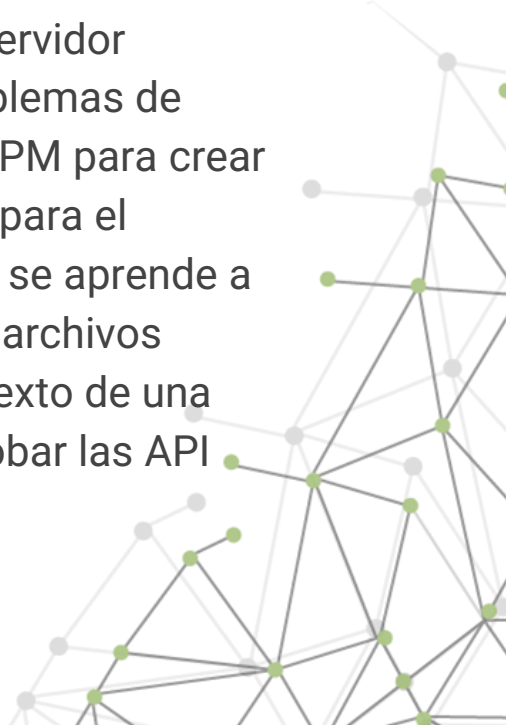
Introducción a Node

Node y Express (Parte I)

{desafío} latam_

¿Qué aprenderemos en este módulo?

El módulo enseña a crear aplicaciones con JavaScript en el lado del servidor utilizando Node, accediendo al sistema de archivos y resolviendo problemas de persistencia y gestión de datos. Se introduce el Gestor de Paquetes NPM para crear y gestionar proyectos, importar módulos y utilizar paquetes comunes para el desarrollo de Node, especialmente en consultas asíncronas. También se aprende a crear y consultar servicios REST para manejar datos almacenados en archivos JSON en el servidor, considerando errores y estados HTTP, en el contexto de una arquitectura cliente-servidor. Además, se aborda el testing y cómo probar las API REST.



{desafío} latam_

Describir las características fundamentales del entorno Node.js y su utilidad para el desarrollo de aplicaciones web

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

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconocer las características básicas del entorno Node Express identificando sus componentes para el desarrollo de aplicaciones web.*
- *Describir el uso del entorno Node Express y las problemáticas que son posibles de resolver para el desarrollo de aplicaciones web.*
- *Reconocer las ventajas y potencialidades de Express para el desarrollo de aplicaciones empresariales bajo el entorno Node.js.*
- *Distinguir la aplicación de tareas con Node puro versus Express integrado para el desarrollo de aplicaciones en el entorno Node.js.*
- *Reconocer el procedimiento de Incorporación de Express como integrador de un proyecto Web acorde al entorno Node.js.*

{desafío} latam_

/* ¿Qué es node? */

¿Qué es Node?

Node, es una tecnología que nos provee de un entorno para la ejecución de código JavaScript fuera del navegador, quiere decir, que permite ejecutar código JavaScript como si se tratara de un programa instalado en nuestro computador o una aplicación de servidor.

Node trabaja con el motor de Google Chrome llamado V8, y este es el motor intérprete que se encarga de tomar ese código JavaScript y transformarlo en un lenguaje conocido para la máquina en la cual se requiere ejecutar.

