Node y el gestor de paquetes

Paquetes NPM para el procesamiento de datos

{desafío} latam_

Utilizar el gestor NPM para la administración de dependencia y paquetes reconociendo el rol que tiene dentro del ciclo de vida de un proyecto Node.js

- Unidad 1:
  Introducción a Node
- Unidad 2:
  Node y el gestor de paquetes
- Unidad 3:
  Persistencia

Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Utilizar el gestor NPM para la instalación de paquetes en el entorno Node configurando sus versiones.*
- *Utilizar un módulo dentro de un programa Node mediante sentencias de importación para el uso de sus funcionalidades.*

{desafío} latam_

¿Alguna vez te has preguntado cómo los desarrolladores de software manejan y procesan grandes cantidades de datos en sus aplicaciones?

{desafío} latam_

/* Moment */

Moment

Moment.js simplifica considerablemente las tareas relacionadas con fechas y tiempos al proporcionar una serie de métodos y funcionalidades que facilitan la manipulación, el análisis y el formateo de fechas y horas en diferentes formatos.

Su instalación se da con el siguiente comando:

```
npm i moment
```

En el **sitio oficial de Moment.js** encontrarás un montón de propiedades y métodos con los que podrías estar todo el día jugando

Moment

Características

Análisis de fechas

dd/mm/aaaa

Formateo de fechas

Operaciones matemáticas

Comparaciones de fechas

Manipulación de zonas horarias

Cálculos de duración

Soporte para internacionalización

Demostración "Aplicando Moment.js"

{desafío} latam_

Ejercicio guiado

Aplicando Moment.js

Crear una mini aplicación que imprima por consola la ejecución del método moment, esto devolverá un formato de fecha correspondiente al momento en el que es ejecutada la función, de esta manera podrás tener un primer encuentro con este paquete.

- **Paso 1:** Importar el paquete moment en una constante.

```
const moment = require('moment')
```

Ejercicio guiado

Aplicando Moment.js

- **Paso 2:** La importación del paquete en sí es un método, así que envía su llamado directamente por consola.

```
console.log(moment())
```

Ahora corre la aplicación y deberás recibir algo como lo que se muestra en la siguiente imagen.

```
→  ejercicio lectura git:(master) ✗ node index.js
Moment<2020-09-15T01:43:12-03:00>
```

Sustituye el console.log que hiciste anteriormente con la siguiente línea de código

```
console.log(moment().subtract(10,
'days').format('MMM Do YY'))
```

¿Qué muestra?

{desafío} latam_

Demostración
"¿Qué día será en el futuro?"

Ejercicio guiado

¿Qué día será en el futuro?

Desarrollar una mini aplicación que devuelva por consola la fecha actual sumandole 10.000 días, además supondremos que estas son consultas que serán registradas como un historial por lo que deberán tener un identificador único.

- **Paso 1:** Importar los paquetes chalk, uuid y moment.
- **Paso 2:** Crear un objeto “consulta”
- **Paso 3:** Almacenar en la propiedad “fecha” la fecha actual sumada 10.000 días
- **Paso 4:** Generar un identificador único universal
- **Paso 5:** Imprimir por consola el objeto “consulta”

```
→  ejercicio lectura git:(master) ✗ node index.js
{ fecha: 'Feb 1st 48', ID: '6605edc9-1d62-46fe-9d14-f0f669adf0af' }
```

{desafío} latam_

```
// Paso 1
const chalk = require('chalk')
// Para versiones recientes
import chalk from 'chalk'
const { v4: uuidv4 } = require('uuid')
const moment = require('moment')
// Paso 2
const consulta = {
 // Paso 3
 fecha: moment().add(10000, 'days').format('MMM Do YY'),
 // Paso 4
 ID: uuidv4(),
}
// Paso 5
console.log(consulta)
```

/* Lodash */

Lodash

¿Qué es?

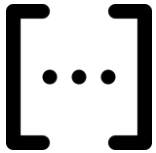
Es una biblioteca de utilidades de JavaScript que proporciona funciones y métodos de utilidad para **realizar operaciones comunes de programación de manera más eficiente y sencilla**. Fue diseñada para simplificar tareas repetitivas y complejas que a menudo surgen durante el desarrollo de aplicaciones, como la manipulación de arrays, objetos, strings y otros tipos de datos.

Para instalar lodash deberás ocupar el siguiente comando:

```
npm i lodash
```

Lodash

Características



Manipulación de arrays

Manipulación de objetos

Trabajo con strings

Operaciones de colecciones

Manejo de funciones

Validación y transformación de datos

Flujo de control

Demostración "Pares o nones"

Ejercicio guiado

Pares o nones

Desarrollar una aplicación que divida un arreglo de números en 2 arreglos, separando los números pares e impares. Sigue los siguientes pasos para el desarrollo de este ejercicio:

- **Paso 1:** Importar el paquete lodash en una constante. Esta librería tiene una peculiaridad que es ser usada como un guión bajo, es decir la constante que crees declararla con un "_"

```
/const _ = require('lodash')
```

