scraping de datos de la pagina ‘cuevana3’

Jose Antonio Herrera Ramírez

Contenido

[Resumen Ejecutivo 2](#_Toc482533717)

[Introducción 2](#_Toc482533718)

[Desarrollo 3](#_Toc482533719)

[Obtención de datos 3](#_Toc482533720)

[Almacenamiento de datos 5](#_Toc482533721)

[Conclusiones y Recomendaciones 6](#_Toc482533722)

[Bibliografía 7](#_Toc482533723)

# Resumen Ejecutivo

Como parte de la implementación de la metodología previamente establecida para hacer scraping de datos en una página web, se implementó este scraper para extraer los datos de la página ‘Cuevana3’, los datos extraídos son de las películas que la pagina ofrece. El programa realiza la obtención de datos de las películas recorriendo todas las paginas disponibles y va guardando esta información en una base de datos no relacional debido a la comodidad con el manejo de datos orientados a documentos y Javascript.

En esta recolección se obtienen diferentes tipos de datos los cuales pueden servir luego para realizar consultas a los datos, filtrarlos por algún criterio o usarlos de cualquier manera.

# Introducción

Hoy en día el web scraping es una técnica muy utilizada entre los campos de la minería de datos, esta técnica consiste en extraer la información de páginas webs utilizando bots(programas) que realizan este proceso automáticamente.

Este scraper pretende dar una introducción al desarrollo e implementación de esta técnica, abarcando la extracción y manipulación de datos en un entorno como NodeJS. El entorno y las librerías usadas pueden ser familiares para aquellos usuarios Javascript ya que todo se desarrolla bajo este lenguaje.

Este trabajo es de interés para aquellos que desean envolverse de alguna manera en técnicas de minería de datos o talvez para algún usuario que necesite un conjunto de datos para manipular en su aplicación, etc. El punto es que hoy en día es una de las técnicas más utilizadas para extraer la información y muchos sistemas o navegadores la implementan.

Sin más preámbulo, a continuación, se detalla todo el proceso de desarrollo de esta herramienta aplicada a la extracción de datos sobre una página web de películas.

# Desarrollo

Como lenguaje de programación se utilizó Javascript en el entorno NodeJS, mediante npm se instalaron paquetes para cumplir diversas funcionalidades, a continuación, se mencionan y explican cada uno de los paquetes usados en el programa:

* **Express:** un framework del lado del servidor, bastante robusto, ofrece muchas funciones para web apps, utilizado para el lado del servidor.
* **Request:** una librería para simplificar las peticiones http, utilizada para hacer las peticiones a las páginas web y obtener el html de estas.
* **Cheerio:** una librería construida con el core de jQuery diseñada para el lado del servidor a fines de la manipulación de elementos del DOM.
* **Mongoose:** El ODM(Object-Document Mapping) con soporte oficial para MongoDB, utilizado para interactuar con la base de datos.

## Obtención de datos

La obtención de datos de la página web se logró con la ayuda de los paquetes ‘Request’ y ‘Cheerio’, el primero funciona para realizar peticiones y obtener el html de cada una de las paginas pedidas para luego hacer navegar por el DOM con la ayuda del paquete ‘Cheerio’ el cual provee una manera muy simple de hacer esto, al estar basado en el core de jQuery es casi la misma manera para acceder al DOM que distingue a jQuery.

A continuación, se ve un ejemplo en el uso de estos paquetes:

**var** url = 'http://www.cuevana3.com/';  
request(url,**function**(error, response, html) {  
 **if**(!error){  
 **var** $ = cheerio.load(html);  
 callPaginationLink(url);  
 }  
});

Como se aprecia, el método request tiene un formato request(url,callbackFunction).

En el callback podemos inyectar el html obtenido de la página para luego ser usado por Cheerio y navegar por el DOM.

Con la variable $ definida por cheerio ahora podemos usar el DOM Traversing de jQuery como se aprecia a continuación:

**var** imgNode = $('.info\_movie img');

El código anterior guarda en la variable *imgNode* el nodo img perteneciente al nodo con la clase info\_movie.

La página contiene un formato definido para mostrar los datos de todas las películas, por lo que no fue de mayor complejidad a la hora de extraer los datos, en general el proceso fue de navegar hasta el contenedor que tiene toda la información e ir navegando por cada uno de sus hijos, que en este caso eran etiquetas <span>, lo tedioso del proceso radicaba en darle formato a los datos, ya que estos eran presentados como texto plano, sin embargo ya se tenía un control sobre todos los nodos hijos, por lo que solo fue necesario ver cual nodo identificar el nodo y se interpretaba que siempre iba a ser la fecha de lanzamiento, por ejemplo.

En el proceso, todos los datos se iban guardando en un json para luego ser transformados a un modelo de Mongoose para su posterior almacenamiento en la BD.

## Almacenamiento de datos

El motor de almacenamiento de usado es MongoDB, esto debido a la comodidad para trabajar desde JavaScript con BD’s no relacionales orientadas a documentos.

La instancia de MongoDB se ejecuta en la plataforma de Azure como un DBaas(DataBase as as service) por lo que Azure se encarga de la gestión del servicio y no se necesita instalar ningún software para ejecutar la instancia ya que es hospedado en la nube.

Para establecer la conexión con la BD en la nube, Azure brinda un string de conexión en el formato establecido por MongoDB, el cual se ve de esta manera:

mongodb://[username:password@]host1[:port1][,host2[:port2],...[,hostN[:portN]]][/[database][?options]]

Luego desde Mongoose se establece una conexión mediante el método ***mongoose.connect****(connectionStr)*.

Al trabajar con este ODM, podemos realizar esquemas de los datos que al final serán convertidos en modelos para luego ser instanciados como un objeto que tiene sus propios métodos, entre ellos el **save(),** que funciona para guardar una instancia del modelo en la BD. Esto se logra con el json previamente formado en la obtención de datos al ser tradado en una expresión como la siguiente.

**new** Movie(json).save();

Siendo ‘Movie’ el nombre del modelo y ‘json’ el objeto formado en la obtención de datos, esta expresión crea una instancia de una película y lo guarda en la BD.

# Conclusiones y Recomendaciones

Se concluyó que mediante esta herramienta se pueden obtener cualquier cantidad de datos que luego pudiesen ser analizados o implementados en otro sistema.

Asimismo, se definió un entorno para desarrollar esta herramienta en la plataforma NodeJS y se analizó cada uno de los componentes de este scraper, esto puede servir de base para implementar esta herramienta en otras aplicaciones.

Se recomienda leer la documentación de cada uno de los paquetes utilizados para tener una mejor perspectiva del papel que cumple cada uno en el desarrollo de la herramienta.

# Bibliografía

GitHub. (2017). cheeriojs/cheerio. [online] Available at: <https://github.com/cheeriojs/cheerio> [Accessed 3 Apr. 2017].

GitHub. (2017). request/request. [online] Available at: <https://github.com/request/request> [Accessed 3 Apr. 2017].

Mongoosejs.com. (2017). Mongoose ODM v4.9.3. [online] Available at: <http://mongoosejs.com>/ [Accessed 3 Apr. 2017].

Azure.microsoft.com. (2017). *Microsoft Azure: plataforma y servicios de  
informática en la nube. [online] Available at:* [https://azure.microsoft.com/es-  
es/](https://azure.microsoft.com/es-es/) *[Accessed 14 May 2017].*