Características
de Node

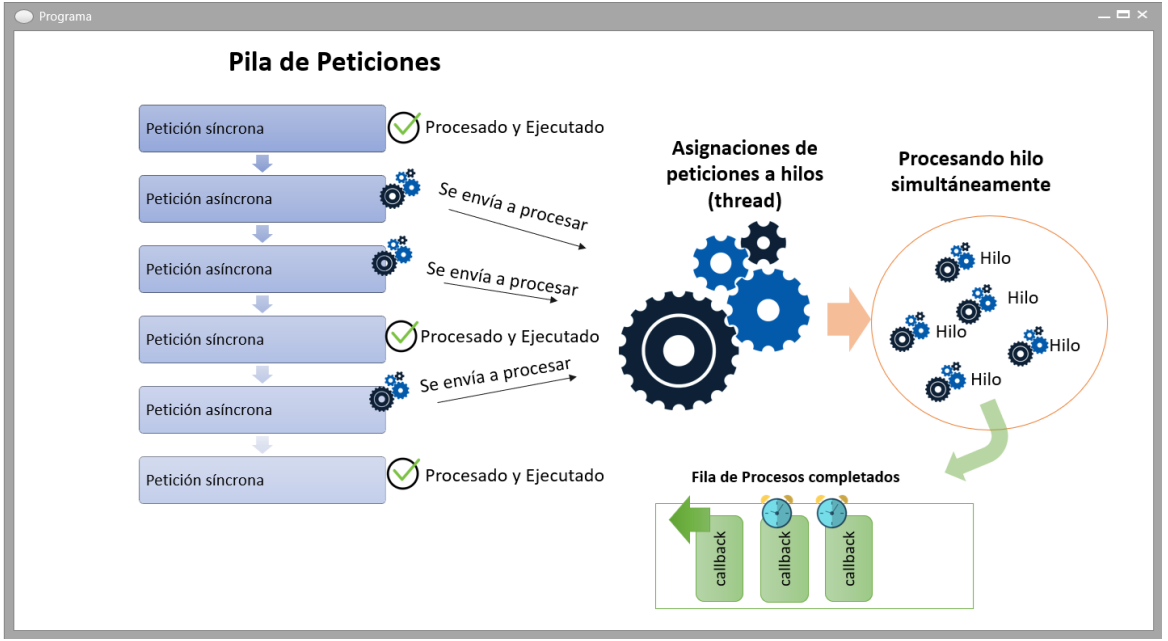
- Node sigue un proceso por cada tarea que procesa, esto se conoce como el “ciclo de vida de un proceso”

- Una de las características que más destacan en Node es su velocidad y rendimiento en los procesos que realiza, esto se logra al manejar sus operaciones de manera asíncrona, es decir, permite responder rápidamente a una serie de peticiones o llamadas simultáneas, optimizando el rendimiento en sus procesos.

Características

Ciclo de vida de un proceso Node

Node genera una pila de peticiones tanto síncronas como asíncronas y las va evaluando a medida que van llegando. Esta lista de peticiones apiladas una debajo de otra también se conoce como el Single Thread



{desafío} latam_

Características

Ventajas

1

Velocidad y rendimiento

Gracias a su integración con el motor V8 de Chrome, Node aprovecha todas las características de optimización que V8 posee para el tratamiento veloz y rápido del código Javascript.

2

Basado en el lenguaje JavaScript

JavaScript, es el lenguaje de la web, y esta popularidad en parte se debe al nacimiento de Node. Esto nos da la posibilidad de realizar desarrollo con un mismo lenguaje tanto para el Backend y Frontend.

3

Popularidad

Como mencionamos, Node y JavaScript gozan de gran popularidad, cada día crece más el interés y la cantidad de desarrolladores que lo utilizan. Esto permitirá que Node crezca y perdure en el tiempo.

4

Multiplataforma

Puede ejecutarse en máquinas Windows, Mac OS X y Unix.

5

Manejo de alto tráfico

Es ideal para aplicaciones de alto tráfico como Twitter, WhatsApp, etc.

Características

Desventajas

1 — 2 — 3

Inestabilidad de la API

La API cambia frecuentemente y no permanece estable. Una API que cambia frecuentemente, trae cambios incompatibles para versiones anteriores por lo que te obliga a mantenerte atento a estos cambios para hacer las actualizaciones correspondientes.

Problemas heredados

Node hereda los problemas de JavaScript, como por ejemplo la representación del float.

Potencia de cálculo

No ofrece mayor potencialidad para cálculos que las ofrecidas por otros lenguajes de servidor como C# y Java, aunque esta desventaja sólo aplica a proyectos de procesamiento matemático altamente complejos.

