مستندات خزنده‌ی وِب[1]

علی علی‌محمّدی ۹۶۱۳۰۲۷

- در فایل فشرده‌ی zip، فایلِ Web_Scraping.ipynb که یک Jupyter Notebook شامل اسکریپت‌هایی به زبانِ Python وجود دارد.
- این کُد بر روی بسترِ Google Colab توسعه داده شده است و با آن سازگار است.
- از کتاب‌خانه‌های BeautifulSoup برای عملیات اشتقاق[2] و استخراج داده‌های موجود در سندِ[3] HTML، requests برای دریافت محتویات HTML یک صفحه با استفاده از URL آن صفحه، json برای سهولت در استفاده‌ی فُرمَتِ JSON[4]، Pillow برای نمایش تصاویر و عکس‌ها و base64 برای کدگذاری[5] عکس‌ها به نحوی که در قالبِ JSON Object قابل ذخیره باشد (زیرا نمی‌توان آرایه‌ی بایت[6] را بدون کدگذاری و به صورت خام در JSON Object ذخیره کرد)، استفاده شده است.
- نام سلول‌های کُد به همراه کامنت‌های برنامه به وضوح روند کار را تشریح می‌کنند. همچنین نام توابع به صراحت بیان‌گر کاربرد و منظور آن تابع است. همچنین نحوه‌ی ذخیره‌سازی اطلاعات در JSON Object نیز با توجه کلیدها[7]، گویا و واضح است.
- برنامه از قابلیت اطمینان بالایی برخوردار است و داده‌ها را به طور پیوسته در طی عملیات خزش ذخیره می‌کند. بدین ترتیب، در صورت به وجود آمدن وقفه[8] در اجرای برنامه، مثل قطعی اینترنت یا توقف عملیات توسط کاربر یا سیستم‌عامل، داده‌هایی که تا آن لحظه خزش شده‌اند، از دست نمی‌رود و درون فایل خروجی موجود خواهند بود.
- در هنگام اجرای برنامه، می‌توان تعداد کمینه‌ی محصولاتی لازم است مورد خزش قرار گیرند، در ورودی تابع scrape_category وارد نمود؛ بدین ترتیب، برنامه پس از اتمام خزش یک صفحه از فهرست محصولات، به صفحه‌ی بعدی می‌رود و آن را مورد خزش قرار می‌دهد و این چرخه را تا زمانی که حداقل item_min عدد محصول مورد خزش قرار گرفته باشند، ادامه می‌دهد.
- به ازای خزش هر دسته (هر بار فراخوانی تابع scrape_category)، یک فایل با پسوندِ json. و با نام آن دسته ساخته می‌شود و در هر سطر آن، یک JSON Object حاوی اطلاعات یک محصول وجود دارد؛ به عنوان مثال، برای دسته‌ی «تخت‌ها»، اطلاعات در فایلی با نامِ beds.json ذخیره می‌شوند.
- در رابطه با فایلِ robots.txt می‌توان گفت که این فایل به صورت محترمانه (!) از رُبات‌های جست‌وجوگر و خزنده تقاضا می‌کند که محدودیت‌هایی را در نظر بگیرند و آن‌ها را رعایت کنند؛ به عنوان مثال، اجازه‌ی جست‌وجوی محتویات وب‌سایت را برای دسته‌ی خاصی از botهای جست‌وجوگر صادر می‌کند و دسته‌ای دیگر را ممنوع می‌کند. در برخی از موارد نیز اجرای درخواست‌ها[9] را ممنوع اعلام می‌کند و از botها می‌خواهد که دستورات و درخواست‌های این‌چنینی را به سمت سرور ارسال نکنند یا از آن‌ها می‌خواهد که برای جست‌وجو در آن وب‌سایت، استفاده از چه فیلترهایی مجاز است. البته شایان به ذکر است

