Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Campus Tecnológico Local San Carlos

Logotipo, nombre de la empresa

Descripción generada automáticamente

IC-8057 Introducción al Desarrollo de Páginas Web

Profesor: Efrén Jiménez Delgado

Investigación Web Scraping (parte 2)

Estudiante:

Agustín Brenes Umaña - 2019042448

Grupo 51

Fecha de entrega: 28-03-2022

# Resumen

La siguiente investigación trata sobre el Web Scraping, el cual funciona para obtener información estructurada de manera eficiente de una o muchas páginas web. Se abarcarán técnicas que recorren los documentos HTML o XML de la página y herramientas que automatizan el proceso. Entre las herramientas, se presentarán tanto herramientas que no requieren el uso de código como bibliotecas para distintos lenguajes de programación, las cuales funcionan para distintas partes del proceso para hacerlo más sencillo. Además de las metodologías para ejecución y herramientas para mayor eficiencia, se discutirán sobre diferentes asuntos legales con los que se involucran los usuarios o empresas que deseen realizar el Scraping en distintas páginas web. Por último, se planteará un caso para realizar el Scraping en alguna página web con el lenguaje de programación Python y se documentará los atributos que se decidió elegir para su almacenamiento en una base de datos.

# Tabla de Contenidos

[Resumen 2](#_Toc99398121)

[Tabla de Contenidos 3](#_Toc99398122)

[Introducción 4](#_Toc99398123)

[Desarrollo 4](#_Toc99398124)

[Técnicas Existentes 4](#_Toc99398125)

[Herramientas Existentes 5](#_Toc99398126)

[Metodología de extracción para una página web 5](#_Toc99398127)

[Conclusiones y Recomendaciones 7](#_Toc99398128)

[Bibliografía 9](#_Toc99398129)

# Introducción

Desde antes de la invención del internet, los seres humanos hemos tenido la necesidad de obtener conocimiento e información de otras personas. Hace años, este proceso de aprendizaje era mucho más lento y tedioso, ya que requería que las personas se sentaran a leer y organizar la información obtenida para poder utilizarla luego. Sin embargo, desde la creación del internet, dicho acceso a información se ha vuelto mucho más sencillo y eficiente. Desde entonces, se han desarrollado técnicas para facilitar esta tarea al máximo, hasta el punto de llegar a la automatización del proceso. Una de estas técnicas de obtención y almacenamiento masivo de datos es el Web Scraping. En esta investigación se presentará dicha metodología con un ejemplo concreto programado, además de la documentación de las bibliotecas usadas y la información extraída de la página de ejemplo.

# Desarrollo

El Web Scraping, también conocido como Web Extraction, es una técnica de extracción de datos de la World Wide Web y almacenamiento en sistemas de archivos o bases de datos para su obtención o análisis posterior. Dicha consecución de información puede ser de manera manual con un usuario y un browser, o puede realizarse con automatización inteligente con un web crawler (Zhao, 2017). El objetivo principal para realizar el Scraping puede variar entre personas o compañías que lo realicen, desde monitoreo de precios hasta investigación de mercado. Sea cual sea la razón de su utilización, al realizarlo, se obtienen datos estructurados de cualquier sitio web público, lo cual facilita su análisis y utilización en distintos proyectos (Kenny, 2021).

## Técnicas Existentes

Existen múltiples técnicas para realizar Web Scraping, desde el copy-paste manual hasta la utilización de lenguajes de programación para extraer los datos. Algunas de las técnicas existentes como el HTML Parsing, DOM Parsing y el uso del XPath, las cuales se pueden ejecutar con el lenguaje Javascript, navegan a través de la estructura de árbol de los documentos HTML y XML para obtener información precisa sobre la estructura de la página web (Chapman, 2020). La agregación vertical es otra técnica utilizada por compañías que tienen a su disposición gran poder computacional, y esta es ejecutada sin intervención humana en muchos casos. Por último, el matching de patrones de texto es otra técnica que involucra el uso del comando UNIX grep y se usa con lenguajes de programación populares como Perl o Python (Chapman, 2020).

