

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

Al finalizarla, el estudiante:

- Habrá trabajado usando **React Router** en una aplicación **React**.
- Habrá trabajado con varias vistas en una aplicación **React**.

Objetivos

El objetivo de la práctica es mostrar cómo trabajar con distintas vistas y, por ende, con el encaminador de **React**. Para ello, vamos a crear una pequeña aplicación que, mediante una ruta estática muestra una vista que lista bandas y, en otra dinámica, el contenido de una determinada banda seleccionada por el usuario.

Preparación del entorno

Para comenzar, crearemos un proyecto de aplicación **React** mediante el generador de **Justo**:

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

```
> justo -g react
```

A la hora de responder:

- Indicar **Y** cuando el generador pregunte si usará **React Router**.
- Seleccionar **hashHistory** cuando le pregunte qué historia de URLs usar.
Las historias de URLs se presentan en la siguiente lección.
- Seleccionar **Font Awesome** cuando le pregunte bibliotecas adicionales a usar en la aplicación.
Esto creará un elemento `<link>` en `app/index.html`, permitiendo así su uso.

4. Echar un vistazo al proyecto generado.

Por favor, tómese su tiempo. Preste especial atención a:

- El archivo `app/index.html`. Busque el `<link>` que incorpora la biblioteca **Font Awesome**.
- El archivo `app/routes/map.jsx` que define el mapa de rutas de la aplicación.
- El archivo `app/index.jsx` que ha dejado de tener como componente raíz `<App/>` y ha pasado a tener `<Router>`.
- La carpeta de vistas `app/views`, donde se definen la vista envoltorio de la aplicación, `App.jsx`, y la índice de la raíz de la aplicación, `AppIndex.jsx`. Eche un vistazo a cada una de las vistas.

5. Instalar las dependencias del proyecto:

```
> npm install
```

6. Mostrar el catálogo de tareas automatizadas, mediante **Justo**, del proyecto:

```
> justo -c
```

7. Invocar la tarea **build** del catálogo del proyecto para construir la aplicación:

```
> justo build
```

8. Abrir el archivo `dist/index.html` con el navegador.

Debe aparecer lo siguiente:

```
Hello world!  
View:AppIndex
```

Creación rutas

La aplicación dispondrá de una ruta compuesta `/bands` que contendrá una vista que lista todas las bandas dadas de alta; y otra, `/bands/:id`, que muestra los datos de las bandas individualmente. Vamos a crear el archivo de rutas para esta sección de la aplicación y las rutas correspondientes.

1. Editar el archivo `app/routes/map.jsx` y echarle un vistazo detenidamente.

Observe que se define una ruta compuesta para la raíz de la aplicación, `/`. En ella, se define como vista envoltorio la vista `App` y como su vista índice `AppIndex`. Recordemos que la vista envoltorio se muestra siempre que se acceda a algo que cuelgue de su ruta asociada. Mientras que la índice sólo cuando se acceda directamente a la ruta compuesta directamente, sin acceso particular a ninguna ruta que cuelgue de ella.

2. Ir a la consola.
3. Generar el archivo de rutas `bands` mediante el generador:

```
> justo -g react route file
```

Tenga en cuenta lo siguiente:

- Indique `bands` como nombre del archivo.
- Indique `bands` como punto de montaje o *path*.
- Seleccione `Immutable` como tipo de vista envoltorio (*layout view*).
- Seleccione `Immutable` como tipo de vista índice (*index view*).

Observe que se crea la carpeta `app/views/bands` donde ubicar sus vistas específicas. Es simplemente para facilitar el mantenimiento y la organización del código. También se crea las vistas `Layout` e `Index` para la ruta compuesta que estamos creando.

4. Editar el archivo `app/routes/map.jsx` y observar que se ha insertado una referencia al nuevo archivo de rutas:

```
{require("../bands").default}
```

5. Editar el archivo `app/routes/bands.jsx` y echarle un vistazo detenidamente.

La ruta compuesta tiene como vista envoltorio `app/views/bands/Layout` y como índice `app/views/bands/Index`. Ambas las ha creado el generador.

