

Centro de e-Learning SCEU UTN - BA. Medrano 951 2do piso

(1179) // Tel. +54 11 7078- 8073 / Fax +54 11 4032 0148

www.sceu.frba.utn.edu.ar/e-learning

Curso:

Professional Webmaster

Módulo 4:

Aplicaciones web con Node.js

Unidad 1:

Introducción



Presentación

En esta unidad comenzamos a trabajar con el framework NodeJS para la creación de sitios web.



Objetivos

Que los participantes logren...

- Comprender el ecosistema Node.js/JavaScript
- Entender y utilizar los distintos paquetes de npm .
- Conocer y utilizar el framework Express.js



Bloques temáticos

1. Conceptos básicos de Node.js.
2. Manejo de paquetes de npm.
3. Introducción a Express.

1. Conceptos básicos de Node.js

Node.js es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S (Entrada/Salida) sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema más grande de librerías de código abierto en el mundo.

Fue concebido como un entorno de ejecución de JavaScript orientado a eventos asíncronos y está diseñado para construir aplicaciones en red escalables.

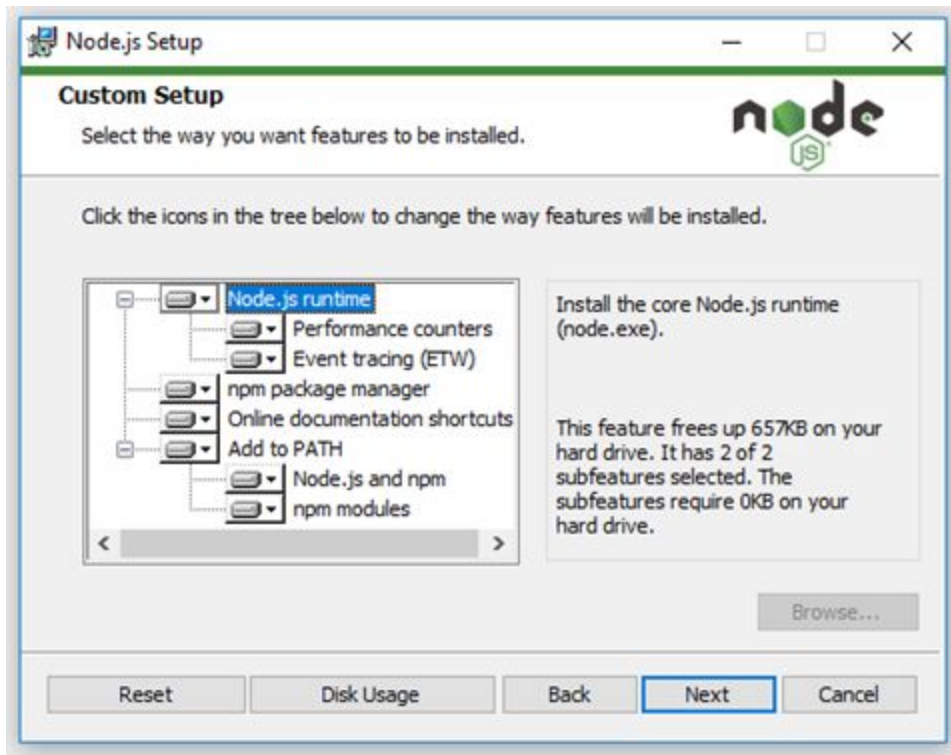
Node permite además la creación de sitios y aplicaciones web contruidos enteramente usando JavaScript, así como también la creación y utilización de un sin fin de herramientas.

Instalación

Node.js (de aquí en adelante Node) se puede descargar desde su sitio web en <https://nodejs.org/es/>. La página principal nos ofrece 2 versiones, la LTS y la Actual. La principal diferencia entre estas es que la LTS (Long Term Service) va a recibir actualizaciones y parches de seguridad durante mucho más tiempo. En cambio la versión denominada "Actual" es la que incluye las últimas características del entorno.

Salvo que vayamos a instalar Node en un servidor para correr una aplicación web grande, en el uso diario se recomienda descargar e instalar la versión más nueva.

Durante la instalación es importante que tengamos habilitadas todas las opciones que se muestran a continuación



Una vez terminada la instalación podemos verificar que todo haya salido correctamente abriendo la consola (CMD) y escribiendo `node -v`. Este comando devuelve la versión instalada de Node, al momento de escribir esto es la **v12.19.0**.

Si seguimos los pasos de la instalación deberíamos tener también instalado **npm** (Node Package Manager) que es el manejador de paquetes de Node, y es la herramienta que nos permitirá instalar todas las librerías de Javascript que necesitemos en nuestros proyectos.

Podemos verificar la instalación de npm escribiendo `npm -v` en la consola.