## Herramientas Existentes

Existen múltiples herramientas que ayudan al proceso de Web Scraping con la automatización, desde browsers programables hasta bibliotecas para lenguajes como Python, Perl y Javascript. Google Sheets, por ejemplo, tiene una función de importar archivos XML a la hoja de cálculo. Esta puede ser utilizada para extraer toda la información necesaria para su uso posterior, aunque esta debe estar bien estructurada para que los patrones sean bien reconocidos. Por otra parte, Selenium es una herramienta de automatización en navegadores web que puede ser utilizada para copiar el comportamiento de un ser humano obteniendo información de una página web cualquiera (Chapman, 2020).

Para entrar en las herramientas programadas, existe el web crawler llamado Nutch, el cual posee uno de los algoritmos más veloces existentes. Este requiere que las páginas que se van a utilizar sean programadas manualmente en el programa, para luego extraer y almacenar la información obtenida (Chapman, 2020). Por último, existen bibliotecas en leguajes de programación como BeautifulSoup para Python, Nodejs-Web-Scraper para NodeJS, Goutte para PHP, Boilerpipe para Java, Watir para Ruby y Axios para Javascript. Todas estas funcionan para diferentes partes de la obtención de datos en la página, desde hacer requests HTTP y limpiar los tags innecesarios de la página, hasta copiar el comportamiento humano al interactuar con la página web.

## Metodología de extracción para una página web

Para esta investigación, se planteó un caso de Web Scraping con la página web de CR Autos de autos usados en zona verde (<https://crautos.com/autosusados/zonaverde/>). El caso utilizaría el lenguaje de programación Python, para el cual se escribe un programa el cual debe obtener los datos de la página utilizando la herramienta llamada BeautifulSoup junto a la biblioteca de requests. Con esta última se obtiene toda la información del HTML. Una vez obtenida toda la información cruda, esta se parsea con la biblioteca de BeautifulSoup. Una vez que esta ha sido parseada y se extraen los datos que se desean analizar o persistir, se conecta a la base de datos MongoDB, la cual almacenaría todos los elementos de la lista obtenida en el sitio. La página tiene una lista de elementos que representa los carros usados híbridos o eléctricos en venta. Esta información sería almacenada en la base de datos para su análisis posterior. Entre los atributos que se obtienen con su XPATH respectivo se adjuntan a continuación:

1. **Cilindrada:** Atributo que se utiliza para conocer el atributo de los cilindros del automóvil. XPATH: *class="technical mytext2" tbody tr(0) td(1)*
2. **Estilo:** Atributo que se utiliza para conocer el estilo del carro, como Sedán o Todo terreno 4x4. XPATH: *class="technical mytext2" tbody tr(1) td(1)*
3. **Combustible:** Atributo que se utiliza para conocer el tipo de combustible que necesita, como eléctrico o disel. XPATH: *class="technical mytext2" tbody tr(2) td(1)*
4. **Transmisión:** Atributo que se utiliza para conocer si el auto es de tipo manual, automático o híbrido. XPATH: *class="technical mytext2" tbody tr(3) td(1)*
5. **Estado:** Atributo que se utiliza para conocer qué tan nuevo y que tanta calidad posee el auto actualmente. XPATH: *class="technical mytext2" tbody tr(4) td(1)*
6. **Kilometraje:** Atributo que se utiliza para conocer la cantidad de kilómetros que ha recorrido el auto durante su tiempo de vida en uso. XPATH: *class="technical mytext2" tbody tr(5) td(1)*
7. **Placa:** Atributo que se utiliza para conocer con qué número termina la placa del auto. XPATH: *class="technical mytext2" tbody tr(6) td(1)*
8. **Color exterior:** Atributo que se utiliza para conocer el color de la pintura en el exterior de la carrocería. XPATH: *class="technical mytext2" tbody tr(7) td(1)*
9. **Color interior:** Atributo que se utiliza para conocer el color de la tapicería y el interior del vehículo. XPATH: *class="technical mytext2" tbody tr(8) td(1)*
10. **# de puertas:** Atributo que se utiliza para conocer la cantidad de puertas que posee el auto. XPATH: *class="technical mytext2" tbody tr(9) td(1)*
11. **Ya pagó impuestos:** Atributo que se utiliza para conocer si el vendedor del auto ya pagó los impuestos del auto que está en venta. XPATH: *class="technical mytext2" tbody tr(10) td(1)*
12. **Precio negociable:** Atributo que se utiliza para conocer si el vendedor del auto está dispuesto a negociar con el comprador por un precio que este proponga. XPATH: *class="technical mytext2" tbody tr(11) td(1)*
13. **Se recibe vehículo:** Atributo que se utiliza para conocer si el vendedor del auto está dispuesto a intercambiar el carro en venta por otro que tenga el comprador. XPATH: *class="technical mytext2" tbody tr(12) td(1)*
14. **Provincia:** Atributo que se utiliza para conocer la provincia de donde proviene el carro del vendedor. XPATH: *class="technical mytext2" tbody tr(13) td(1)*
15. **Costo de traspaso:** Atributo que se utiliza para conocer el precio por el cual se realizaría el traspaso de propiedad del vehículo entre el vendedor al comprador. Este puede varias depende del precio final de venta y el valor fiscal del vehículo. XPATH: *class="technical mytext2" tbody tr(14) td(1)*
16. **Fecha de ingreso:** Atributo que se utiliza para conocer la fecha en la cual se puso en venta el carro en la página. XPATH: *class="technical mytext2" tbody tr(15) td(1)*

