ALCYON
IT Servicios

PARTE II: Manejo de eventos y funciones

ALCYON
IT Servicios

INTRODUCCIÓN

En este documento veremos:

- Qué son las etiquetas vacías <>...</>
- Cómo crear eventos
- Cómo crear funciones
- Utilización de constantes
- Utilización del event.target
- Uso de hook useRef

INTRODUCCIÓN

Creación del proyecto

Crearemos un nuevo proyecto desde el cmd o el terminal de VSC:

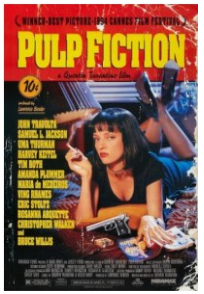
create-react-app eventos

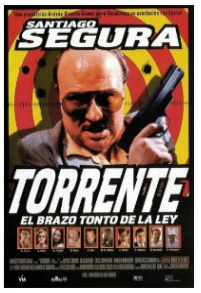
Preparar el entorno

Vamos a editar el fichero **src/App.css** y borraremos el contenido

Editamos el fichero **src/App.js** y vaciamos el contenido del return de la función App()

Crearemos una carpeta **src/img** y guardaremos dos portdas de películas de las que utilizamos en la actividad del documento anterior u otras nuevas





ALCYON
IT Servicios

INTRODUCCIÓN

Crear el contenido jsx en App.js

Editamos el fichero **App.js** para añadir el contenido que vamos a mostrar en el navegador

```
<div className="App">
      <p className='contador'>1</p>
      <p><button>Convertir</button></p>
      <div className='imagen'>
        <img />
      </div>
      <p><input className='texto'/></p>
</div>
```

Y el fichero App.css para añadir el css necesario

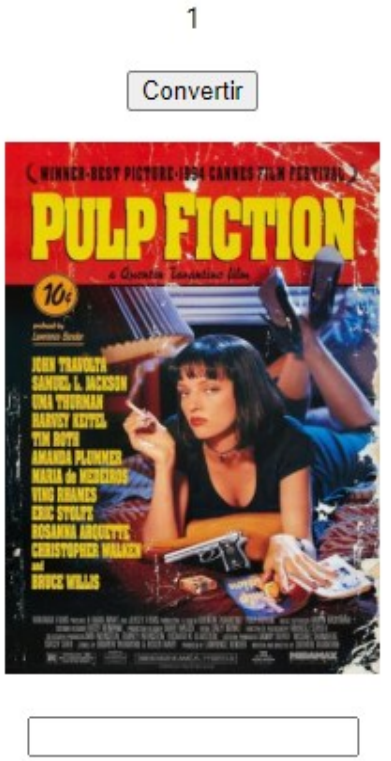
```
* {text-align: center;}
img {width: 200px;}
```

INTRODUCCIÓN

Crear el contenido jsx en App.js

Por último, importamos las dos imágenes comentadas antes y seleccionamos una de ella para mostrarla por defecto

```
import imagen0 from './img/pulp-fiction.jpg';
import imagen1 from './img/torrente.jpg';
…
<img src={imagen0} alt='peliculas'/>
```



ALCYON
IT Servicios

ETIQUETAS VACIAS

Hemos visto que todo el contenido de la página se encuentra dentro de la etiqueta . Es obligatorio que exista una caja contenedora para el resto de componentes por lo que no podemos omitir esta caja. Lo que si podemos hacer es utilizar las etiquetas <> y </> como etiquetas contenedoras del resto de etiquetas

~~ ~~
<>
    …/...
</>
~~ ~~

NOTA: Otra alternativa es utilizar <Fragment></Fragment> pero implica importarlo con: import { Fragment } from 'react'

ALCYON
IT Servicios

EVENTOS

Vamos a añadir algunos eventos para practicar como utilizarlos con react. A tener en cuenta que, al igual que sucede en otros frameworks como por ejemplo Angular, los eventos se añaden en linea, dentro de las etiquetas HTML y no se utilizan listeners

Ejercicio 1: Contador

Añadiremos un evento en la etiqueta con className 'contador' de forma que, cada vez que pulsemos sobre la etiqueta, se incremente en uno el número que forma parte del contenido de la misma

```
<p className='contador' onClick={incrementar}>1</p>
```