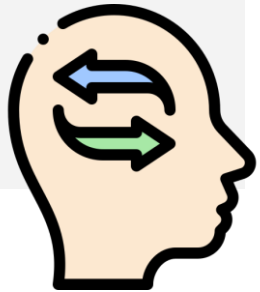
Características
Especiales

Gestor de paquetes (NPM)

Posee un gestor de paquetes que nos permite instalar nuevas funcionalidades o módulos, y un manejo de dependencias por separado sin necesidad de una herramienta externa.

Cambio de mentalidad

Requiere un cambio de mentalidad en relación a los lenguajes tradicionales del servidor, debido a la utilización de la asincronía en su programación, lo que conlleva a que la curva de aprendizaje no sea tan rápida en comparación con otros entornos.



Características

Problemas o necesidades que podrías resolver

- Crear servidores
- Crear y consumir APIs REST
- Conexiones con bases de datos
- Aplicaciones multiplataformas
- Chats
- Conexiones con electrónica
- Gestión de archivos
- Testing de aplicaciones
- Ejecución de tareas simultáneas
- Algoritmos de inteligencia artificial

/* El motor V8 de Google */

El motor V8 de Google

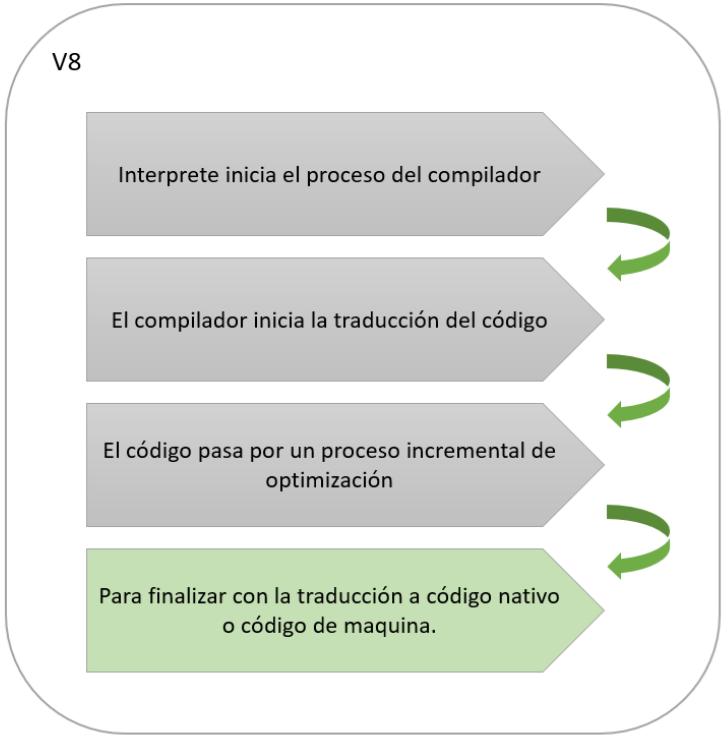
V8 de Chrome, es un proyecto de código abierto desarrollado en el lenguaje C++ que compila código Javascript o WebAssembly en código de máquina. Esto quiere decir, que traduce de lenguaje JavaScript a lenguaje reconocido por la computadora, lenguaje de máquina o bien llamado código nativo del sistema operativo sobre el cual se ejecutará nuestra implementación.

El motor V8 de Google

¿Cómo funciona?

El motor V8 toma el código Javascript y genera esta serie de pasos que permitirán la traducción del código JavaScript en código reconocido por la máquina (nuestra computadora o servidor).

```
<script>
function hola()
{
 alert('hola mundo');
}
</script>
```



/* ¿Qué es Express? */

¿Qué es Express?

