JSX (práctica)

Tiempo estimado: 15min

El objeto de esta práctica es afianzar, reforzar y consolidar los conocimientos teóricos presentados en la lección.

Al finalizarla, el estudiante:

- Habrá añadido contenido HTML mediante JSX.

Objetivos

El objetivo de la práctica es mostrar cómo trabajar con JSX. Por un lado, cómo añadir código HTML a una página desde el código JavaScript de la aplicación. Por otra parte, cómo generar el archivo empaquetado, teniendo en cuenta que Browserify no sabe trabajar con JSX sino sólo con JavaScript.

Preparación del entorno

Para comenzar, crearemos un proyecto de aplicación React mediante el generador de Justo, ya instalado en la práctica anterior:

1. Abrir una consola.
2. Crear el directorio de la práctica e ir a él.
3. Invocar el generador de Justo:

   ```
   > justo -g react
   ```

   Responder a las preguntas realizadas por el generador, teniendo en cuenta lo siguiente:

   - Responda N cuando le pregunte si usar React Router.
4. Instalar las dependencias del proyecto:

   ```
   > npm install
   ```
5. Mostrar el catálogo de tareas automatizadas del proyecto:

   ```
   > justo -c
   ```
6. Invocar la tarea build del catálogo del proyecto para construir la aplicación:

   ```
   > justo build
   ```
7. Abrir el archivo dist/index.html con el navegador, por ejemplo, mediante la tarea catalogada chrome:

   ```
   > justo chrome
   ```

   Debe aparecer el mensaje Hello World!

Añadidura de código HTML

En este punto, la idea es generar una tabla HTML con contenido, primero estáticamente y después dinámicamente:

1. Editar el archivo app/index.jsx.

   Tal como veremos en una lección posterior, el componente raíz de la aplicación React se especifica mediante ReactDOM.render(). En este caso, se indica un elemento vista. Puede observar su definición en app/views/App.jsx. Concretamente, vaya a su método render().

   La idea es sustituir el contenido de la vista <App /> por código HTML.
2. Sustituir la definición del elemento raíz por lo siguiente:

   ```
   ReactDOM.render(
     <table>
       <thead>
   ```



```
        <tr>
          <th>Banda</th>
          <th>Año</th>
        </tr>
      </thead>

      <tbody>
        <tr>
          <td>The National</td>
          <td>1999</td>
        </tr>

        <tr>
          <td>Echo & the Bunnymen</td>
          <td>1978</td>
        </tr>

        <tr>
          <td>Nada Surf</td>
          <td>1992</td>
        </tr>

        <tr>
          <td>Perturbazione</td>
          <td>1988</td>
        </tr>
      </tbody>
    </table>,
    document.getElementById("react-app")
);
```

3. Guardar cambios.
4. Ir a la consola.
5. Construir la aplicación y abrirla en el navegador:

```
> justo build chrome
```

Debe mostrar:

Banda	Año
The National	1999
Echo & the Bunnymen	1978
Nada Surf	1992
Perturbazione	1988

6. Editar el archivo app/index.jsx.
7. Definir la constante de datos siguiente, después de la sección de importaciones:

```
const bands = [
  {
    name: "The National",
    year: 1999,
    origin: "US"
  },
  {
    name: "Echo & the Bunnymen",
    year: 1978,
    origin: "UK"
  },
  {
    name: "Nada Surf",
    year: 1992,
    origin: "US"
  },
  {
    name: "Perturbazione",
```



```
    year: 1988,
    origin: "Italy"
  }
];
```

8. Mostrar una tabla con los datos de la constante anterior, usando el método map() del *array*. Para ello, reemplazar el elemento raíz de la aplicación por:

```
ReactDOM.render(
  <table>
    <thead>
      <tr>
        <th>Banda</th>
        <th>Año</th>
        <th>Origen</th>
      </tr>
    </thead>

    <tbody>
      {
        bands.map(function(band) {
          return (
            <tr>
              <td>{band.name}</td>
              <td>{band.year}</td>
              <td>{band.origin}</td>
            </tr>
          );
        })
      }
    </tbody>
  </table>,
  document.getElementById("react-app")
);
```

El método map() devolverá un *array* como el siguiente:

```
[
  <tr><td>The National</td> ...</tr>,
  <tr><td>Echo & the Bunnymen</td> ...</tr>,
  <tr><td>Nada Surf</td> ...</tr>,
  <tr><td>Perturbazione</td> ...</tr>
]
```

Siendo estos elementos lo que se añadirán al elemento <tbody>.

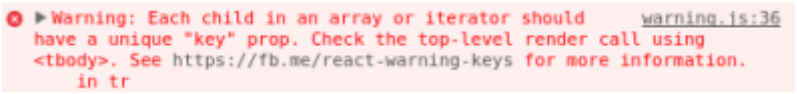
9. Guardar cambios.
10. Ir a la consola.
11. Reconstruir y abrir la aplicación:

```
> justo build chrome
```

Ahora, debe mostrar:

Banda	Año	Origen
The National	1999	US
Echo & the Bunnymen	1978	UK
Nada Surf	1992	US
Perturbazione	1988	Italy

12. Ir al navegador.
13. Abrir la consola de JavaScript. En el caso de Chrome, mediante Ctrl+Shift+J.
14. Observar el mensaje de error generado por React, pero que no le ha impedido hacer su trabajo:



El error se debe a que no añadimos el atributo clave en el elemento <tr> para cada fila generada dinámicamente mediante la función de transformación pasada al método map() del *array*.

15. Editar el archivo app/index.jsx.
16. Añadir el atributo key, fijando su valor al nombre de la banda:

```
<tbody>
  {
    bands.map(function(band) {
      return (
        <tr key={band.name}>
          <td>{band.name}</td>
          <td>{band.year}</td>
          <td>{band.origin}</td>
        </tr>
      );
    })
  }
</tbody>
```

17. Guardar cambios.
18. Ir a la consola.
19. Reconstruir y reabrir la aplicación:

```
> justo build chrome
```

20. Abrir la consola de JavaScript y comprobar que ya no aparece el mensaje de error anterior.

Cada vez que genere código HTML dinámicamente, no olvide añadir el atributo key a los elementos <tr> y <li>. Es muy frecuente hacerlo.

Uso de Browserify

A lo largo de la práctica, hemos usado la tarea build del proyecto para generar el archivo empaquetado de la aplicación React. Recordemos que Browserify no entiende JSX, pero sí JavaScript. Así pues, lo que hay que hacer es transformar JSX a JavaScript puro, antes de que Browserify lo procese. Esto se consigue usando el transformador babelify, un componente de Browserify que usa el *transpiler* Babel. La tarea build lo hacía por nosotros, pero ahora veamos cómo hacerlo manualmente. Esto nos ayudará a comprender mejor el funcionamiento de Browserify.

1. Ir a la consola.
2. Comprobar si tenemos instalado browserify globalmente:

```
> browserify -h
```

Si no lo está, instalarlo:

```
> npm install -g browserify
```

3. Generar el archivo empaquetado mediante browserify:

```
> browserify app/index.jsx --extension=.jsx -o dist/scripts/react-app.js -t [ babelify
--presets [ es2015 react ] ]
```

4. Abrir la aplicación en el navegador:

```
> justo chrome
```