Uso básico de Node

En su forma más simple, podemos abrir la consola y escribir solamente `node`. Esto abrirá una instancia que nos permitirá ir ingresando expresiones de Javascript que serán evaluadas e impresas en el momento.

Algunos ejemplos de cosas que podemos probar:

- Operaciones matemáticas
- Asignaciones de variables
- Definición de funciones
- Y cualquier otro código Javascript válido

Para salir de la instancia de node escribimos **.exit** (con punto antes de la e) o presionamos la combinación de teclas CTRL+C.

También podemos ejecutar scripts directamente con Node. Creamos un archivo nuevo llamado index.js y escribimos el siguiente código:

```
var hoy = new Date( )

console.log('Hoy es ' + hoy)

var i
for(i = 0; i < 10; i++) {
  console.log(i)
}
```

Lo guardamos y lo ejecutamos escribiendo **node index.js**. El resultado debería ser la fecha y hora actual y un conteo de 0 a 9 como respuesta en la consola.

2. Manejo de paquetes de npm

Inicialización de un proyecto con Node.js

Para comenzar un proyecto con Node.js desde cero es necesario correr el script de inicialización `npm init`. Este comando debe ser corrido desde la consola, en la carpeta donde deseemos inicializar el proyecto.

El script nos hará una serie de preguntas básicas sobre nuestro proyecto. Podemos contestar todas apretando la tecla ENTER o llenar los datos que creamos necesarios.

Al finalizar, encontraremos un archivo llamado `package.json` donde se guardó toda la información que acabamos de cargar. Así mismo, en ese archivo, es donde el manejador de paquetes de Node, npm, va a ir guardando las dependencias o librerías de nuestro proyecto, así como también, varias de las herramientas que usaremos para desarrollarlo.

Instalación y uso de librerías con npm

La instalación de librerías se hace mediante el comando `npm install`. A este debemos indicarle qué librería deseamos que baje e instale y este lo hará de forma automática. Las librerías las podemos buscar en el sitio <https://www.npmjs.com/>, que es el repositorio central de npm.

A modo de ejemplo, vamos a instalar una librería llamada `moment` (<http://momentjs.com/>). Su página de npm es <https://www.npmjs.com/package/moment>, donde encontraremos información de la misma.

Ejecutamos el siguiente comando `npm install moment --save`. El argumento `--save` le indica a npm que deseamos que guarde la referencia a la librería como una dependencia en el archivo `package.json`. Una vez terminada la instalación podemos abrir este archivo para verificarlo.

3. Introducción a Express

Es el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web populares de Node.

Proporciona mecanismos para:

- Escritura de manejadores de peticiones con **diferentes verbos HTTP** en diferentes caminos URL (rutas).
- Integración con motores de **renderización de "vistas"** para generar respuestas mediante la introducción de **datos en plantillas**.
- Añadir procesamiento de peticiones **"middleware"** adicional en cualquier punto dentro de la ejecución de una petición.

A pesar de que Express es en sí mismo bastante minimalista, los desarrolladores han creado librerías para trabajar con:

- Envío de mail: **nodemailer**
- Sesiones: **express-session**
- Base de datos: **mysql**


Instalación

Paso 1

Suponiendo que ya tenemos instalado **Node.js**, creamos un directorio para que contenga la aplicación y convertirlo en el directorio de trabajo.

Paso 2

Vamos a ejecutar una herramienta llamada `express-generator`. Esta se encarga de generar la plantilla básica de un proyecto de Express. Mediante el parámetro `--view` vamos a especificar que queremos usar Handlebars como motor de templates.



```
npx express-generator --view=hbs
```

Paso 3

También vamos a instalar **Nodemon**, que es el encargado de reiniciar automáticamente el servidor de desarrollo cada vez que cambiamos un archivo. Esto nos evita tener que estar constantemente reiniciándolo manualmente, lo que termina siendo tedioso y muy agotador.



```
npm i nodemon
```

Paso 4

Y por último con el comando `npm i` instalamos todas las dependencias que `express-generator` dejó en el archivo `package.json`.



```
npm i
```

Paso 5

Modificamos el **archivo package.json** para que el comando start ejecute nodemon en vez de node.

```
{  
  "name": "sitio-transportes-node",  
  "version": "0.0.0",  
  "private": true,  
  "scripts": {  
    "start": "nodemon ./bin/www"  
  }  
}
```

Paso 6

En la consola ejecutamos el comando **npm start**, luego vamos a un navegador y escribimos: **localhost:3000** para ver sitio.



Bibliografía utilizada y sugerida

Artículos de revista en formato electrónico:

Node.js Disponible desde la URL: <https://nodejs.org/es/>

Express. Disponible desde la URL: <https://expressjs.com/es/>