Aplicaciones Node (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 creado un paquete de aplicación.
- Habrá instalado un paquete de aplicación.
- Habrá usado una aplicación Node.

Objetivos

El objetivo de la práctica es crear un paquete que instale un comando que realice sumas o restas de los valores pasados como argumentos mediante una sintaxis como la siguiente:

```
sum valor valor valor…
sub valor valor valor…
```

Preparación del entorno

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

1. Abrir una consola.
2. Comprobar si hay una versión más reciente del paquete justo-generator-node:

   ```
   $ npm outdated -g justo-cli justo-generator-node
   ```

   Si así fuera, instalar la nueva versión:

   ```
   $ npm update -g justo-cli justo-generator-node
   ```
3. Crear el directorio de la práctica e ir a él. Llámelo calcul.
4. Invocar el generador de Justo:

   ```
   $ justo -g node
   ```

   Seleccionar app cuando le pregunte el tipo de paquete. E indicar calcul como nombre del comando de aplicación a crear cuando se instale el paquete.
5. Echar un vistazo al proyecto:
   - Observe el módulo bin/calcul.js. Es el archivo en el que definiremos la lógica del comando calcul.
   - Observe la propiedad bin del archivo package.json. Indica que debe crear el comando calcul con el contenido del archivo bin/calcul.js cuando se instale el paquete mediante npm install.
   - Observe la propiedad main. Indica que el punto de entrada es el módulo index.js. Esto es así por si se desea usar el paquete como biblioteca. Una paquete puede utilizarse tanto como aplicación como biblioteca. Si deseamos proporcionar funcionalidad de aplicación y de biblioteca, simplemente indique el punto de entrada en esta propiedad.
6. Instalar las dependencias del proyecto:

   ```
   $ npm install
   ```
7. Mostrar el catálogo de tareas automatizadas del proyecto:

   ```
   $ justo -c
   ```

Creación del módulo de cálculo

Vamos a crear el módulo de cálculo:

1. Crear el archivo lib/calcul.js:

```
/**
 * Suma los valores pasados como argumentos.
 *
 * @param ...args:number[]  Los valores a sumar.
 */
export function sum(...args) {
  var res = 0;
  for (let arg of args) res += arg;
  return res;
}

/**
 * Resta los valores pasados como argumentos.
 *
 * @param ...args:number[]  Los valores a restar.
 */
export function sub(...args) {
  var res = 0;
  for (let arg of args) res -= arg;
  return res;
}
```

2. Editar el archivo index.js y añadir la API del paquete cuando se use como biblioteca:

```
export {sum, sub} from "./lib/calcul";
```

3. Guardar cambios.

Creación del comando a instalar

Ahora, vamos a crear el archivo del comando a instalar, el que realiza los cálculos atendiendo a los argumentos pasados en la línea de comandos:

1. Modificar el archivo bin/calcul.js:

```
#!/usr/bin/env node

"use strict";

const cmd = process.argv[2];
const vals = process.argv.slice(3);
var res = 0;

//(1) op
if (cmd == "sum") {
  for (let val of vals) res += Number(val);
} else if (cmd == "sub") {
  for (let val of vals) res -= Number(val);
} else {
  console.error(`Operación no válida: ${cmd}.`);
  process.exit(1);
}

//(2) mostramos resultado
console.log(res);
```

2. Guardar cambios.
3. Ir a la consola.
4. Generar el paquete final:

```
$ justo build
```

Instalación del paquete

Ahora, vamos a instalar el paquete y a probar la aplicación:

1. Ir a la consola.
2. Instalar el paquete globalmente:


```
$ npm install -g --production dist/es5/nodejs/calcul/
```

3. Comprobar la instalación del paquete:

```
$ npm ls -g calcul
```

4. Realizar unos cálculos de prueba:

```
$ calcul sum 1 3 5 7 9
25
$ calcul sub 1 3 5 7 9
-25
$ calcul div 1 2
Operación no válida: div.
$
```

5. Abrir node en modo interactivo.
6. Importar el paquete calcul:

```
> calcul = require("calcul")
{ sum: [Getter], sub: [Getter] }
>
```

Esto es posible gracias a que hemos definido un módulo de entrada mediante la propiedad main del archivo package.json. Y en él, hemos definido la API del paquete por si se desea utilizar como biblioteca en vez de como aplicación. Esto no es necesario, pero muchos paquetes de NPM presentan esta dualidad.

7. Salir del intérprete de node.
8. Desinstalar el paquete:

```
$ npm uninstall -g calcul
```