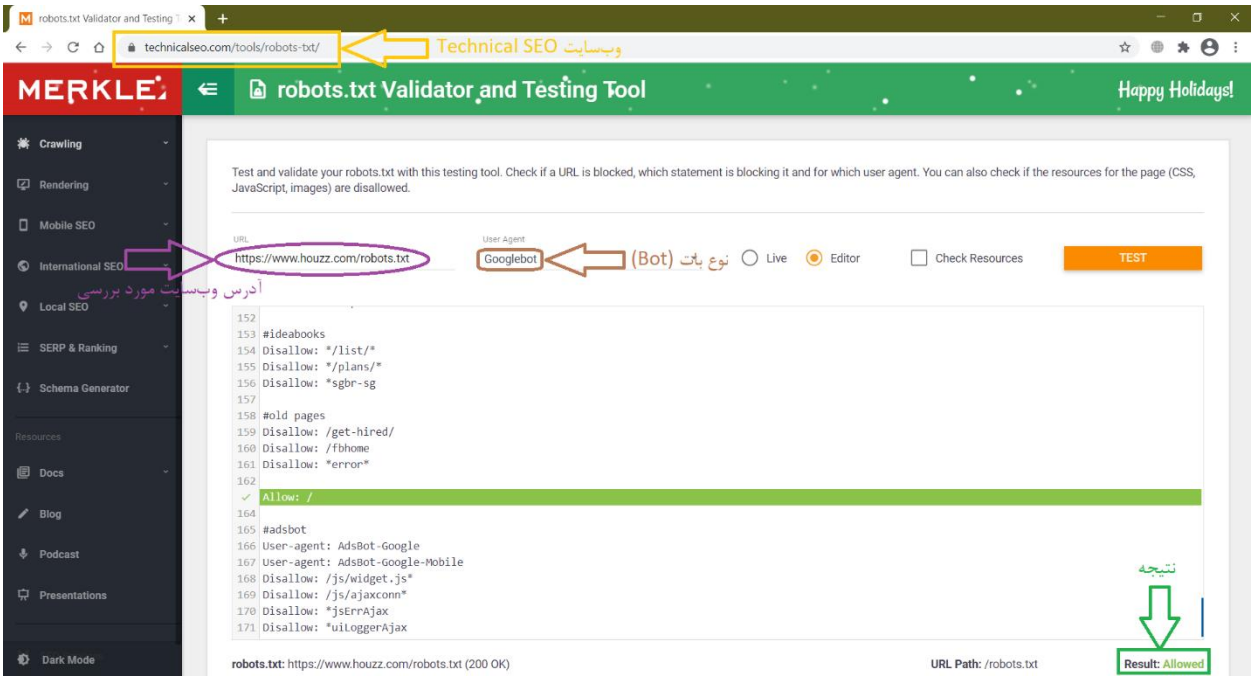
---

[1] Web Scraper
[2] Parsing
[3] Document
[4] JavaScript Object Notation
[5] Encoding
[6] Byte-Array
[7] Keys
[8] Interruption
[9] Queries

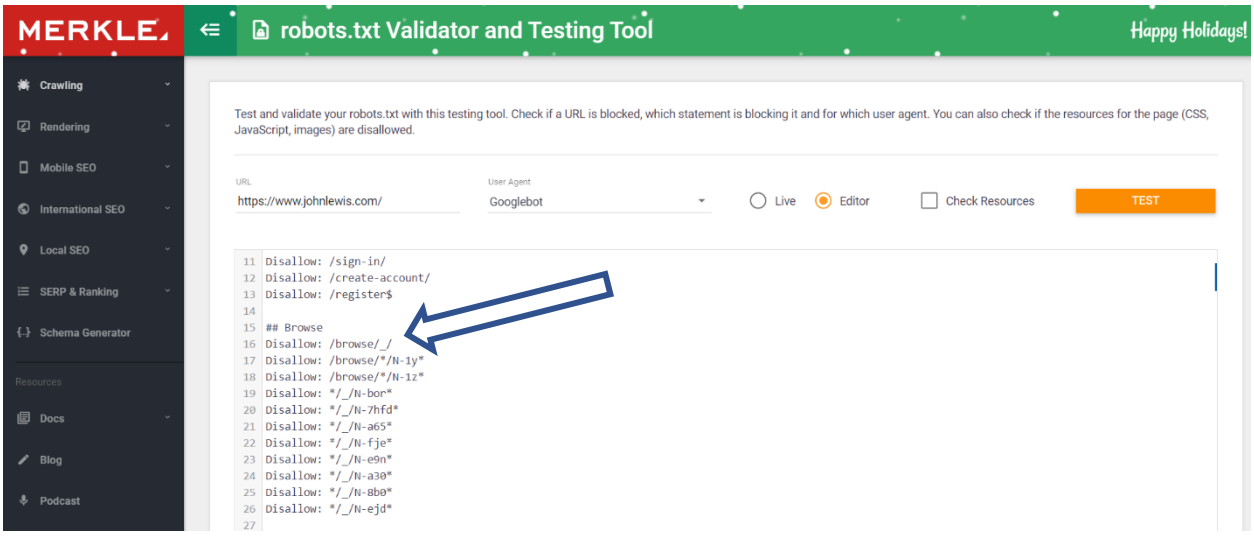
که رُبات‌های جست‌وجوگر لزوماً این قوانین را رعایت نمی‌کنند؛ به عنوان مثال، موتور جست‌وجوی گوگل[10] اعلام کرده است که از تاریخ یکُم سپتامبر سال ۲۰۱۹، دیگر از قوانینِ robots.txt پیروی نمی‌کند.

