Scraping web
y
Automatización web
con Python

Parte 1
Scraping web

¿Qué es el **scraping**?

Entendemos por scraping al conjunto de técnicas que nos permiten obtener información de una página web, de manera automatizada, cuyo contenido fue originalmente creado para ser visualizado sólo por personas.

Por ejemplo: un bot que hace scraping de páginas de productos en Amazon, para obtener sus precios y consultar posibles bajadas de forma automática.

¿Por qué no usar una API?

Se suele recurrir al scraping cuando queremos obtener datos de un sitio web donde:

- No existe una API
- La API es de pago
- La API no nos ofrece toda la información que necesitaríamos obtener
- La API tiene limitaciones que queremos evitar (por ejemplo X solicitudes por minuto/hora/día)

Legalidad y ética

- El scraping no es ilegal
- ¿Es ético? Depende de lo que vayamos a hacer...
- Algunos sitios web especifican en sus términos y condiciones “no tratar informáticamente la información” del sitio web
- Sitios como Twitter hacen alusión específica al scraping
- No deberíamos abusar haciendo muchas peticiones

En consideración a Twitter y al acceso y uso que le concede de sus Servicios, usted accede a que Twitter y sus proveedores y socios puedan emplazar publicidad en los Servicios o publicidad relacionada con el contenido o información mostrados de los Servicios que haya enviado usted o que hayan enviado otros. También accede a no realizar un uso inapropiado de los Servicios, por ejemplo, interfiriendo con ellos o accediendo a ellos empleando un método distinto a la interfaz e instrucciones que nosotros le hayamos proporcionado. No realizará ninguna de las siguientes acciones al acceder a nuestros Servicios o hacer uso de ellos: (i) acceder, sabotear o usar las áreas no públicas de los Servicios, de los sistemas informáticos de Twitter o de los sistemas técnicos de los proveedores de Twitter; (ii) probar, escanear o comprobar la vulnerabilidad de cualquier sistema o red o infringir o sortear cualquier medida de autenticación de seguridad; (iii) acceder o investigar o intentar o acceder o investigar los Servicios en modo alguno (de forma automática o de cualquier otra forma) si no es a través de nuestras interfaces actualmente disponibles, publicadas y proporcionadas por Twitter (y solo según los términos y condiciones aplicables), a no ser que se le haya permitido específicamente hacerlo mediante un acuerdo distinto con Twitter (NOTA: el rastreo de los Servicios está permitido si se hace según las disposiciones del archivo robots.txt, sin embargo, el "scraping" de los Servicios sin el consentimiento previo escrito de Twitter está expresamente prohibido); (iv) insertar ningún encabezamiento de paquete TCP/IP ni ninguna parte de la información del encabezamiento en ningún correo electrónico o publicación, ni emplear los Servicios de ningún modo para enviar información de identificación de fuente alterada, errónea o falsa; o (v) interferir con, u obstaculizar, (o intentar hacerlo), el acceso de ningún usuario, host o red, incluyendo, entre otros, el envío de virus, sobrecargas, acumulaciones, spam, bombardeo de correos electrónicos del Servicio, o insertando script en la creación de Contenido de tal forma que interfiera o cree una carga indebida sobre los Servicios. También nos reservamos el derecho de acceder, leer, conservar y divulgar cualquier información que creamos que es necesariamente razonable para (i) cumplir con cualquier ley aplicable, norma, proceso legal o solicitud gubernamental, (ii) hacer cumplir los Términos, incluyendo la investigación de posibles infracciones de los mismos, (iii) detectar, prevenir o resolver de cualquier modo fraudes, problemas de seguridad o técnicos, o (iv) responder a solicitudes de ayuda de usuarios, o (v) proteger los derechos, la propiedad o la seguridad de Twitter, de sus usuarios y del público. Twitter no divulga información personal identificable a terceros, excepto según lo dispuesto en nuestra Política de privacidad.

Buenas prácticas para el scraping

- Controlar el número de peticiones lanzadas
- Emplear un user agent personalizado
- Aleatorizar la frecuencia con la que lanzamos peticiones
- Vigilar cuándo una página ha podido cambiar el formato de su código HTML

Partes involucradas en el scraping

- **Librería HTTP** que lance requests (peticiones) al sitio web deseado. Nos devuelve su código HTML
- **Parser HTML** que nos permita “navegar” por el código HTML fácilmente, para obtener la información que deseemos
- **Nuestro código**, donde lanzaremos peticiones, obtendremos el HTML y lo parsearemos para obtener y trabajar con la información deseada

Scraping y JavaScript

- Algunas páginas web generan parte de su contenido mediante la ejecución de uno o varios códigos JavaScript
- En estos casos, si nos limitamos a obtener el código fuente sin ejecutar el JavaScript, no podríamos obtener los datos deseados

Librerías para hacer scraping en Python
requests + beautifulsoup

- Requests: nos permite lanzar peticiones a las páginas queremos visualizar, obteniendo su código fuente HTML

  Página oficial: http://docs.python-requests.org/en/master/

- Beautifulsoup: librería para parsear el código HTML obtenido

  Página oficial: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Librerías para hacer scraping en Python
requests-html

- Combina la librería requests para realizar peticiones, con un parser de HTML
- Soporta ejecución de JavaScript, simulación de User Agent y persistencia (connection pooling y cookies)

Página oficial: https://html.python-requests.org/
Github: https://github.com/kennethreitz/requests-html

Parte 2

Automatización web

¿Qué es la **automatización web**?

Son aquellas técnicas utilizadas para interactuar con una página web y realizar acciones sobre la misma de manera automática y programada, simulando un comportamiento humano.

Guarda relación con el scraping, ya que suele ser necesario analizar el código de la página (al menos la primera vez) para saber cómo funciona, cómo está estructurada y cómo interactuar con ella.

¿Para qué se usa la automatización web?

- Para hacer web testing: por ejemplo, simular actividad humana sobre un portal para comprobar si se sobrecarga o cómo reacciona ante un cierto número de usuarios en línea
- Para automatizar tareas en páginas web
- Como complemento al scraping, cuando para acceder a los datos que necesitamos las herramientas de scraping se quedan cortas.

Partes involucradas en automatización web

- **Webdriver**: es un navegador web que se puede automatizar. Ejemplos:
  - Chromedriver (Google Chrome/Chromium)
  - Geckodriver (Mozilla Firefox)
  - Operachromiumdriver (Opera basado en Chromium)
- **Framework**: enlace entre nuestro código y el webdriver Ejemplo: Selenium (multiplataforma y multilenguaje)
- **Nuestro código**, que contiene todas las acciones que se realizan sobre una página (acceder a una dirección, rellenar formularios, pulsar elementos...)

Acciones en automatización

- Acceder a la página deseada
- Hacer scraping para:
  - Obtener información (textos, enlaces, objetos...)
  - Buscar elementos sobre los que interactuar
- Interactuar sobre elementos:
  - Pulsar sobre objetos (enlaces, imágenes, botones…)
  - Rellenar formularios
  - Pulsar teclas
- Capturas de pantalla, cambio entre pestañas/ventanas y marcos...

Buenas prácticas en automatización web

- Aleatorizar las acciones para simular un comportamiento humano (carga de página, navegación, tiempos, visualización de contenidos…)
- Vigilar qué User Agent utilizamos
- No abusar y realizar acciones repetitivas con una frecuencia aleatoria
- Implementar mecanismos que detecten y reaccionen ante posibles cambios en la estructura de la página