upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

DESPLIEGUE DE SERVICIOS MULTIMEDIA
Máster Universitario en Ingeniería de Telecomunicación

AppGaztaroa

Commit 03: "Componentes funcionales en React Native"

Marko Galarza Galarza
Mikel Sagues García

Componentes funcionales en React Native

Commit 03: "Componentes funcionales en React Native"

Al igual que en React, en React Native también distinguimos entre dos tipos de componentes: los *Class Components* (o componentes de clase) y los *Functional Components"* (o componentes funcionales). La elección de uno u otro en cada momento depende del contexto y el objetivo que se desee lograr. Para repasar este concepto se recomienda, como siempre, la lectura la bibliografía de este capítulo.

Aunque ya hemos utilizado un componente funcional en el capítulo anterior (*Calendario*), en este capítulo profundizaremos un poquito más en este concepto y veremos la forma en que los diferentes componentes en una aplicación React Native pueden comunicarse entre ellos.

En este ejercicio:

- Profundizaremos en el concepto anteriormente señalado.
- Crearemos la primera interacción con el usuario a través de la interfaz de usuario, analizando la forma en que React Native permite el envío de parámetros entre componentes.
- Emplearemos el componente *Card* de *React Native Elements* para renderizar información.

1. Componentes funcionales en React Native

En primer lugar, añadiremos un nuevo componente funcional a nuestra aplicación. Éste será el responsable de renderizar los detalles de las excursiones montañeras que muestra nuestra aplicación. Para ello, creamos un fichero de nombre *DetalleExcursionComponent.js* en la carpeta *componentes*, con el siguiente código fuente:

```
import React from 'react';
import { Text, View } from 'react-native';
import { Card } from '@rneui/themed';

function RenderExcursion(props) {

    const excursion = props.excursion;

        if (excursion != null) {
            return(
            <Card>
              <Card.Title>{excursion.nombre}</Card.Title>
```

```
              <Card.Divider/>
              <Card.Image source={require('./imagenes/40Años.png')}></Car
d.Image>
              <Text style={{margin: 20}}>
                {excursion.descripcion}
              </Text>
            </Card>
            );
        }
        else {
            return(<View></View>);
        }
}

function DetalleExcursion(props) {
    return(<RenderExcursion excursion={props.excursion} />);
}

export default DetalleExcursion;
```

Analizar con calma:

- La forma en que el contenido es renderizado mediante el componente *Card* de React Native Elements. Consultar la bibliografía relativa a este componente.
- La forma en que las propiedades son "pasadas" de una función a otra. Éstas, a su vez, han sido proporcionadas a *DetalleExcursion* por el componente *Campobase*, que actualizaremos a continuación.

Actualizamos el componente *Campobase* para hacer que su método *return()* renderice, debajo del componente *Calendario* que habíamos creado antes (el cual a su vez contenía el *FlatList*) el detalle de una de las excursiones, mediante el componente *DetalleExcursion* que acabamos de crear. Para ello:

```
[...]
import DetalleExcursion from './DetalleExcursionComponent';
import { View } from 'react-native';

[...]

    this.state = {
      excursiones: EXCURSIONES,
      seleccionExcursion: null
    };

[...]

  onSeleccionExcursion(excursionId) {
    this.setState({seleccionExcursion: excursionId})
  }
```

```
[...]

    return (
        <View>
            <DetalleExcursion excursion={this.state.excursiones.filter((e
xcursion) => excursion.id === this.state.seleccionExcursion)[0]} />
            <Calendario excursiones={this.state.excursiones} onPress={(
excursionId) => this. onSeleccionExcursion(excursionId)} />
        </View>
    );

[...]
```

En el código anterior, prestar especial atención a lo siguiente:

- Se ha creado una nueva variable de estado: *seleccionExcursion*. Su objetivo es almacenar el identificador de la excursión que se desea renderizar.
- El componente *Calendario*, además de recibir como parámetro el array de *excursiones*, recibe también una nueva función que definimos con el nombre *onPress()*[1].
- Se ha creado una función *onSeleccionExcursion()* para actualizar el estado en lo referente a *seleccionExcursion.* Esta función es llamada dentro de la función *onPress()* que se acaba de mencionar.
- Es necesario incorporar un *View* que englobe los dos componentes dentro de la función *return()*.
- Analícese la forma en que se filtra la excursión seleccionada empleando un filtro JavaScript.

