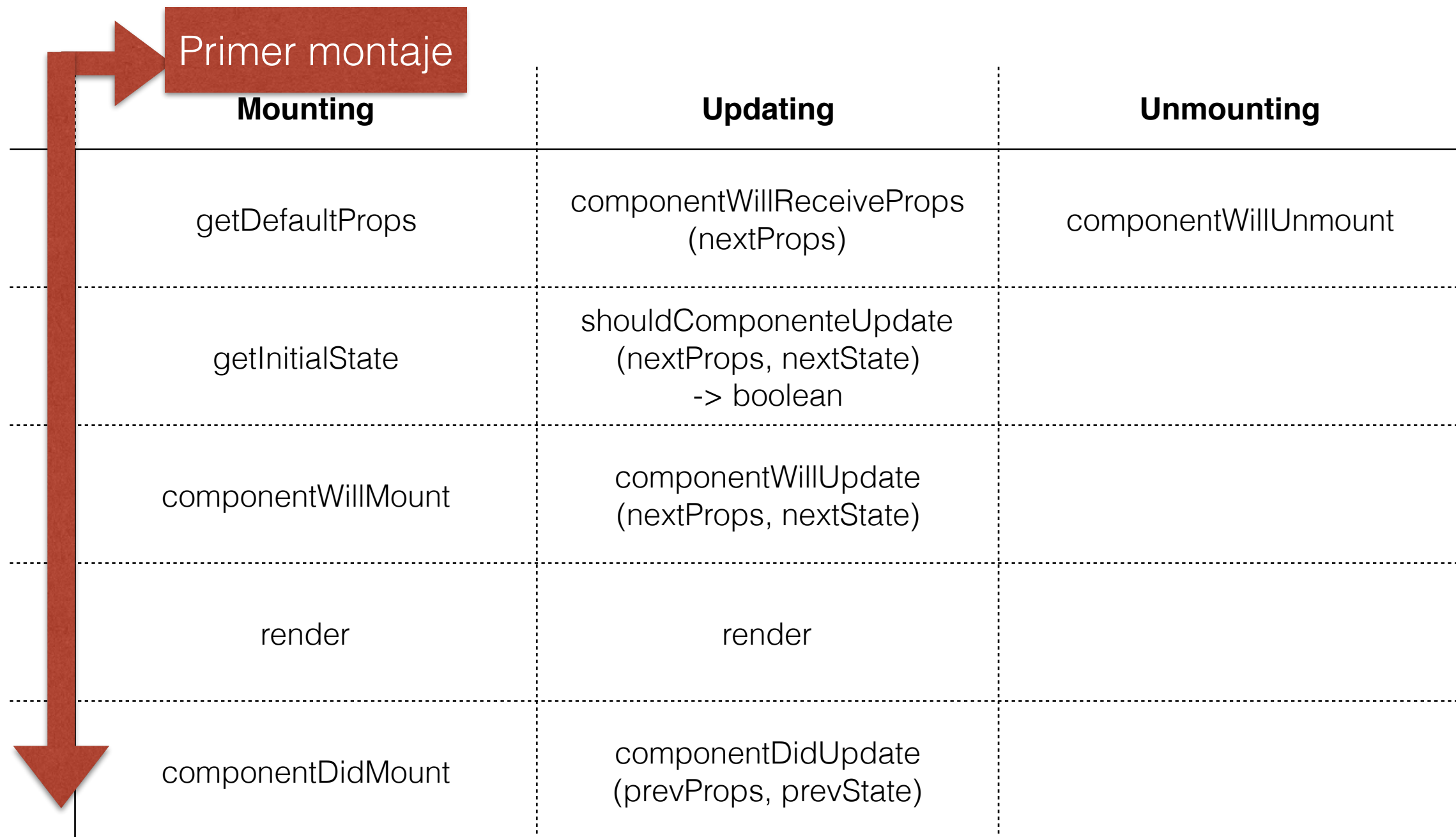
- Un componente React tiene una serie de métodos que React llama en un orden predefinido
- Ya hemos utilizado **getInitialState** y **getDefaultProps** (o `this.state` en el constructor y `class.defaultProps` con clases ES6)
- Existen 3 momentos en el ciclo de un componente: mounting, updating, unmounting (creación, actualización, destrucción)

