Generadores

Tiempo estimado: 10min

Para finalizar el curso, vamos a presentar los generadores, componentes que han comenzado a hacerse muy populares dentro de la comunidad de desarrollo, porque facilitan enormemente nuestro trabajo y aumentan la productividad.

La lección comienza introduciendo el concepto de generador. A continuación, se presenta el generador oficial de Express, o sea, el desarrollado por el equipo de Express. Finalmente, el de Justo.js, bastante más completo que el anterior.

Al finalizar la lección, el estudiante sabrá:

- Qué es un generador.
- Cómo instalar y usar el generador express-generator.
- Cómo instalar y usar el generador justo-generator-express.

Introducción

Un generador (*generator*) es un componente que ayuda a realizar determinadas tareas en proyectos, principalmente aunque no necesariamente, de software. Tienen básicamente dos funciones:

- La creación de la estructura inicial de un proyecto como, por ejemplo, de una aplicación Express, Node o React.
- La ayuda en la creación de determinados elementos de un proyecto como, por ejemplo, la creación de rutas o encaminadores de una aplicación Express.

Los generadores son similares a los *plugins*, proporcionan una funcionalidad reutilizable. Pero se utilizan para producir directorios y/o archivos a partir de plantillas. Suelen recolectar datos del usuario y entonces generar la estructura de directorios más adecuada para lo indicado.

El equipo de Express proporciona un generador oficial. Este generador es muy sencillo y útil para proyectos de pequeño tamaño. Por otra parte, tenemos el generador de Justo.js que es más completo. Veamos cada uno de ellos.

Generador express-generator

El generador oficial de Express es express-generator. Es muy sencillo. La información hay que dársela mediante la línea de comandos a modo de opciones. Básicamente, su principal ventaja es que puede configurar como motor de plantillas Handlebars, EJS o Hogan. A diferencia del generador de Justo que sólo trabaja con Handlebars.

Instalación del generador

El generador se encuentra disponible mediante NPM, recomendándose una instalación global:

```
npm install -g express-generator
```

Una vez instalado, se recomienda comprobar que se tiene acceso al generador, implementado mediante el comando express, mostrando su ayuda mediante la opción -h o --help o bien su versión, -V o --version:

```
express -h
```

Generación de la estructura de directorios

Para crear la estructura inicial de directorios de un proyecto, no hay más que invocar el comando express y lo creará. Mediante la opción --hbs se indica que configure Handlebars como motor de plantillas. Si además

deseamos indicar que cree el archivo .gitignore, indicar --git. Si no se indica ningún nombre de directorio, esto es, el nombre del directorio del proyecto, la estructura la creará en el directorio actual.

A continuación, se muestra un ejemplo ilustrativo:

```
> express --hbs --git

   create : .
   create : ./package.json
   create : ./app.js
   create : ./.gitignore
   create : ./public
   create : ./routes
   create : ./routes/index.js
   create : ./routes/users.js
   create : ./views
   create : ./views/index.hbs
   create : ./views/layout.hbs
   create : ./views/error.hbs
   create : ./public/images
   create : ./public/javascripts
   create : ./public/stylesheets
   create : ./public/stylesheets/style.css
   create : ./bin
   create : ./bin/www

   install dependencies:
     > cd . && npm install

   run the app:
     > SET DEBUG=generador:* & npm start

>
```

Una vez generada la estructura del proyecto, no hay más que instalar las dependencias y ejecutar la aplicación mediante npm start.

Generador justo-generator-express

Justo es una herramienta de automatización desarrollada en Node que puede extenderse muy fácilmente mediante JavaScript. La automatización (*automation*) es el proceso mediante el cual se convierte tareas manuales en automáticas. Al automatizar las tareas, podemos ejecutarlas más fácil y rápidamente que si tuviéramos que hacerlo una y otra vez manualmente.

La automatización se lleva a cabo mediante herramientas de automatización (*automation tools*), aplicaciones de software que permiten convertir e implementar tareas manuales repetitivas en automáticas. Entre otras tenemos Ansible, Grunt, Gulp, Justo o Puppet, cada una de ellas orientadas a un determinado mercado. No todas ellas proporcionan soporte nativo para generadores, pero Justo sí lo hace.

El generador de Justo para Express presenta las siguientes características con respecto al oficial, o sea, al desarrollado por el equipo de Express:

- Configura Handlebars como motor de plantillas. El oficial permite configurar otros adicionales como EJS y Hogan.
- Permite configurar nodemon como monitor de cambios. Express no lo hace.
- Permite crear encaminadores, recordemos, los archivos ubicados en la carpeta routes. El oficial de Express no permite añadir nuevos archivos una vez creada la estructura. Aunque, claro está, se pueden crear manualmente. Además, para cada nuevo encaminador, actualiza el archivo routes/map.js.
- Permite añadir nuevas rutas a encaminadores. El oficial no lo permite.
- Permite crear archivos de plantilla Handlebars. El oficial no lo hace. Una vez generada la estructura del proyecto, hay que crearlos manualmente.
- Permite configurar componentes de *middleware* como, por ejemplo, Helmet, Morgan, serve-static y

serve-favicon.