Express es un famoso framework escrito en JavaScript, utilizado por muchas empresas de reconocimiento mundial y recomendado por expertos en la materia. Desde el lado del servidor, aparece en el mundo del desarrollo con el objetivo de facilitar y agilizar la creación de aplicaciones web, trabajando en el backend y creando servidores, middlewares, enrutadores, capas de seguridad, API REST para consumo de aplicaciones Front-end, junto con gestionar contenido de forma dinámica, comunicaciones con bases de datos, entre otras cosas.

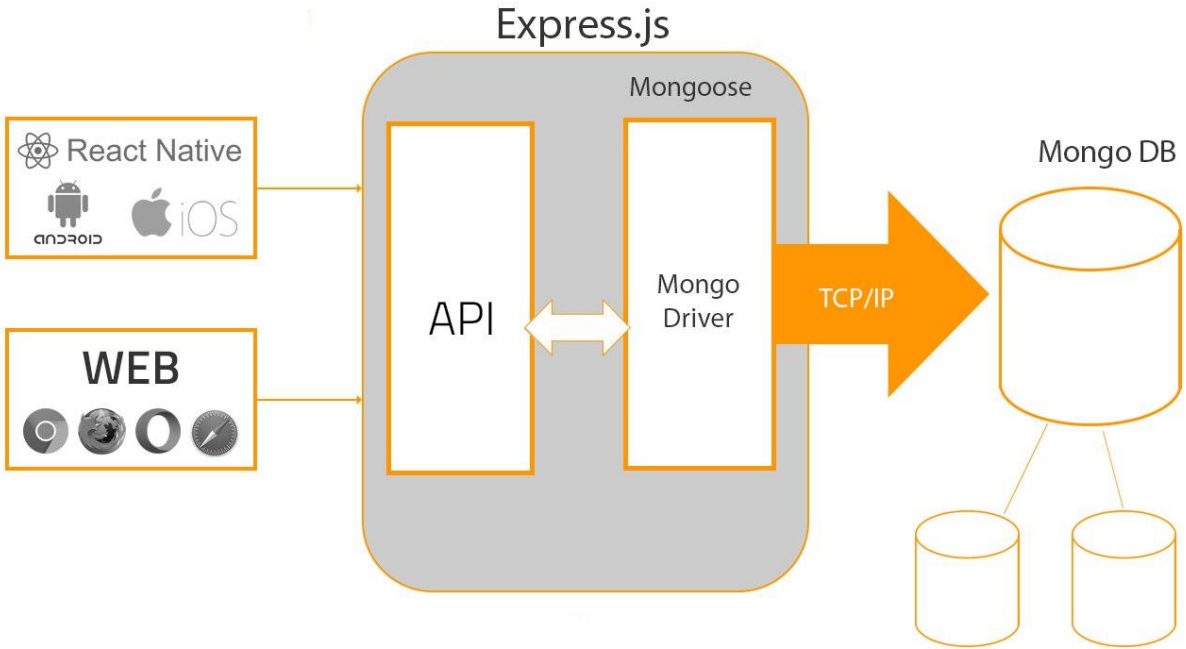
¿Por qué aprender Express?

Los frameworks son creados con la principal idea de facilitar el desarrollo de aplicaciones, conteniendo miles de líneas de código con funciones y algoritmos complejos que ofrecen por medio de un simple llamado, soluciones rápidas a problemas complejos, como pueden ser las siguientes:

- La creación de servidores.
- Configuración de rutas.
- Creación y aplicación de middlewares.
- Seguridad de la data que viaja por protocolo HTTP.
- Creación de API REST.
- Manejo de errores devueltos por el servidor.
- Manejo de sesiones de usuarios, entre otros.

¿Por qué aprender Express?

En la siguiente imagen se aprecia que Express es una de las piezas fundamentales dentro de una aplicación, ya que sostiene la infraestructura para conectar las fuentes de datos, con la lógica de negocio y sus clientes móviles o Web.



