



DESARROLLO DE APLICACIONES WEB AVANZADO

Laboratorio N° 7

Pruebas de Software



DOCENTE
Jorge Castañeda



CURSO
Desarrollo de Aplicaciones Web Avanzado

Laboratorio 7: Pruebas de Software

Objetivos:

Al finalizar el laboratorio el estudiante será capaz de:

- Identificar las principales características de Jest.js
- Importación y uso de librerías principales de Jest.js
- Creación de módulos propios de prueba.
- Exportar sus propios módulos de prueba.

Seguridad:

- Ubicar maletines y/o mochilas en el gabinete del aula de Laboratorio.
- No ingresar con líquidos, ni comida al aula de Laboratorio.
- Al culminar la sesión de laboratorio apagar correctamente la computadora y la pantalla, y ordenar las sillas utilizadas.

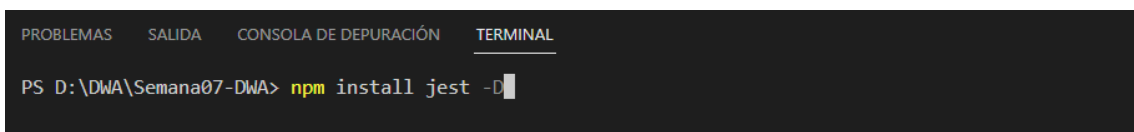
Equipos y Materiales:

- Una computadora con:
 - Windows 7 o superior
 - Conexión a la red del laboratorio
- Instalador de node.js

Procedimiento:

1. Crear un Package.Json

Crear un package.json en un proyecto. Luego instalar la librería jest.js desde npm.



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL
PS D:\DWA\Semana07-DWA> npm install jest -D
```

Configurar el package.json

```
package.json > {} author
1  {
2    "name": "semana07-dwa",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "jest --verbose"
8    },
9    "keywords": [],
10   "author": {
11     "name": "Jorge Castañeda",
12     "email": "jcastaneda@tecsup.edu.pe"
13   },
14   "license": "ISC",
15   "devDependencies": {
16     "jest": "^29.5.0"
17   },
18   "jest": {
19     "testEnvironment": "node"
20   }
21 }
22
```

2. Creación de los métodos de prueba

Crear el archivo `for_testing.js` dentro de la carpeta `utils` y agregar el siguiente código:

```
utils > JS for_testing.js > <unknown>
1  const palindrome = (string) => {
2
3    if(typeof string == 'undefined') return
4
5    return string
6      .split('')
7      .reverse()
8      .join('')
9  }
10
11  const average = array => {
12    let sum = 0;
13    array.forEach(num => { sum += num });
14    return sum / array.length;
15  }
16
17  module.exports = {
18    ...
19    palindrome,
20    average
21  }
```

3. Preparar las pruebas en Jest

Crear una carpeta llamada tests, dentro crear los archivos palindrome.test.js y el archivo average.test.js con el siguiente código:

palindrome.test.js

```
tests > 🚩 palindrome.test.js > 🧪 test('palindrome of undefined') callback > 📄 result
1  const {palindrome} = require ('../utils/for_testing')
2
3  test('palindrome of midudev', () => {
4    const result = palindrome('midudev')
5
6    expect(result).toBe('vedudim')
7  });
8
9  test('palindrome of empty string', () => {
10   //const result = palindrome()
11   const result = palindrome('')
12
13   expect(result).toBe('')
14 })
15
16 test('palindrome of undefined', () => {
17   const result = palindrome()
18
19   expect(result).toBeUndefined()
20 })
```

average.test.js

```
tests > 🚩 average.test.js > ...
1  const {average} = require('../utils/for_testing')
2
3  describe('average', () => {
4    test('of one value is the value itself', () => {
5      expect(average([1])).toBe(1)
6    })
7
8    test('of one value is the value itself', () => {
9      expect(average([1, 2, 3, 4, 5, 6])).toBe(3.5)
10    })
11  })
```

4. Ejecutar el test

En una terminal ejecutar el siguiente código:

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL
PS D:\DWA\Semana07-DWA> npm run test
```

Anota las capturas de pantalla con tus observaciones y resultados:

Creación del package.json:

```
D:\Lab07_Deysilloya>mkdir proyecto07

D:\Lab07_Deysilloya>cd proyecto07

D:\Lab07_Deysilloya\proyecto07>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (proyecto07) labo07-2023-dll
version: (1.0.0) 0.0.1
description: Package.json del lab07
entry point: (index.js)
test command:
git repository:
keywords:
author: Deysi Rubi Lloja Lucero
license: (ISC)
About to write to D:\Lab07_Deysilloya\proyecto07\package.json:
{
  "name": "labo07-2023-dll",
  "version": "0.0.1",
  "description": "Package.json del lab07 ",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Deysi Rubi Lloja Lucero",
  "license": "ISC"
}

Is this OK? (yes) yes
```