- Permite configurar la aplicación Express para servir una aplicación React. El oficial no lo hace.

El repositorio Git oficial del generador se encuentra github.com/justojsg/justo-generator-express.

Instalación del generador

Justo es una herramienta que implementa varios aspectos de automatización como, por ejemplo, la automatización de tareas, los generadores y las pruebas de unidad. Para trabajar con los generadores, hay que instalar dos módulos disponibles en NPM: justo-cli, la interfaz de usuario de línea de comandos de Justo; y justo-generator-express, el generador específico de Express. Ambas cosas hay que instalarlas globalmente:

```
npm install -g justo-cli justo-generator-express
```

Una vez instalado, lo primero es comprobar que se tiene acceso al comando justo. Para ello, podemos consultar su ayuda:

```
justo -h
```

Si todo ha ido bien, con justo-cli se habrá instalado el comando justo. Recordemos, la interfaz de línea de comandos de Justo.

Por otra parte, tenemos que comprobar el acceso al generador de Express. Por ejemplo, como sigue:

```
justo -g express help
```

La opción -g se utiliza para invocar el generador que se indica, en nuestro caso, el de Express, instalado con justo-generator-express.

Generación de la estructura de directorios

Para crear la estructura de directorios inicial del proyecto, no hay más que invocar el generador:

```
justo -g express
```

Para crear la estructura de directorios y la configuración inicial del proyecto, el generador necesita información del usuario, la cual va solicitando una a una. Una vez recolectada toda la información necesaria, generará la estructura de directorios en el directorio actual. Entre la información que solicita al usuario encontramos:

- Descripción, página web, autor y un colaborador del proyecto.
- Repositorio Git del proyecto.
- Nombre del *script* con el que arrancar la aplicación en producción.
- Algunos componentes de *middleware* a instalar y configurar como, por ejemplo, Helmet, Morgan, serve-favicon, serve-static, body-parser, cookie-parser, cookie-session y express-session.

  Si alguno de los componentes a instalar requiere configuración especial, el generador la solicitará para hacer una buena configuración inicial. Por ejemplo, si decidimos usar Morgan como registro de eventos, solicitará el formato de las entradas; de serve-favicon, la ruta del archivo favicono; etc.
- Si usar nodemon como monitor de cambios.
- Si se servirá una aplicación React.

  En este caso, el generador configurará la aplicación Express para servir una aplicación de este tipo.

Una vez generada la estructura del proyecto, no hay más que instalar las dependencias y ejecutar la aplicación mediante npm start. Si hemos configurado nodemon, podemos arrancar con npm run start-dev.

Creación de encaminadores

Una de las ventajas del generador de Justo para Express es que no se conforma con crear simplemente la estructura de directorios. Nos ayuda a crear elementos del proyecto a lo largo de todo el ciclo de desarrollo. Por ejemplo, podemos crear encaminadores con él. Además de crear el correspondiente archivo en la carpeta routes, lo registrará en el archivo routes/map.js. Lo que ayuda a añadir encaminadores muy fácilmente.

No hay más que usar el comando router:


```
justo -g express router
```

De nuevo, solicitará la información que necesita y, a partir de ella, creará el archivo y actualizará map.js. Entre la información solicitada tenemos: el nombre del directorio, dentro de la carpeta routes, al que añadir el encaminador y la plantilla a reproducir cuando se acceda a la ruta raíz (/) del componente.

Veamos un ejemplo ilustrativo:

```
> justo -g express router
? Add to subfolder api
? Router name band
? View to render when root (/) requested band/index

  Generation
  [ OK ] Check whether 'routes/api' exists (0 ms)
  [ OK ] Generate routes\api\band.js (100 ms)
  [ OK ] Append content to routes\map.js (27 ms)

  OK 3 | Failed 0 | Ignored 0 | Total 3
>
```

Creación de rutas

El generador de Justo también nos asiste en la creación de rutas en un encaminador, mediante el comando route:

```
justo -g express route
```

Ejemplo:

```
> justo -g express route
? Router folder api
? Router name band
? Route path :name
? View to render when path requested band/info
? Method get

  Generation
  [ OK ] Append content to routes\api\band.js (46 ms)

  OK 1 | Failed 0 | Ignored 0 | Total 1
>
```

En este caso, lo que hace es registrar la ruta en el archivo del encaminador indicado. Proporciona soporte para los métodos HTTP más utilizados: GET, POST, PATCH, etc.

Creación de plantillas handlebars

Para añadir un archivo plantilla Handlebars, podemos usar los comandos hbs view y hbs partial:

```
justo -g express hbs view
justo -g express hbs partial
```

El primero añade la plantilla al directorio views. Mientras que el segundo a views/partials.