6. Editar el archivo `app/views/bands/Layout` y echarle un vistazo detenidamente.
7. Editar el archivo `app/views/bands/Index` y echarle un vistazo detenidamente.

Ésta es la vista que utilizaremos para mostrar la tabla de bandas.

8. Ir a la consola.
9. Añadir la vista `Info` a la carpeta `/app/views/bands` mediante el generador:

```
> justo -g react view
```

Esta vista la utilizaremos para mostrar la información de una determinada banda. Responda teniendo en cuenta lo siguiente:

- Seleccione `bands` como carpeta donde crearla.
- Indique `Info` como nombre de vista.
- Seleccione `Immutable` como tipo de componente vista.

10. Editar el archivo `app/views/bands/Info.jsx` y echarle un vistazo.
11. Añadir la ruta `/bands/:id` que se asocie a la vista `Info` mediante el generador:

```
> justo -g react route
```

Responda como sigue:

- Archivo de rutas donde añadir la ruta: `bands`.
- *Path*: `:id`.

Recordemos que hay que indicar la ruta relativa con respecto a su padre. No toda la ruta con respecto a la raíz de la aplicación. Así pues, al indicar `:id`, la ruta real será `/bands/:id`.

- Vista asociada: `Info`.

12. Editar el archivo `app/routes/bands.jsx` y echar un vistazo a la nueva ruta:

```
<Route path=":id" component={require("../views/bands/Info").default} />
```

Gracias al generador de **Justo**, trabajar con rutas es bastante fácil. Parte del trabajo sucio, lo hace por nosotros.

Definición de datos

Vamos a crear el archivo de datos de la aplicación:

1. Crear el archivo `bands.js` en la carpeta `app/data`:

```
export default {
  "leonard-cohen": {
    name: "Leonard Cohen",
    year: 1934,
    origin: "Canada"
  },
  "tom-petty": {
    name: "Tom Petty",
    year: 1950,
    origin: "US"
  },
  "nacho-goberna": {
    name: "Nacho Goberna",
    year: 1964,
    origin: "España"
  },
  "elvis-costello": {
    name: "Elvis Costello",
    year: 1954,
    origin: "UK"
  }
};
```

Creación de componente tabla

Ahora, vamos a crear el componente tabla que muestra los datos:

1. Ir a la consola.
2. Crear el componente tabla mediante el generador:

```
> justo -g react component
```

Respuestas:

- Cuando pregunte en qué subcarpeta añadir el componente, seleccionar **<other>** y a continuación escribir `bands`.
- Nombre del componente: `BandTable`.
- Tipo de componente: **Immutable**.
- Tipo de estructura: **Simple**.
- Tipo de implementación: **Function**.

Este componente *no* es una vista, sino un componente que usaremos en una vista. De ahí que lo creamos mediante el comando **component** y lo añadamos a la carpeta `app/components`.

3. Editar el archivo `app/components/bands/BandTable.jsx`.

4. Importar el archivo de datos:

```
import bands from "../../data/bands";
```

5. Importar el componente `Link` de `React Router`:

```
import {Link} from "react-router";
```

Este componente se presenta en la siguiente lección. Se utiliza para crear enlaces `HTML <a>`.

6. Modificar la función de reproducción del componente:

```
export default function BandTable(props) {
  return (
    <table>
      <thead>
        <tr>
          <th>Mostrar</th>
          <th>Banda</th>
          <th>Año</th>
        </tr>
      </thead>

      <tbody>
        {
          Object.keys(bands).map(function(id) {
            var band = bands[id];

            return (
              <tr key={band.name}>
                <td>
                  <Link to={`/${bands[id]}`>
                    <span className="fa fa-edit" />
                  </Link>
                </td>
                <td>{band.name}</td>
                <td>{band.year}</td>
              </tr>
            );
          })
        }
      </tbody>
    </table>
  );
}
```

No hay que olvidar definir el atributo `key`, cuando trabajamos con el método `map()` de los *arrays*.

7. Guardar cambios.

Definición de vistas

Las vistas ya las hemos creado mediante el generador, pero no las hemos modificado como corresponde para completar la práctica. Ha llegado el momento de hacerlo.