- برای درک بهتر این فایل و بررسی دقیق‌تر آن می‌توان از ابزار مربوط به آن در وب‌سایت «تکنیکال‌سِئو[11]» استفاده کرد. این وب‌سایت امکان استفاده از انواع bot خزنده را فراهم می‌سازد و در پایان نتیجه‌ی کلی را نیز اعلام می‌کند.

✓ برای بررسی و مقایسه‌ی وب‌سایت‌های Houzz و John Lewis از ابزار فوق استفاده شده است؛ نتیجه در تصاویر زیر مشاهده می‌شود:



نتیجه‌ی بررسی وب‌سایت Houzz



نتیجه‌ی بررسی وب‌سایت John Lewis

[10] Google Search Engine
[11] Technical SEO

- ✓ همان‌طور که مشاهده می‌شود، وب‌سایتِ Houzz اجازه‌ی جست‌وجوی محصولات را در فایلِ robots.txt صادر کرده است اما درباره‌ی وب‌سایتِ John Lewis چنین نیست؛ بلکه اجازه‌ی دسترسی را مسدود کرده است. بنابراین به نظر می‌رسد که وب‌سایتِ Houzz برای خزش مناسب‌تر از وب‌سایتِ John Lewis است.
- ✓ همچنین نحوه‌ی آدرس‌دهی در URL صفحه‌های وب‌سایتِ Houzz دارای ساختاری بسیار مشخص هستند و به راحتی می‌توان آدرس صفحه‌های مورد نیاز را از روی الگوی آن ساخت و مورد استفاده قرار داد اما درباره‌ی وب‌سایتِ John Lewis چنین نیست و الگوها[12] به راحتی قابل تشخیص نیستند و الگوهای پیچیده‌ای دارد که کار را برای دسترسی خودکار خزنده به صفحه‌های مختلف آن وب‌سایت دشوار می‌سازد.

- درباره‌ی Selenium و Scrapy می‌توان گفت که هر دو فریم‌وُرک‌های پایتون[13] هستند که برای خزش وب استفاده می‌شوند. این در حالی است که Selenium برای خودکارسازی تعاملات مرورگر وب[14] و تست خودکارِ Web Applicationها استفاده می‌شود اما Scrapy برای دانلود HTML، پردازش و ذخیره‌سازی آن مورد استفاده قرار می‌گیرد.
- از ویژگی‌های مهم Selenium می‌توان به این نکته اشاره کرد که به راحتی می‌توان با هسته‌ی مفاهیمِ JavaScript (DOM)[15] کار کرد. همچنین مدیریت و به کار بردن AJAX و PJAX در آن آسان است.
- یکی از مزایای اصلی Scrapy این است که بر روی بسترِ Twisted که یک فریم‌وُرک شبکه‌ی ناهمزمان[16] است، ساخته شده است که یعنی Scrapy هنگام ارسال درخواست‌ها به کاربران از مکانیسم غیر مسدودکننده[17] استفاده می‌کند.
- به طور کلی، Scrapy در دسته‌ی Web Scraping API قرار می‌گیرد، در حالی که Selenium در دسته‌ی Browser Testing Tools قرار دارد.

در صورت وجود هرگونه سؤال یا ابهام، با ایمیل alialimohammadi@ce.aut.ac.ir در تماس باشید.

[12] Patterns
[13] Python Frameworks
[14] Automate Web-browser Interaction
[15] Core JavaScript Concepts
[16] Asynchronous Networking Framework
[17] Non-blocking Mechanism