Instalación del test:

```
D:\Lab07_Deysilloya\proyecto07>npm set strict-ssl false

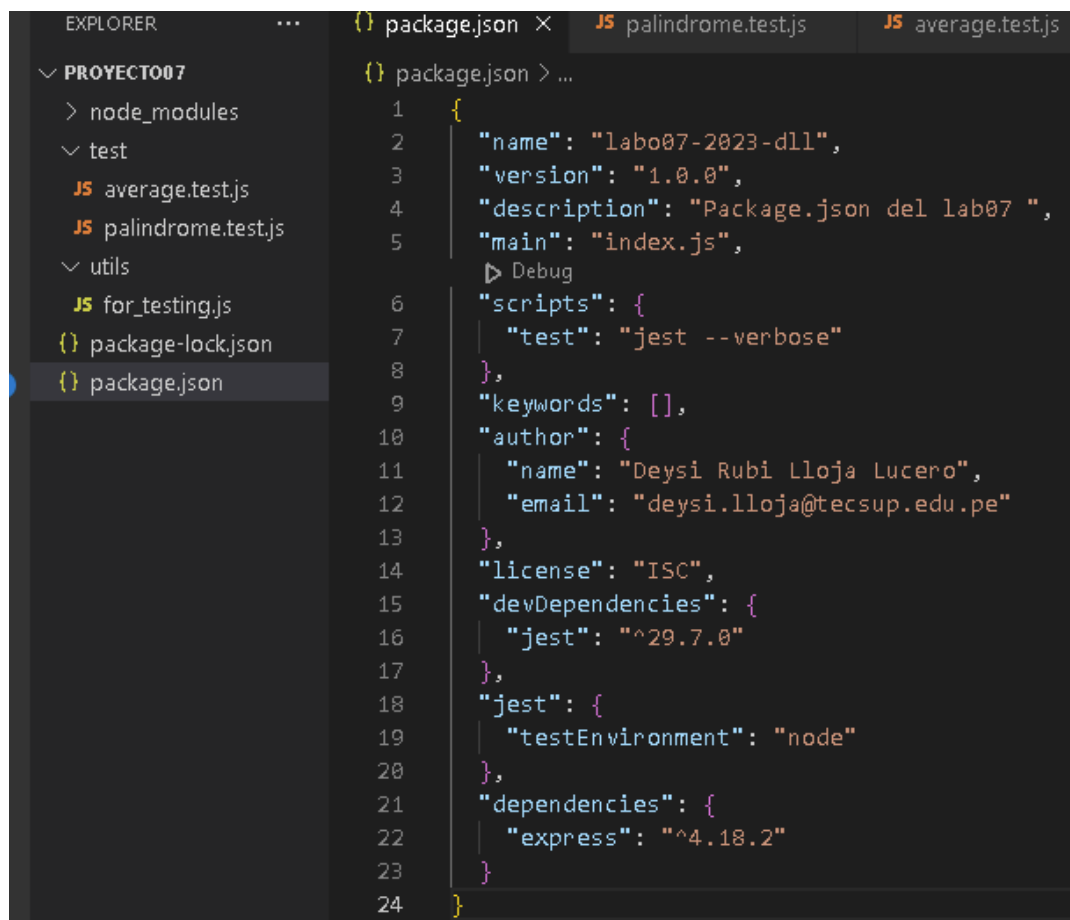
D:\Lab07_Deysilloya\proyecto07>npm install jest D
npm WARN deprecated D@1.0.0: Package no longer supported. Contact support@npmjs.com for more info.

added 290 packages, and audited 291 packages in 23s

31 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Package.json:

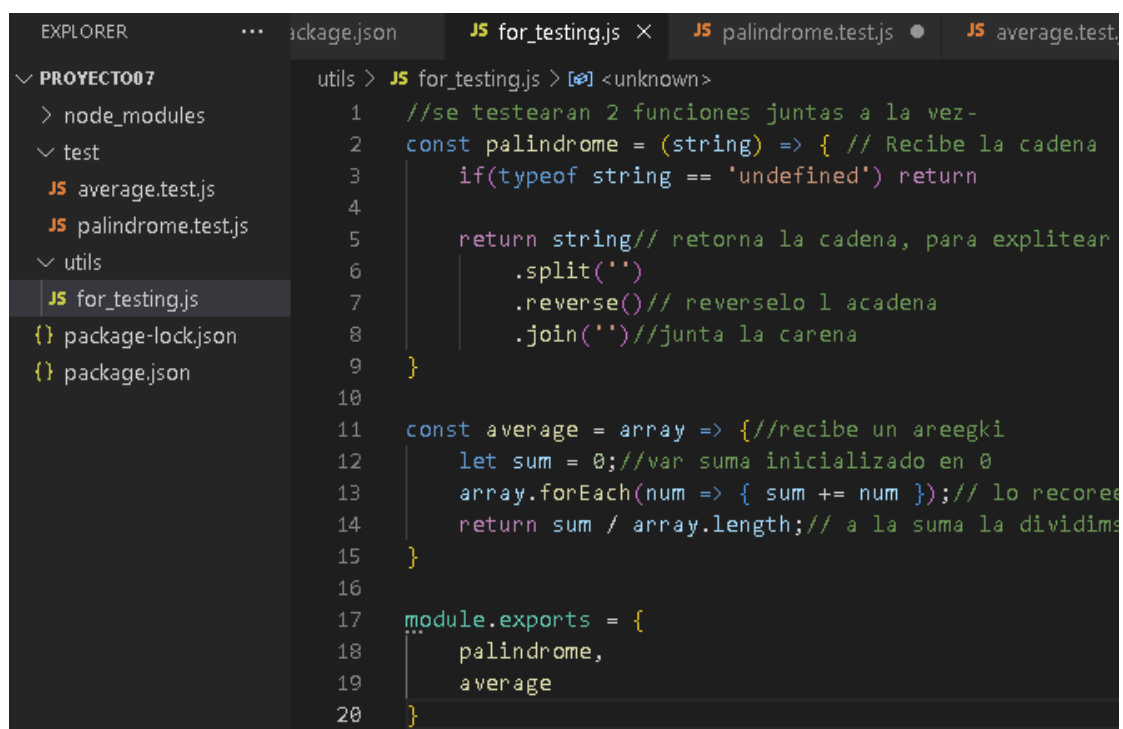