Express y su origen

La página web oficial lo define de la siguiente manera:

"Express es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto sólido de características para las aplicaciones web y móviles." *Fuente: Express js*

Características

Infraestructura web rápida	Minimalista	Flexible	Servidores	Entorno de Node
El rendimiento de Express es envidiable, porque al final se traduce en puro código JavaScript y la velocidad de interpretación/compilación destaca en comparación con sus homólogos.	Permite con simples llamados de sus métodos obtener funciones complejas, sin necesidad de conocer sus procesos internos.	Permite configurar su propia estructura a voluntad, definiendo tu propia estructura de desarrollo basada en Express..	La creación de un servidor en Express con solo 3 instrucciones toma menos de 1 minuto.	Ya que su creación fue hecha bajo este entorno, se pueden aprovechar todas las potencialidades de Node para sumar a su versatilidad. .

Características

Creación de APIs REST	Renderización de contenido dinámicos	Documentación	Comunidad
De los usos más comunes, es su enrutamiento interno compatible con el protocolo HTTP y sus métodos, además de middlewares que ofrecen a tiempo de leopardo la creación de APIs y APIs REST.	La compatibilidad con motores de plantillas como Pug, Ejs, Handlebars, entre otros.	En su página oficial se puede encontrar la documentación escrita por su equipo de desarrollo, la cual está disponible en varios idiomas.	Para comienzos del año 2020, solo en NPM se registran más de 12 millones de descargas, sin contar con sus subpaquetes creados por desarrolladores en todo el planeta.

/* Creación de servidores con Node puro y Express */

Instalación de Express

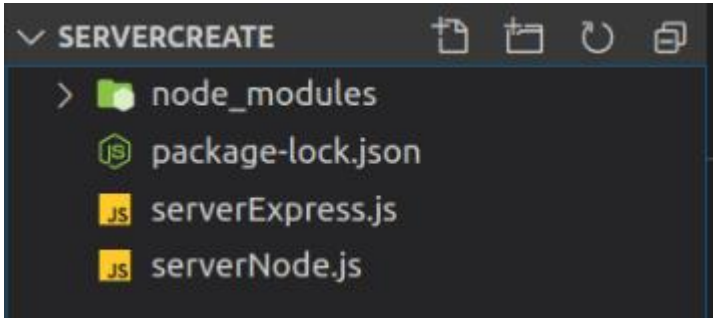
Lo primero, será instalar Express a través de NPM, así que crea una carpeta y ábrela en tu editor de código favorito. En esta muestra, usaré Visual Studio Code aprovechando su terminal integrada. Ocupa la siguiente línea de comandos para instalar Express.

```
npm install express --save
```

Con la carpeta creada y nuestro framework instalado, procederemos a crear 2 archivos, el primero llamado “serverExpress.js” y el segundo “serverNode.js”, ¿Cuál será el objetivo de esto? Ver la diferencia entre crear un servidor con Node puro y llegar a lo mismo usando Express.

Instalación de Express

En la siguiente imagen puedes ver como debería estar tu árbol de archivos actualmente.



Considerando que en el mundo de la tecnología todo puede cambiar en cualquier momento y tenemos actualizaciones consecutivamente, te recomiendo que visites siempre la página oficial de Express.

Creación de servidores

El código para iniciar un servidor con Node puro y una ruta inicial es el siguiente:

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

http
  .createServer((req, res) => {
    if (req.url == "/Inicio" && req.method == "GET") {
      res.end("Hola mundo! Servidor con Node js puro");
    }
  })
  .listen(3000, () => {
    console.log("El servidor está inicializado en el puerto
3000");
  });
```

{desafío} latam_

Creación de servidores

Ahora, veamos el código para iniciar un servidor con la misma ruta GET /Inicio con Express.