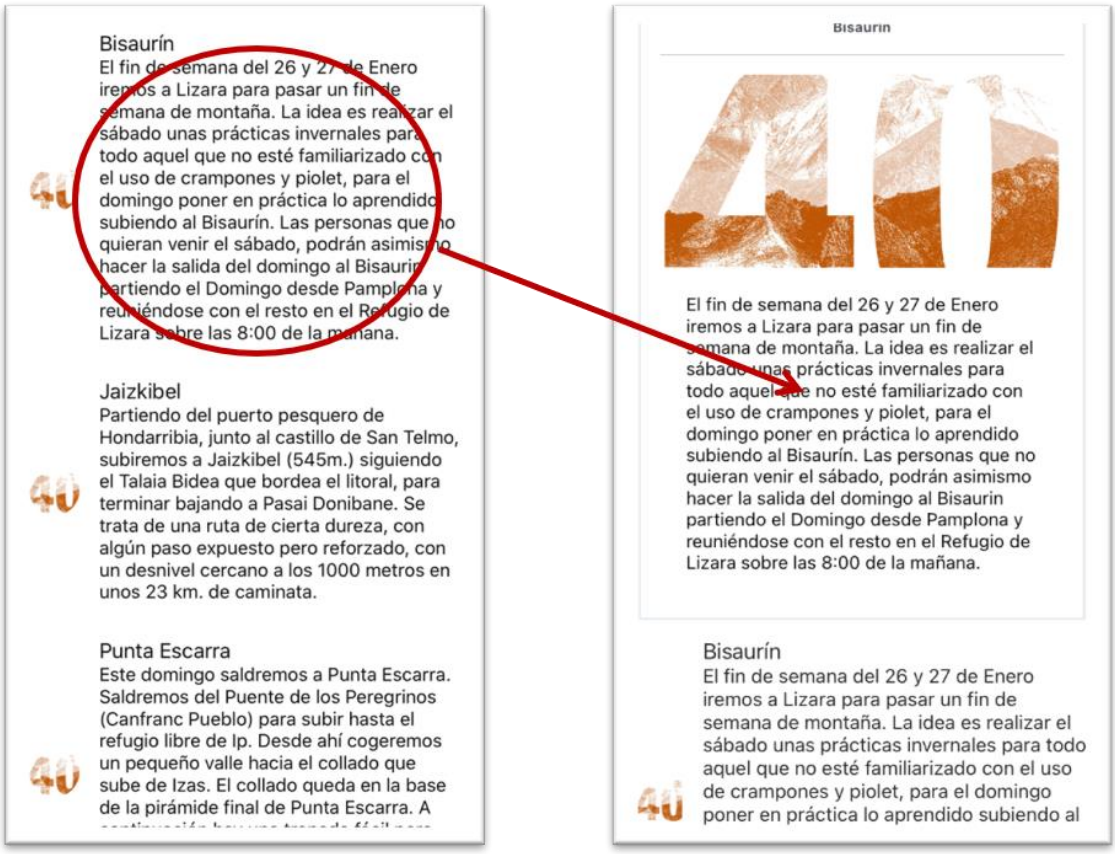
Finalmente, añadimos al *ListItem* del componente *Calendario* la función *onPress()* para que responda a la interacción del usuario, devolviendo a *Campobase* el *item.id* de la excursión seleccionada, mediante el componente *Calendario*:

```
            <ListItem
                key={index}
                onPress={() => props.onPress(item.id)}
                bottomDivider>
 [...]
                  />
```

Nótese que en este caso *onPress* de *ListItem* sí que es una función definida en React Native Elements, mientras que la función *onPress()* del componente *Calendario* podría haber recibido cualquier nombre.

Una vez completado lo anterior, nuestra aplicación debería mostrar el siguiente aspecto y funcionalidad:

[1] A pesar de estar escrito en inglés, este nombre se ha elegido aleatoriamente y podría ser cualquier otro. No se trata de una función definida en React Native.



Nótese que, en este caso, la funcionalidad es bastante "disfuncional", ya que nos hemos limitado a incorporar el componente *card* renderizado, quedándonos "bloqueados" en dicha pantalla.

Si todo es correcto, ya podemos hacer el *commit* correspondiente a este capítulo.

2. Bibliografía

- ***Function Components* en React Native**
  https://reactnative.dev/docs/intro-react?syntax=functional#your-first-component
  https://medium.com/@Zwenza/functional-vs-class-components-in-react-231e3fbd7108
  https://www.robinwieruch.de/react-function-component
- **Card** (React Native Elements)
  https://reactnativeelements.com/docs/components/card
- **ListItem** (React Native Elements)
  https://reactnativeelements.com/docs/components/listitem
- **Objetos JavaScript y objetos JSON**
  https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/Basics
  https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON

- **View**
  https://reactnative.dev/docs/view
- **Text**
  https://reactnative.dev/docs/text