Indicamos que, cuando se pulse el botón del ratón se ejecute la función incrementar que definiremos en el siguiente paso

Además mostraremos el contador inicial desde una variable que definiremos después:

```
<p className='contador' onClick={incrementar}>{contador}</p>
```

ALCYON
IT Servicios

EVENTOS

Ejercicio 1: Contador

Dentro de la función App() definimos la función incrementar. Tenemos dos opciones:

- Sintaxis clásica

  function incrementar() { … }

- Sintaxis moderna en forma de flecha (es la recomendada)

  const incrementar = () => { … }

  NOTA: En javascript en general se suelen utilizar siempre constantes para aquellas variables o funciones no mutables

ALCYON
IT Servicios

EVENTOS

Ejercicio 1: Contador

Vamos a crear también la variable contador inicializada a 1

```
let contador = 1
```

Y que utilizaremos para sumarle uno y volver a escribir el nuevo contenido.

Más adelante veremos que no funcionará el Data Binding

Utilizaremos un objeto especial que se genera de forma automática cada vez que se activa un evento y que se conoce como objeto eventObject

ALCYON
IT Servicios

EVENTOS

Ejercicio 1: Contador

El objeto eventObject se genera automáticamente al hacer click en la caja y lo capturaremos como parámetro de entrada en la función incrementar:

- Sintaxis clásica

  function incrementar(**e**) { **console.log(e)** }

- Sintaxis flecha

  const incrementar = (**e**) => { **console.log(e)** }

Si vemos la consola veremos que este objeto contiene infinidad de información sobre la etiqueta donde se ha producido el evento. Concretamente nos interesará el objeto e.target y, más concretamente, e.target.innerText para recuperar el contenido de la etiqueta

EVENTOS

Ejercicio 1: Contador

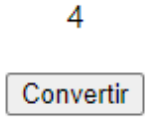
Para sumar uno al contenido de la etiqueta haríamos:

contador++

Vemos que el contador en pantalla no cambia y esto es porque **react** no permite que utilicemos variables para leer contenido o atributos de etiquetas y mucho menos modificarlo. React tiene un control exclusivo del DOM de forma que todas las variables para interactuar con éste las controla React

Para enviar el contenido del contador al DOM utilizaremos:

e.target.innerText = contador



EVENTOS

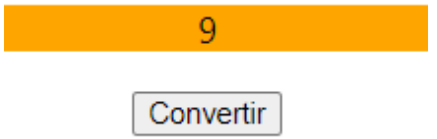
Ejercicio 1: Contador

Vamos a añadir una mejora consistente en evitar que el contador crezca de forma infinita. En cuanto llegue a 10 en vez de continuar con 11 se volverá a reiniciar a 1:

```
if (contador > 10) contador = 1;
```

Y otra mejora consistente en avisar de forma visual al usuario cuando el contador está por encima de 8 para advertirle que se acerca al límite. Por ejemplo cambiaremos el color de fondo de la caja utilizando el atributo style.backgroundColor

```
if (contador > 8) {
    e.target.style.backgroundColor = 'orange'
} else {
    e.target.style.backgroundColor = 'transparent'
}
```



NOTA: en JS no podemos utilizar background-color porque el guión es la resta en todos los lenguajes de programación

ALCYON
IT Servicios

REFERENCIAS

Ejercicio 2: Cambio de divisas

Vamos a ver en este segundo ejercicio como utilizar las referencias de React para poder acceder al contenido de una etiqueta cuando se produce un evento en una etiqueta distinta (fijaos que en el ejercicio anterior accedemos al contenido de la propia etiqueta que tiene el evento)

Haremos un ejercicio en el cual, cuando pulsemos el botón '*Convertir*', calcularemos el importe en Leu (divisa de Rumanía) la cantidad que tengamos en la etiqueta del contador que hemos utilizado antes y dejar el resultado en esta misma etiqueta

ALCYON
IT Servicios

REFERENCIAS

Ejercicio 2: Cambio de divisas

Añadimos el evento al botón '*convertir*':

```
<p><button onClick={convertir}>Convertir</button></p>
```

Creamos la función correspondiente

```
const convertir = () => { … }
```

Y, nos creamos una constante con el cambio a fecha agosto/23

```
const cambio = 4.93;
```

ALCYON
IT Servicios

REFERENCIAS

Ejercicio 2: Cambio de divisas

¿Cómo recuperamos el valor del contador para aplicar el cambio?