Ciclo de vida de un componente

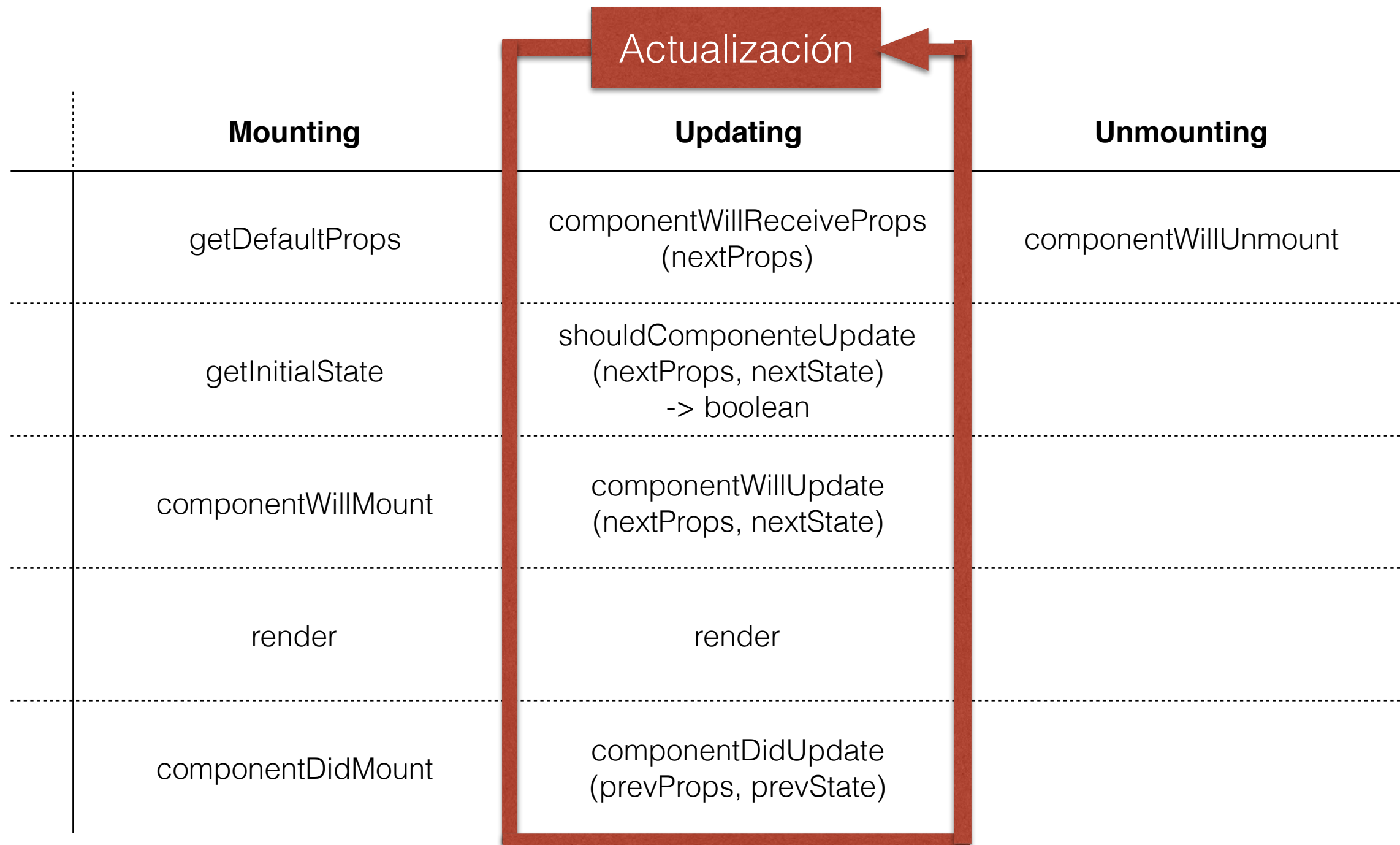
 class constructor()