1. Editar la vista `app/views/bands/Index.jsx`.

2. Importar el componente `BandTable`:

```
import BandTable from "../../components/bands/BandTable";
```

3. Ir al método `render()` y definirlo para que muestre la tabla de bandas:

```
render() {
  return (
    <BandTable />
  );
}
```

4. Guardar cambios.

5. Editar la vista `app/views/bands/Info.jsx`.

6. Importar los datos:

```
import bands from "../../data/bands";
```

7. Ir al método `render()` y hacer que muestre el contenido de la banda indicada en el parámetro `:id` de la ruta a la que está asociada la vista:

```
render() {
  var band;

  //(1) get data
  band = bands[this.props.params.id];

  //(2) render
  return (
    <div>
      Artista: {band.name}<br />
      Año: {band.year}<br />
      Origen: {band.origin}
    </div>
  );
}
```

Recordemos que la ruta la definimos como `/bands/:id`. El valor actual del parámetro `:id` se accede mediante `this.props.params.nombreParámetro`.

8. Guardar cambios.

9. Editar el archivo `app/views/App.jsx`.

10. Añadir un enlace para poder acceder a la vista `/bands`:

```
render() {
  return (
    <div>
      <Link to="/bands"><span className="fa fa-list" /> Bandas</Link>
      {this.props.children}
    </div>
  );
}
```

11. Guardar cambios.

12. Ir a la consola.

13. Generar la aplicación y abrirla en el navegador:

```
> just build chrome
```

Debe aparecer algo como lo siguiente:



The screenshot shows a blue underlined link labeled "Bandas" and the text "View:AppIndex" below it.

Observe la ruta que aparece en el navegador. Debe ser similar a:





```
/dist/index.html#/?_k=3gq26r
```

Esto se debe a que estamos usando como historial `hashHistory`. En la próxima lección, veremos cómo mejorarlo.

Por otra parte, observe el texto `View:AppIndex`. Esto es lo que se muestra cuando se accede a la vista índice de la aplicación. Su contenido se encuentra en la vista `app/views/AppIndex`. Puede cambiarlo en cualquier momento. Lo dejamos para que comprenda cómo funcionan las cosas.

14. Hacer clic en el enlace de Bandas.

Debe ser redirigido a algo como:

Bandas		
View:bands/Layout		
Mostrar	Banda	Año
	Leonard Cohen	1934
	Tom Petty	1950
	Nacho Goberna	1964
	Elvis Costello	1954

Observe que el enlace Bandas sigue apareciendo. Esto se debe a que lo definimos en la vista `app/views/App.jsx`. La vista envoltorio que contiene todo lo que se mostrará en toda vista que cuelgue de ella. A su vez, ahora aparece el texto `View:bands/Layout`. Ésta es similar a la anterior, pero a nivel de la ruta `/bands/`. Observe que las vistas envoltorio se anidan. Generalmente, en `bands/Layout` definiríamos enlaces o cosas específicas del elemento relacionado con las bandas como, por ejemplo, un enlace para añadir una nueva banda.

- Hacer clic en el enlace de Leonard Cohen.

Debe mostrarse la vista `Info`, la cual mostrará los datos de `leonard-cohen`:

[Bandas](#)
View:bands/Layout
Artista: Leonard Cohen
Año: 1934
Origen: Canada

Observe la ruta que aparece en el navegador:

`/dist/index.html#/bands/leonard-cohen?_k=jehy41`

La parte que nos importa es `/bands/leonard-cohen`, la cual entra dentro de la ruta dinámica `/bands/:id` que definimos al comienzo de la práctica y que asociamos a la vista `Info`. La parte que sigue al interrogante (?) es específica del historial `hashHistory`. No le preste atención.

- Hacer clic en Bandas y observar que cambia el URL y, por lo tanto, la vista que debe mostrar el encaminador, teniendo en cuenta las rutas que hemos definido.
- Hacer clic en el enlace Tom Petty.

Otra vez se muestra la vista `Info`, pero esta vez con los datos referentes a Tom Petty.