```

1  {
2    "name": "labo07-2023-d11",
3    "version": "1.0.0",
4    "description": "Package.json del lab07 ",
5    "main": "index.js",
6    "scripts": {
7      "test": "jest --verbose"
8    },
9    "keywords": [],
10   "author": {
11     "name": "Deysi Rubi Lloja Lucero",
12     "email": "deysi.lloja@tecsup.edu.pe"
13   },
14   "license": "ISC",
15   "devDependencies": {
16     "jest": "^29.7.0"
17   },
18   "jest": {
19     "testEnvironment": "node"
20   },
21   "dependencies": {
22     "express": "^4.18.2"
23   }
24 }

```

Archivo for_testing.js :



```

1  //se testearan 2 funciones juntas a la vez-
2  const palindrome = (string) => { // Recibe la cadena
3    if(typeof string == 'undefined') return
4
5    return string// retorna la cadena, para explitear
6      .split('')
7      .reverse()// reverselo l acadena
8      .join('')//junta la carena
9  }
10
11  const average = array => { //recibe un areegki
12    let sum = 0; //var suma inicializado en 0
13    array.forEach(num => { sum += num }); // lo recorreo
14    return sum / array.length; // a la suma la dividimo
15  }
16
17  module.exports = {
18    palindrome,
19    average
20  }

```

Archivo test/ palindrme.tes.js :