	Mounting	Updating	Unmounting
	getDefaultProps	componentWillReceiveProps (nextProps)	componentWillUnmount
	getInitialState	shouldComponentUpdate (nextProps, nextState) -> boolean	
	componentWillMount	componentWillUpdate (nextProps, nextState)	
	render	render	
	componentDidMount	componentDidUpdate (prevProps, prevState)	

Ciclo de vida de un componente



Ciclo de vida de un componente



Ciclo de vida de un componente

			Destrucción
	Mounting	Updating	Unmounting
	getDefaultProps	componentWillReceiveProps (nextProps)	componentWillUnmount
	getInitialState	shouldComponentUpdate (nextProps, nextState) -> boolean	
	componentWillMount	componentWillUpdate (nextProps, nextState)	
	render	render	
	componentDidMount	componentDidUpdate (prevProps, prevState)	

Ciclo de vida de un componente

- El flujo en React siempre es unidireccional como se ve en la tabla anterior, en la etapa de actualización
- Dentro de **render** no modificamos ni propiedades ni estado, lo hacemos en los manejadores de eventos
- **render** es una función pura: dadas las mismas props y mismo estado devuelve exactamente lo mismo

Optimización

- Los métodos del ciclo de vida nos permiten influir en el mismo o conocer el momento actual de nuestro componente (¿estoy ya visible? ¿me he actualizado?)
- **shouldComponentUpdate** es el método con el que podemos *cancelar* una llamada a render devolviendo *false*. Por defecto devuelve *true*
- Tenemos acceso a las próximas *props* y próximo *state* por lo que podemos decidir que no necesitamos otro render
- De esta forma ahorramos a React que ejecute **render** para al final no cambiar nada

Optimización

- React facilita una utilidad para esto: **shallowCompare**
- **npm install -S react-addons-shallow-compare**