```
const express = require('express')
const app = express()

app.listen(3000, () => {
 console.log('El servidor está inicializado en el puerto 3000')
})

app.get("/Inicio", (req, res) => {
    res.send("Hola mundo! Servidor con Express js &#128526;")
})
```

Creación de servidores

Ahora que tenemos ambos archivos escritos, levantemos los servidores empezando con Node. Para esto, ocupa la siguiente línea de comando en la terminal:

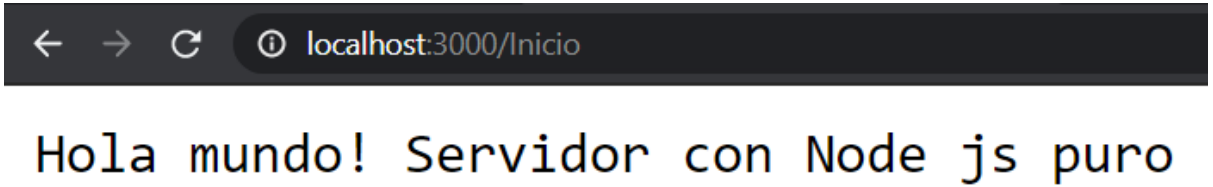
```
node serverNode.js
```

Deberás ver el mensaje que te mostramos en la siguiente imagen por consola:

```
$ node serverNode.js
El servidor está inicializado en el puerto 3000
```

Creación de servidores

Ahora, abre tu navegador y consulta la siguiente dirección: http://localhost:3000/Inicio y deberás ver lo que te mostramos en la siguiente imagen:



Ahora es el turno de Express, cancela la terminal donde levantaste el servidor con Node puro y ocupa la siguiente línea de comando para levantar el servidor:

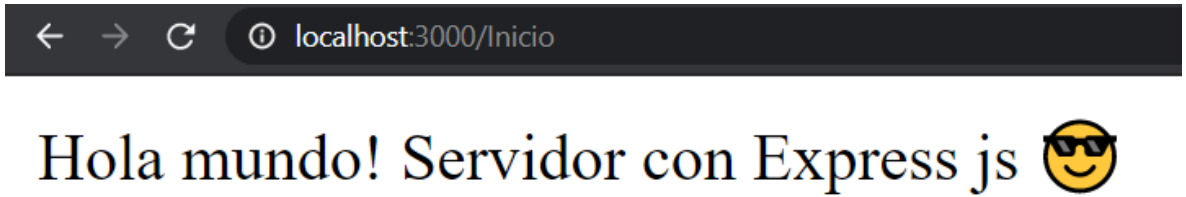
```
node serverExpress.js
```

Creación de servidores

Deberás ver el mensaje que te mostramos en la siguiente imagen por consola. Notarás que es el mismo que obtuvimos al levantar el servidor con Node:

```
$ node serverExpress.js
El servidor está inicializado en el puerto 3000
```

Dirígete nuevamente a tu navegador y consulta la misma dirección http://localhost:3000/Inicio, deberás ver lo que te mostramos en la siguiente imagen:



/* Consideraciones para el desarrollo inicial con Express */

Consideraciones para el desarrollo inicial con Express

- Si intentas abrir el servidor en el puerto 3000 desde el navegador, obtendrás la siguiente respuesta: “Cannot GET /”, no te debes preocupar, esta es solo la forma de Express para decirte que aun no se ha configurado la respuesta para una consulta a la ruta raíz.

- Otra consideración importante, es la instancia del paquete Express en el código.Como puedes notar, es almacenada en una constante llamada “app”, y su método “listen” es invocado pasándole como parámetro el puerto y la función callback.

¿Cuáles son algunas ventajas y desventajas de utilizar Node en el desarrollo web?

¿Cuáles son las principales características y ventajas de utilizar Express como framework de backend en el desarrollo de aplicaciones web?

{desafío} latam_

Próxima sesión...

- *Aplicar procedimiento de Instalación del entorno Node Express para la creación de un programa básico.*

- *Aplicar procedimiento de creación de un programa Node Express simple para verificar la instalación del entorno.*

{desafío} latam_

{desafío}
latam_

Academia de
talentos digitales