Ejercicio guiado

Pares o nones

- **Paso 2:** Crear un arreglo de números en secuencia del 1 al 6.

```
const numeros = [1, 2, 3, 4, 5, 6]
```

- **Paso 3:** Usar el método llamado “partition” que recibe como primer parámetro el arreglo de números y como segundo parámetro tendrás el callback de este método, entonces se debe escribir una función de flecha que recibe un parámetro identificando cada uno de los elementos del arreglo y retorne los que cumplan con la siguiente condición “n % 2”

```
console.log(_.partition(numeros, (n) => n % 2))
```

Ejercicio guiado

Pares o nones

¿Cómo se vería si no se utiliza Lodash?

```
const numeros = [1, 2, 3, 4, 5, 6]
let pares = []
let impares = []
for (let index = 0; index < numeros.length; index++) {
 if (numeros[index] % 2) {
   impares.push(numeros[index])
 } else {
   pares.push(numeros[index])
 }
}
let arregloFinal = [impares, pares]
console.log(arregloFinal)
```

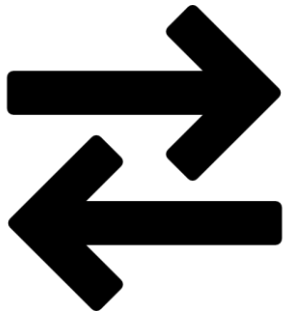
/* Axios */

Axios

Axios es una popular biblioteca de JavaScript que se utiliza para realizar solicitudes HTTP desde aplicaciones web. Se utiliza comúnmente en aplicaciones frontend y backend para realizar peticiones a servidores web y obtener datos. Axios simplifica la comunicación entre el cliente y el servidor al proporcionar una interfaz fácil de usar para enviar solicitudes y recibir respuestas.

Su instalación se consigue a través del siguiente comando

```
npm install axios
```



Axios

- La sintaxis básica para usar esta librería es

```
axios
.<verbo http>( <url a consultar>)
.then( [callback con la data de la consulta] )
.catch( [callback con el error de la consulta] )
```

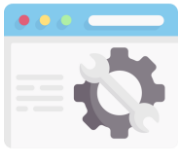
Axios

Características

Sintaxis clara y sencilla

Soporte para promesas

Interceptores de solicitud y respuesta



Soporte para navegadores y Node.js

Manejo automático de la serialización y deserialización

Cancelación de solicitudes

Demostración "Aplicando Axios"

Desafío guiado

Aplicando Axios

Consumir la API de Rick and Morty en el siguiente endpoint https://rickandmortyapi.com/api/character/1, el objetivo será imprimir en consola el nombre de este personaje. Instala el paquete NPM en un proyecto nuevo y sigue los siguientes pasos para el desarrollo de este ejercicio.

- **Paso 1:** Importar el paquete “axios” en una constante

```
const axios = require("axios");
```

Desafío guiado

Aplicando Axios

- **Paso 2:** Ocupar la instancia de “axios” y su método “get” pasando como argumento la URL del endpoint a consultar

```
axios
  .get("https://rickandmortyapi.com/api/character/1")
  .then((data) => {
```

Desafío guiado

Aplicando Axios

- **Paso 3:** El método usado en el paso anterior devuelve una promesa, así que en el callback del método “then” recibirás el objeto “data”. En esta API el nombre del personaje se encuentra en el primer nivel del objeto de la data.
- Guardar en una constante el nombre del personaje usando la siguiente especificidad **“data.data.name”**, siendo la primera “data” el objeto de axios, la segunda “data” el objeto correspondiente al personaje y por supuesto la propiedad “name” que devuelve el String del personaje que estamos consultando

```
    const name = data.data.name;
    console.log(name);
})
```

{desafío} latam_

Desafío guiado

Aplicando Axios

- **Paso 4:** En el catch capturar e imprimir un error de consulta en caso de existir.

```
  .catch((e) => {
    console.log(e);
  });
```

Ahora levanta tu aplicación en la terminal y deberás obtener lo siguiente

```
$ node index.js
Rick Sanchez
```

Resumen

- Existen librerías que facilitan el procesamiento de datos en JavaScript, evitando la complejidad del trabajo con código propio.
- Moment es una popular librería para el manejo de fechas en JavaScript
- Lodash es una librería que facilita el trabajo con diversos tipos de datos como arreglos, números, objetos, etc.
- Axios es una librería popular para el consumo de APIs REST en JavaScript.

Después de aprender sobre las librerías Moment, Lodash y Axios, ¿cuáles son los principales objetivos de estas librerías y qué beneficios ofrecen al desarrollador en términos de procesamiento de datos y consumo de APIs?

{desafío} latam_

Próxima sesión...

- *Utilizar el gestor NPM para la instalación de paquetes en el entorno Node configurando sus versiones.*
- *Utilizar un módulo dentro de un programa Node mediante sentencias de importación para el uso de sus funcionalidades.*
- *Ejecutar procedimiento para bajar una aplicación Node utilizando la línea de comandos.*

{desafío} latam_

{desafío}
latam_

Academia de
talentos digitales