Por último, la solución programada se encuentra en el repositorio de Github en el siguiente enlace: <https://github.com/agusbrenes/Web-Scraper>.

# Conclusiones y Recomendaciones

En síntesis, la utilización de técnicas y herramientas para el Web Scraping le ha facilitado la vida a usuarios o empresas que deseen obtener información estructurada de manera eficiente. Lo más recomendable al utilizar herramientas para realizarlo es saber cómo utilizar tanto herramientas que no requieren escritura de código como bibliotecas que ayudan al proceso de obtención y limpieza de datos. En cuanto a lenguajes de programación, muchos tienen soporte para extracción de datos y siempre se puede realizar el Scraping con un poco de código. Al fin de cuentas, realmente queda a decisión del programador dependiendo de cuál es más cómodo o con cuál se tiene más experiencia. Cabe destacar que, al momento de tomar la decisión de realizar Scraping en una página, se debe leer todos los términos y condiciones de esta (en el caso de que tenga). Muchas páginas web no permiten el Scraping libremente, por lo que hay que revisar cuidadosamente que la electa lo permita y no se encuentren conflictos legales. Finalmente, a la hora de utilizar una solución programada para realizar Scraping, es necesario que se revise la estructura completa de la página elegida para poder seleccionar adecuadamente la información que se va a recompilar; con este análisis previo se puede saber qué partes extraer y como limpiar los datos obtenidos.

# Bibliografía

Chapman, R. (2020). *Top 10 Web Scraping Techniques*. Limeproxies. Recuperado de <https://limeproxies.netlify.app/blog/top-10-web-scraping-techniques>.

Joshi, A. (2021). *Legality of Web Scraping*. Grepsr. Recuperado de <https://www.grepsr.com/blog/overview-web-scraping-legality/>.

Kenny, C. (2021). *What Is Web Scraping And How Does It Work?* Zyte. Recuperado de <https://www.zyte.com/learn/what-is-web-scraping/>.

Nodejs Web Scraper. (2022). Recuperado de [https://nodejs-web-scraper.ibrod83.com/.](https://nodejs-web-scraper.ibrod83.com/)

Reitz, K. (2022). *HTTP for humans*. Requests. Recuperado de <https://docs.python-requests.org/en/latest/>.

Richardson, L. (2021). *Beautiful Soup*. Crummy. Recuperado de <https://www.crummy.com/software/BeautifulSoup/>.

Zhao, B. (2017). Web scraping. *Encyclopedia of big data*, 1-3.