```

package.json  JS for_testing.js  JS palindrome.test.js ×  JS average.test.js
test > JS palindrome.test.js > ...
  1  //ESPEJO DE UNA CADENA
  2  //voltea el texto
  3  const { palindrome } = require('../utils/for_testing')
  4
  5  test('palindrome of midudev', () => {
  6    const result = palindrome('midudev')
  7
  8    expect(result).toBe('vedudim')
  9  });
 10
 11  test('palindrome of empty string', () => {
 12    //const result = palindrome()
 13    const result = palindrome('')
 14
 15    expect(result).toBe('')
 16  });
 17
 18
 19  test('palindrome of undefined', () => {
 20    const result = palindrome()
 21    // const result = palindrome('')
 22
 23    expect(result).toBe()
 24  });

```

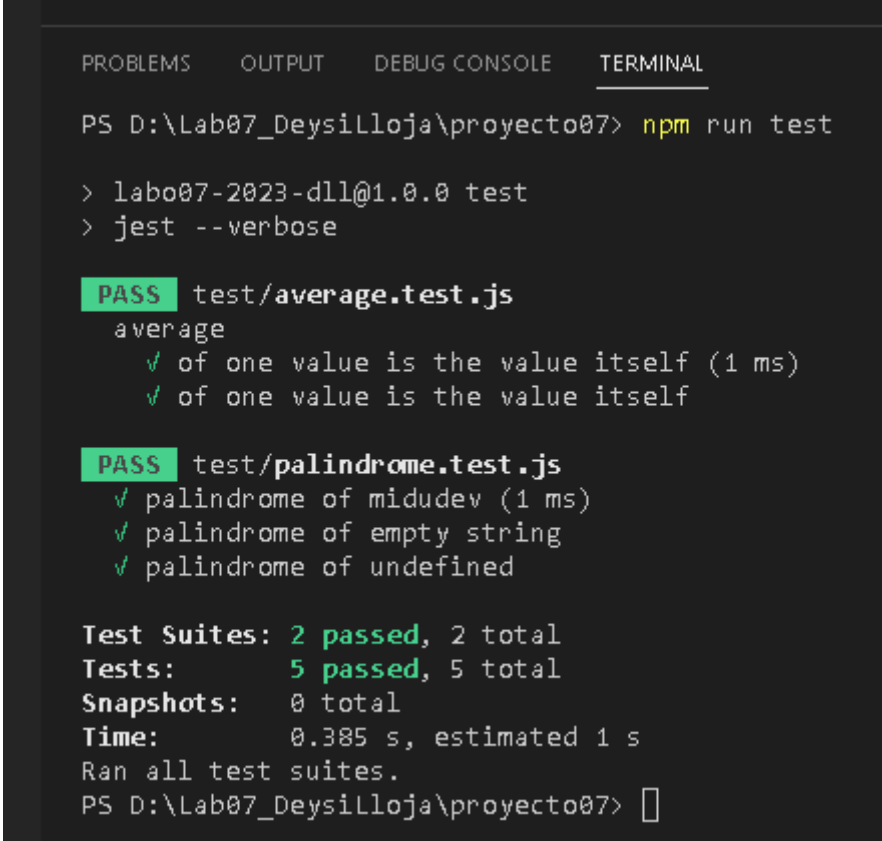
Archivo test/ avergare.test.js :

```

PROYECTO07
├── node_modules
├── test
│   ├── JS average.test.js
│   ├── JS palindrome.test.js
│   └── utils
│       ├── JS for_testing.js
│       ├── {} package-lock.json
│       └── {} package.json
test > JS average.test.js > ...
  1  const { average } = require('../utils/for_testing')
  2  //Se esta importando al archivo for_testing
  3
  4  describe('average', () => {
  5
  6    /*Verifica que cuando se pasa un array con un solo valor [1]
  7    a la función average, el resultado debe ser 1.*/
  8
  9    test('of one value is the value itself', () => {
 10      expect(average([1])).toBe(1)
 11    })
 12
 13    /*Verifica que cuando se pasa un array con varios valores
 14    [1, 2, 3, 4, 5, 6] a la función average, el resultado debe ser 3.5.*/
 15
 16    test('of one value is the value itself', () => {
 17      expect(average([1,2,3,4,5,6])).toBe(3.5)
 18    })
 19  })
 20

```


Resultado:

A screenshot of a terminal window with a dark background. At the top, there are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL', with 'TERMINAL' being the active tab. The terminal shows the following commands and output:

```
PS D:\Lab07_Deysilloja\proyecto07> npm run test
> labo07-2023-dll@1.0.0 test
> jest --verbose

PASS test/average.test.js
  average
    ✓ of one value is the value itself (1 ms)
    ✓ of one value is the value itself

PASS test/palindrome.test.js
  ✓ palindrome of midudev (1 ms)
  ✓ palindrome of empty string
  ✓ palindrome of undefined

Test Suites: 2 passed, 2 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        0.385 s, estimated 1 s
Ran all test suites.
PS D:\Lab07_Deysilloja\proyecto07> 
```

Observaciones:

1. Dentro de este proyecto mediante el comando `npm init`, se ha creado nuestro `package.json` para así poder crear nuestras carpetas y nuestros archivos de tests y poder realizar las pruebas.
2. En caso de que no se pueda instalar el `jest` D. ya que si bien al momento de ejecutar el comando nos sale error, debemos de ejecutar el comando `npm set strict-ssl false` este permitirá desactivar la verificación estricta de SSL en el sistema de gestión de paquetes Node.js (npm).
3. Así mismo dentro del proyecto se ha configurado el archivo `package.js` en la parte del script, ya que se adiciono el "test": "`jest --verbose`", es decir cuando se ejecuta `npm test` en la línea de comandos en el directorio raíz de tu proyecto, se ejecutará Jest con la opción `--verbose`. Jest es una biblioteca de pruebas para JavaScript ampliamente utilizada, y al usar `--verbose`, Jest mostrará información detallada sobre el progreso de las pruebas.
4. Finalmente, se realizaron 2 pruebas, una para explitear una cadena y el segundo test para el cálculo del promedio de los datos de un array.

Notaciones:

Se agregan más funciones de pruebas al archivo `average.test.js`, para así verificar si cumple o no con los requerimientos, verificar si el resultado de la función `average([])` es igual a 0. Es decir para comprobar si la función `average` devuelve 0 cuando se le pasa un array vacío.