Pues podríamos hacerlo utilizando el sistema tradicional que conocemos de JS utilizando un id. Por ejemplo

```
document.getElementById('contador').innerText
```

Pero react nos recomienda que utilicemos las **referencias** que es un sistema parecido al que hemos utilizado con las imágenes, es decir, asignar un nombre (o referencia) a la etiqueta a la que queremos acceder.

Para eso utilizaremos un **hook** de React llamado **useRef**

ALCYON
IT Servicios

REFERENCIAS: useRef

Ejercicio 2: Cambio de divisas

Primero vamos a definir una constante con el nombre de la referencia que quiero utilizar en la etiqueta del contador y le asigno el método del hook useRef() de React:

```
const refCaja = useRef();
```

NOTA: Fijaos que VSC nos abre una lista de opciones cuando escribimos useRef, si escogemos este hook de la lista veréis como, de forma automática, nos incorpora la linea de import del hook:

```
import { useRef } from 'react';
```

Una vez importado el hook y creada la variable refCaja vamos a asignar esta referencia a la caja del contador:

```
<p ref={refCaja} className='contador' onClick={incrementar}>1</p>
```

REFERENCIAS: useRef

Ejercicio 2: Cambio de divisas

Y, por último, en nuestra función convertir, recuperamos el valor de la caja del contador utilizando la referencia, calculamos el cambio y mostramos el resultado en la misma caja:

```
refCaja.current.innerText = Number(refCaja.current.innerText) *  cambio
```

NOTA 1: Es obligatorio incluir current después de refCaja

NOTA 2: Convertimos a numérico el contenido de la caja contador

Si probamos el ejercicio debería funcionar correctamente

ALCYON
IT Servicios

EVENTOS: TARGET

Ejercicio 3: Cambiar imagen

Vamos a realizar un tercer ejercicio consistente en cambiar alternativamente la imagen de la portada de la película cada vez que pulsemos sobre ella de forma que, si tenemos la imagen de nuestra amiga Mia de Pulp Fiction y pulsamos sobre ella, cambiaremos a la imagen de nuestro también amigo José Luis de Torrente y vice-versa

Para ello utilizaremos también el evento onClick y el atributo target del objeto que se genera al producirse el evento

Primero incorporamos el evento en la etiqueta de imagen

```
<img src={imagen0} alt='peliculas' onClick={cambiar}/>
```

ALCYON
IT Servicios

EVENTOS: TARGET

Ejercicio 3: Cambiar imagen

En segundo lugar creamos la función con el argumento de entrada event como parámetro

```
const cambiar = (e) => { … }
```

Para saber que imagen mostrar tenemos que conocer que imagen tenemos visible cada vez que se activa el evento para mostrar la otra. Para ello podemos utilizar el atributo src de la imagen:

```
if (e.target.src = 'pulp-fiction.jpg') {
    e.target.src = imagen1
} else {
    e.target.src = imagen0
}
```

ALCYON
IT Servicios

EVENTOS: TARGET

Ejercicio 3: Cambiar imagen

Si probamos vemos que no funciona. ¿Por qué?

Pues, si mramos el nombre de la imagen utilizando un console.log o un alert de e.target.src vemos:

```
http://localhost:3000/static/media/pulp-
fiction.44dac5134a4b23cbe693.jpg
```

React añade automáticamente un hash a las imágenes (y a otros archivos externos) que cambia cada vez que se carga el componente

