Ejercicios del tema 7

Accede a uno de los siguientes enlaces, dependiendo del navegador en el que quieras probar **React Developer Tools**:

- Mozilla Firefox: https://addons.mozilla.org/es/firefox/addon/react-devtools/
- Google Chrome: https://chrome.google.com/webstore/detail/react-developer-tools/fmkadmapgofadopljbjfkapdkoienihi

Ejercicio 1 - Crea un nuevo proyecto con React e instala Foundation. Necesitarás ejecutar el siguiente comando:

```
npm install --save react-foundation-components
```

Añade el archivo foundation.min.css dentro de la carpeta **src**, tal y como se vio en el tema anterior. Puedes encontrarlo en la carpeta de recursos de este tema o del anterior, o utilizarlo de algún proyecto del tema anterior.

A continuación, sustituye el código del archivo index.js por el siguiente:

```
import React from 'react';
import ReactDOM from 'react-dom';
import './foundation.min.css';
import { Button } from 'react-foundation-components/lib/global/button'
import App from './App.js'

class Ejemplo extends React.Component {
  constructor(){
    super();
    this.state = {resta: 0, x: 0, y: 0};
    this.botonPulsado = this.botonPulsado.bind(this);
    this.handleChange = this.handleChange.bind(this);
    this.handleChange2 = this.handleChange2.bind(this);
  }
  handleChange(e) {
     this.setState({x: parseInt(e.target.value)})
   }
  handleChange2(e) {
    this.setState({y: parseInt(e.target.value)})
  }
  botonPulsado(e){
    this.setState({resta: this.state.x - this.state.y})
  }
  render() {
    return (
      <div>
        <input type="text" onChange={ this.handleChange } value={this.state.x}/>
        <input type="text" onChange={ this.handleChange2 } value={this.state.y}/>
        <Button color="success" onClick={this.botonPulsado}>Restar</Button> <br/>
        <input type="text" value={this.state.resta}></input>
      </div>
    );
  }
}

ReactDOM.render(
  <div><Ejemplo /> </div>,
  document.getElementById('root')
);
```

Prueba a ejecutar el código en el navegador y prueba a realizar cambios en las variables desde la herramienta de desarrollador. Haz una captura de pantalla mostrando cómo cambias el valor de una variable desde la herramienta.

Ampliación - Modifica el código para que el valor de la resta se actualice sin tener que pulsar el botón de restar.

Ejercicio 1 - *Debug & Testing*

Vamos a crear una aplicación que nos permita realizar sumas, restas, multiplicaciones y divisiones. Mediante **Jest** comprobaremos que los resultados coinciden con los esperados. La aplicación tendrá dos cuadros de texto para introducir dos números, cuatro botones para realizar cada una de las sumas y por último un label para mostrar el valor del resultado.

Para empezar tendremos que crear una clase Calculadora, esta clase tendrá un método constructor en el cual crearemos un state para actualizar el valor del label y los binds de **this** en los métodos para poder llamar **this.setState** y actualizar el valor del state.

```
import React, { Component } from 'react';
import './App.css';

class Calculadora extends Component {

  constructor() {
    super();
    this.state = {
      resultado: 0,
      x: 0,
      y: 0,
    };
    this.sumar = this.sumar.bind(this);
    this.restar = this.restar.bind(this);
    this.multiplicar = this.multiplicar.bind(this);
    this.dividir = this.dividir.bind(this);
    this.handleChange = this.handleChange.bind(this);
    this.handleChange2 = this.handleChange2.bind(this);
  }

  render() {
    return (
      <div>
        <div>
          <label>Primer número:
          <input type="number" onChange={ this.handleChange } value={this.state.x}/>
          </label>
          </div>
          <div>
          <label>Segundo número:
          <input type="number" onChange={ this.handleChange2 } value={this.state.y}/>
          </label>
          </div>
          <div>
            <button onClick={this.sumar}>Suma</button>
            <button onClick={this.restar}>Resta</button>
            <button onClick={this.multiplicar}>Multiplicación</button>
            <button onClick={this.dividir}>División</button>
          </div>
          <label>Resultado: {this.state.resultado}</label>
      </div>
    );
  }
```



```
handleChange(e) {
     this.setState({x: (e.target.value)})
   }

  handleChange2(e) {
    this.setState({y: (e.target.value)})
  }

  sumar() {
    var res = parseInt(this.state.x)+parseInt(this.state.y);
    this.setState({resultado: res});
    this.suma(this.state.x, this.state.y);
  }

  suma(x,y){
    var res = parseInt(x)+parseInt(y);
    console.log(res);
  }

  resta(x,y){
    var res = parseInt(x)-parseInt(y);
    console.log(res);
  }

  mult(x,y){
    var res = parseInt(x)*parseInt(y);
    console.log(res);
  }

  div(x,y){
    var res = parseInt(x)/parseInt(y);
    console.log(res);
  }

  restar() {
    var res = this.state.x-this.state.y;
    this.setState({resultado: res});
    this.resta(this.state.x, this.state.y);
  }

  multiplicar() {
    var res = this.state.x*this.state.y;
    this.setState({resultado: res});
    this.mult(this.state.x, this.state.y);

  }

  dividir() {
    var res = this.state.x/this.state.y;
    this.setState({resultado: res});
    this.div(this.state.x, this.state.y);
  }
}
```

Ahora vamos a crear la clase Calculadora.test.js, que será la encargada de realizar los tests. Por problemas que tiene Jest con el código HTML de React, copiaremos los métodos de *suma*, *resta*, *mult* y *div* en la nueva clase para testearlos uno por uno.

De esta manera, tendremos para la función de sumar su correspondiente test:

```
function suma(x,y){
  var res = parseInt(x)+parseInt(y);
  return res;
}

it('suma correctamente', () => {
  expect(suma(5,6)).toBe(11);
});
```

Si ejecutamos **$ jest Calculadora.test.js** obtendremos por pantalla:

```
Timps-MacBook-Air:src timp$ jest Calculadora.test.js
 PASS  src/Calculadora.test.js
  ✓ suma correctamente (6ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        3.196s
Ran all test suites matching "Calculadora.test.js".
Timps-MacBook-Air:src timp$
```

De esta manera comprobamos que nuestro método *suma()* actúa como es debido. A partir de este punto, implementa los tests para las funciones de restar, multiplicar y dividir. Puedes probar a cambiar el valor esperado para comprobar que esa función no pasa el test.