```

test > JS average.test.js > describe('average') callback
1  const { average } = require('../utils/for_testing')//
2  //Se esta importando al archivo for_testing
3
4  describe('average', () => { //DENTRO DE ESTE SE DEFINIEN LOS
5
6      test('of many Zero in array is zero', () => {
7          expect(average([0])).toBe(0)
8      })
9
10
11  test('of many Zero in array is zero', () => {
12      expect(average([])).toBe(0)
13  })
14  })
15
16
17

```

Al ejecutar nuestro entorno de testeo mediante el comando `npm run test`, nos imprime error

```

Time:      1.109 s
Ran all test suites.
PS D:\Lab07_Deysilloja\proyecto07> npm run test

> labo07-2023-d11@1.0.0 test
> jest --verbose

FAIL test/average.test.js
  average
    ✓ of many Zero in array is zero (1 ms)
    ✗ of many Zero in array is zero (1 ms)
    ✓ of one value is the value itself
    ✓ of one value is the value itself

  ● average > of many Zero in array is zero

    expect(received).toBe(expected) // Object.is equality

    Expected: 0
    Received: NaN

```

Esto nos indica que el `average` no está manejando correctamente, ya que en este caso se está pasando un array vacío. La prueba espera que el resultado sea 0, pero la función está devolviendo `NaN` (Not-a-Number), lo que indica que el cálculo del promedio no se realizó correctamente en ese caso.

Conclusiones:

Indicar las conclusiones que llegó después de los temas tratados de manera práctica en este laboratorio.

1. Tras haber realizado el presente laboratorio se concluye que se ha logrado identificar y conocer las principales características del jest.js, si bien el jest nos permite realizar pruebas tipo developer de nuestro proyectos y verificar si cumple o no con los requerimientos y las funcionalidades que se requiere, así mismo para trabajar con los test, después de haber creado nuestro proyecto y nuestro respectivo package.json debemos de instalar la librería del test, para así poder realizar nuestros testeos ya que el test facilita el proceso de pruebas y asegura la calidad del código en proyectos JavaScript..
2. En segundo lugar se concluye que se logró realizar la importación e implementar de manera adecuada la librería jest.js, si bien mediante esta librería se podrán realizar las pruebas, su importación está realizada en el archivo package.json, si bien dentro de estos archivos se encuentran todas las librerías que se instalaron, y que se pueden usar dentro del proyecto. Por otro lado el Jest proporciona un conjunto completo de funciones de aserción como expect, que se puede usar para realizar afirmaciones sobre los resultados esperados.
3. En tercer lugar, mediante el manejo de testeo, dentro del presente laboratorio se han creado nuestros propios módulos de pruebas, para ello se creó un archivo for-testing.js, dentro de este archivo se definieron 2 funciones palindrome y average, si bien función palindrome toma una cadena de caracteres como entrada y verifica si es un palíndromo invirtiendo la cadena y comparándola con la versión original. Mientras que la función average calcula el promedio de un array de números sumando todos los elementos del array y dividiendo la suma por la cantidad de elementos. Por otro lado mediante el module.exports se exportarán los módulos de los que se realizará sus pruebas, para ello se crearon 2 archivos de test.js con su código respectivo, para verificar así si cumple o no.
4. Finalmente, se concluye que se ha logrado exportar los módulos y realizar nuestros test, mediante el comando npm run test de ambas funciones y proyectos implementados. Por otro lado se identificaron algunas funciones, que resultaron nuevas para nosotros, por ejemplo la función toBe() se utiliza para comprobar si el resultado esperado de una operación coincide con el valor que se pasa como argumento y de esta manera que al momento de realizar la prueba se verifique si cumple o no , y mediante el test('palindrome of undefined', () => {...}), esta prueba comprueba si la función palindrome se comporta correctamente cuando se le pasa undefined como argumento.