ALCYON
IT Servicios

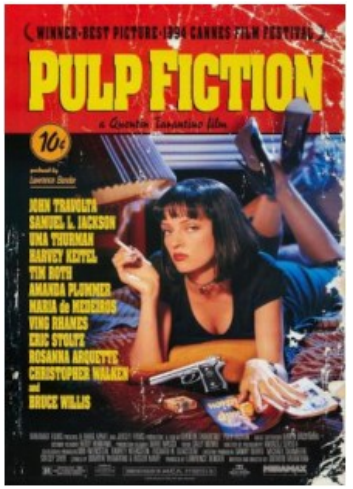
EVENTOS: TARGET

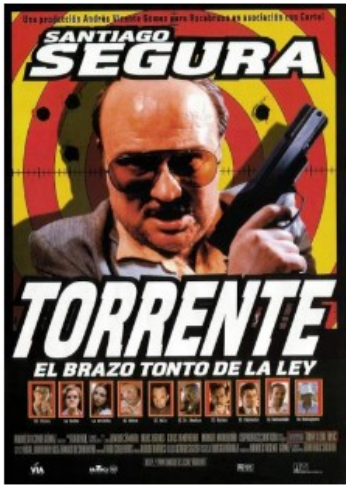
Ejercicio 3: Cambiar imagen

¿Cómo lo solucionamos?:

Pues vamos a utilizar uno de los métodos que tiene JS para comprobar si en un texto se encuentra una porción de texto: includes

```
if (e.target.src.includes('pulp')) {
    e.target.src = imagen1
} else {
    e.target.src = imagen0
}
```





NOTA: includes es un método para comprobar si existe un valor dentro de un array pero JS, al igual que otros lenguajes de programación, tratan los textos como un array de carácteres

ALCYON
IT Servicios

ACTIVIDAD 2

ACTIVIDAD

Para realizar esta actividad utilizaremos la que ya tenemos del documento anterior: la actividad 1 con las tres fichas de películas.

La actividad tiene dos partes:

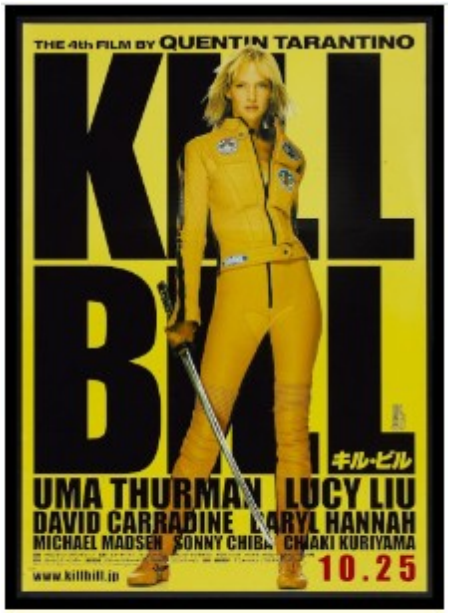
- Cuando pulsemos en el título de una película cambiaremos el texto de esa ficha a 'Visto' y, si volvemos a pulsar, eliminamos el texto

→

→

ACTIVIDAD

- Cuando pulsemos en la imagen de una ficha, ésta cambiará a la imagen **sinportada.jpg** y eliminaremos el color de fondo de la caja contenedora y, cuando pulsemos una segunda vez, eliminaremos la imagen pero conservando el espacio que ocupaba para evitar que el texto del título invada el espacio que ocupaba la imagen



Kill Bill

→

Kill Bill

→

Kill Bill

ALCYON
IT Servicios

ACTIVIDAD: SOLUCION

1. Cambiar texto

En primer lugar activaremos el evento onClick en los tres títulos. Es importante realizar el ejercicio para que no utilicemos código innecesario, es decir, los tres eventos ejecutarán la misma función

```
<h3 onClick={cambiarTexto}>{titulos[0]}</h3>
```

Confeccionamos la función para recoger como parámetro el objeto creado al activarse el evento

```
const cambiarTexto = (e) => { … }
```

ALCYON
IT Servicios

ACTIVIDAD: SOLUCION

1. Cambiar texto

En vez de preguntar por el texto que corresponde al título de la imagen, vamos a simplificar código y preguntar por el texto '*Visto*' de forma que, si tenemos este texto lo eliminamos y, en caso contrario, cambiamos el texto que corresponderá al título de la imagen por el texto '*Visto*':

```
if (e.target.innerText === 'Visto' ) {
        e.target.innerText = ''
} else {
        e.target.innerText = 'Visto'
}
```

