Librerías Clave en Node.js

Las librerías a continuación no son las únicas que existen, y no son camisa de fuerza; tu puedes usar las librerías que quieras y que se ajustan más a tus necesidades en [**https://npmjs.org/**](https://npmjs.org/) encontraras muchos paquetes que puedes usar con su respectiva información.

Librerías globales

SUPERVISOR

[**https://github.com/isaacs/node-supervisor**](https://github.com/isaacs/node-supervisor)

$ npm install -g supervisor

Cuando estamos trabajando en Node es necesario tener que compilar el código cada vez que hacemos un cambio.

SUPERVISOR reinicia la aplicación ante cualquier cambio, con la ventaja de que al ocurrir cualquier error en nuestro código, el servidor continúa su ejecución; al producirse un error, “supervisor“ avisará en la consola y mantendrá el servidor siempre despierto.

NODE-INSPECTOR

[**https://github.com/node-inspector/node-inspector**](https://github.com/node-inspector/node-inspector)

$ npm install -g node-inspector

Node-Inspector es una interfaz de depuración de Node.js utilizando las herramientas de desarrollo Blink (ex WebKit Web Inspector).

El depurador Blink es una gran interfaz depuradora de javascript, que funciona igual de bien para Node. Nodo Inspector es compatible con casi todas las características de depuración de DevTools.

-   Navegar en los archivos de origen
-   Establecer puntos de interrupción
-   Interrumpir en excepciones
-   Continuar a la ubicación
-   Desactivar / activar todos los puntos de interrupción
-   Inspeccione ámbitos, variables, propiedades de objeto
-   Editar las variables y propiedades de objeto

**Debugging con node-inspector**

**1.**En la terminal escribimos

$ node —debug ruta/a/tuprogram.js

**2.** Vamos a [**http://127.0.0.1:8080/debug?port=5858**](http://127.0.0.1:8080/debug?port=5858) en Chrome

Listo debería poder verse la fuente javascript de Node.

FOREVER

[**https://github.com/nodejitsu/forever**](https://github.com/nodejitsu/forever)

Este la forma de correr demonios en Node, es una herramienta CLI para asegurar que un paquete determinado de node se ejecuta continuamente.

$ npm install -g  forever

Nota: Para instalar forever se debe ser el administrador (sudo)

Hay dos formas distintas de utilizarlo siempre: a través de la interfaz de línea de comandos, o exigir el módulo para siempre se ejecute en su propio código.

**Nota:**Con FOREVER podemos mantener en constante ejecución a node-inspector, y si queremos cambiarlo de puerto.

Librerías locales

UNDERSCORE

[**http://underscorejs.org/**](http://underscorejs.org/)

$ npm install underscore --save

Underscore.js es una biblioteca de utilidades para JavaScript que proporciona apoyo a los sospechosos habituales funcionales (each, mapa, reducir, filtros ...) sin extender cualquier objeto de JavaScript.

SWIG

[**http://paularmstrong.github.io/swig/**](http://paularmstrong.github.io/swig/)

$ npm install swig --save

Es un motor de plantillas basado en Node y Javascript; al igual que en Django en los templates se trabaja con  Tags {% %} y Filtros {{ }} para construir las plantillas.

Ej.

<!doctype html>
**<html>**
**<head>**
    **<meta** **charset**="utf-8"**>**
    **<title>**{% block title %}Backend pro{% endblock %}**</title>**
    {% block head %}
    {% endblock %}
**</head>**
**<body>**
    {% block content %}{% endblock %}
**</body>**
**</html>**

Luego de construir la plantilla se renderiza en el server.

**1.** Requiere el paquete swig en el servidor.

swig = **require**('swig');

**2.**Construir la vista del motor, para renderizar tu plantilla

server.engine('html', swig.renderFile );

**3.**Renderizar los templates

server.**set**('views', './app/views');

**Nota:** Se esta usando Swig con Express.

CONNECT-REDIS

[**https://github.com/visionmedia/connect-redis**](https://github.com/visionmedia/connect-redis)

$ npm install connect-redis –save

Es un paquete de Node que sirve para almacenar sesiones de Redis soportado por node\_redis y es  increíblemente rápido.