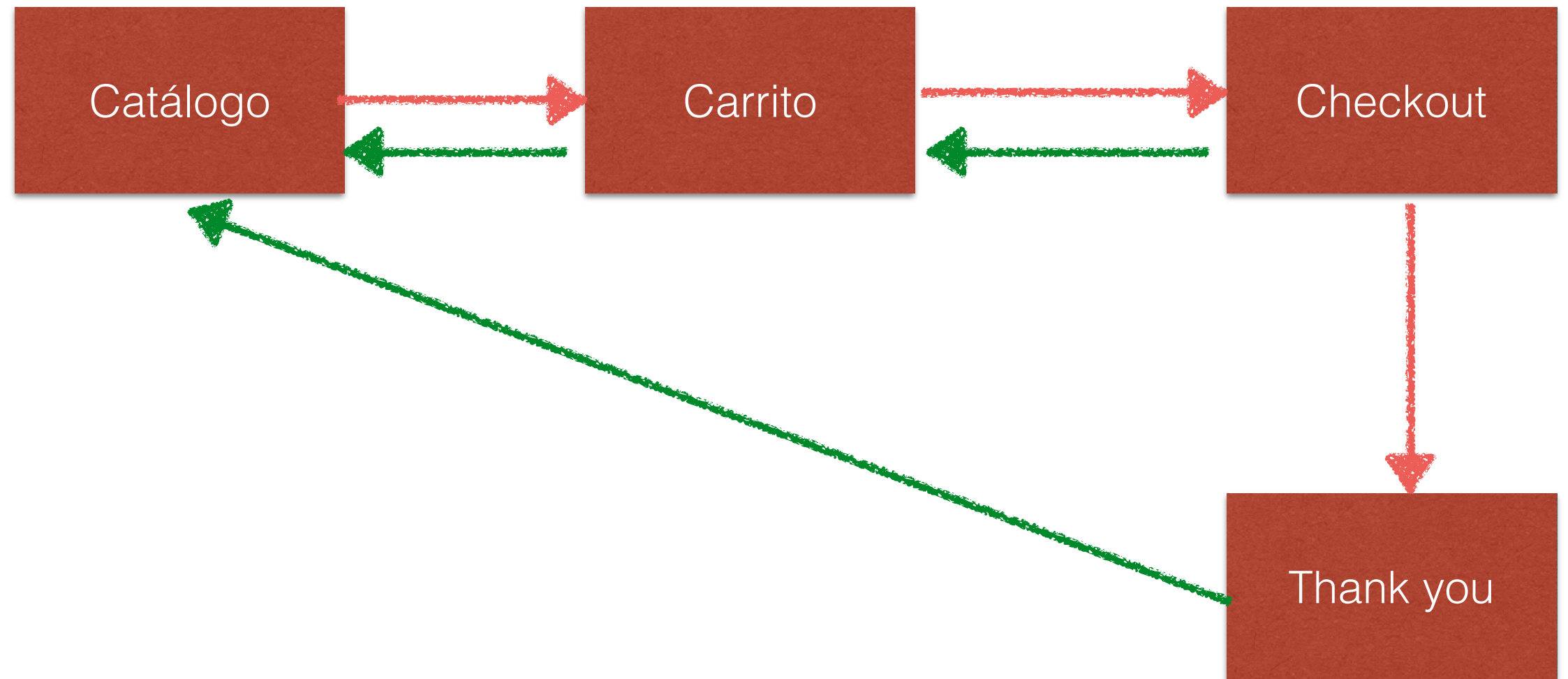
```
import React, { Component } from 'react';
import shallowCompare from 'react-addons-shallow-compare';

export default class MyComponent extends Component {
  shouldComponentUpdate(nextProps, nextState) {
    //comparación "superficial"
    return shallowCompare(this, nextProps, nextState)
  }
  render() {
    const { nombre, apellido, edad } = this.props;
    return (
      <div>
        Nombre: { nombre }<br />
        Apellido: { apellido }<br />
        Edad: { edad }
      </div>
    )
  }
}
```

Ejercicio - Ecommerce

- Vamos a implementar una micro tienda que contiene diferentes pantallas:
 - Catálogo - se muestran la **lista** de productos y se pueden añadir al carrito
 - Carrito - se muestran la **lista** de los productos escogidos, se manipula su cantidad y se vuelve al catálogo o se va al checkout
 - Checkout - se piden datos del usuario, se **validan** y, si es correcto, se va a la página de gracias
 - Confirmación - se muestra un mensaje de confirmación y se puede volver al Catálogo.

Ejercicio - Ecommerce



Ejercicio - Ecommerce

- Tendremos que mostrar la pantalla adecuada según el estado de nuestra tienda
- Podemos mantener una clave **page** en el estado del componente raíz
- La modificamos con **setState({ page: XXX })** cuando queramos navegar entre páginas
- La utilizamos para decidir qué componente pintar

Ejercicio - Ecommerce

```
render() {  
  return (  
    <div className="shopping-cart">  
      { this.getPageComponent(this.state.page) }  
    </div>  
  );  
}
```

Ejercicio - Ecommerce

```
getPageComponent(page) {  
  switch(page) {  
    case 'catalog':  
      return <Catalog ... />;  
    case 'cart':  
      return <Cart ... />  
    case 'checkout':  
      return <Checkout ... />;  
    case 'thank-you':  
      return <ThankYou ... />;  
  }  
}
```

Ejercicio - Ecommerce

- Plantilla HTML disponible en **/ejercicios/tema3/src/plantillas/shoppingcart.html**
- Datos del catálogo en **/ejercicios/tema3/src/data/catalog.js**