ALCYON
IT Servicios

ACTIVIDAD: SOLUCION

2.1. Cambiar imagen

Primero importaremos la nueva imagen **sinportada.jpg** a utilizar en el ejercicio:

```
import imagen3 from './img/sinportada.jpg';
```

En segundo lugar activaremos el evento onClick en las tres imágenes. Es importante realizar el ejercicio para que no utilicemos código innecesario, es decir, los tres eventos ejecutarán la misma función

```
<img src={imagen0} alt='Kill Bill' onClick={cambiarImagen}/>
```

Confeccionamos la función para recoger como parámetro el objeto creado al activarse el evento

```
const cambiarImagen = (e) => { … }
```

ALCYON
IT Servicios

ACTIVIDAD: SOLUCION

2.1. Cambiar imagen

En vez de preguntar por el nombre de la imagen, vamos a simplificar código y preguntar por la imagen '*sinportada*' de forma que, si tenemos este texto como parte del nombre de la imagen, la ocultamos y, en caso contrario, cambiamos la imagen que corresponde a la película por la imagen '*sinportada*' (la **imagen3** que hemos importado):

```
if (e.target.src.includes('sinportada')) {
        e.target.style.visibility = 'hidden' //ocultamos la imagen sin que se libere el espacio
} else {
        e.target.src = imagen3 //cambiar la imagen a sinportada.jpg
}
```

ALCYON
IT Servicios

ACTIVIDAD: SOLUCION

2.2. Cambiar color de fondo

Para cambiar el color de fondo de la caja contenedora vamos a utilizar una propiedad JS llamada parentNode que nos permite situarnos en un objeto del DOM padre a partir de uno de los nodos hijos.

En nuestro caso queremos acceder a la caja con className 'ficha' desde el nodo hijo <img>

Por lo tanto, dentro de la función cambiarImagen añadiremos

```
e.target.parentNode.style.backgroundColor = 'transparent'
```

O bien:

```
e.target.closest('article').style.backgroundColor = 'transparent'
```

Aviso Legal

Los derechos de propiedad intelectual sobre el presente documento son titularidad de David Alcolea Martinez Administrador, propietario y responsable de www.alcyon-it.com El ejercicio exclusivo de los derechos de reproducción, distribución, comunicación pública y transformación pertenecen a la citada persona.

Queda totalmente prohibida la reproducción total o parcial de las presentes diapositivas fuera del ámbito privado (impresora doméstica, uso individual, sin ánimo de lucro).

La ley que ampara los derechos de autor establece que: “La introducción de una obra en una base de datos o en una página web accesible a través de Internet constituye un acto de comunicación pública y precisa la autorización expresa del autor”. El contenido de esta obra está protegido por la Ley, que establece penas de prisión y/o multa, además de las correspondientes indemnizaciones por daños y perjuicios, para quienes reprodujesen, plagiaren, distribuyeren o comunicaren públicamente, en todo o en parte, o su transformación, interpretación o ejecución fijada en cualquier tipo de soporte o comunicada a través de cualquier medio.

El usuario que acceda a este documento no puede copiar, modificar, distribuir, transmitir, reproducir, publicar, ceder, vender los elementos anteriormente mencionados o un extracto de los mismos o crear nuevos productos o servicios derivados de la información que contiene.

Cualquier reproducción, transmisión, adaptación, traducción, modificación, comunicación al público, o cualquier otra explotación de todo o parte del contenido de este documento, efectuada de cualquier forma o por cualquier medio, electrónico, mecánico u otro, están estrictamente prohibidos salvo autorización previa por escrito de David Alcolea. El autor de la presente obra podría autorizar a que se reproduzcan sus contenidos en otro sitio web u otro soporte (libro, revista, e-book, etc.) siempre y cuando se produzcan dos condiciones:

1. Se solicite previamente por escrito mediante email o mediante correo ordinario.

2. En caso de aceptación, no se modifiquen los textos y se cite la fuente con absoluta claridad.

David Alcolea
david-alcolea@alcyon-it.com
www.alcyon-it.com

Queda totalmente prohibida la reproducción total o parcial de las presentes diapositivas fuera del ámbito privado (impresora doméstica, uso individual, sin ánimo de lucro) así como su distribución sin la autorización explícita y por escrito de su autor.