ULPGC
Universidad de Las Palmas de Gran Canaria
Escuela de Ingeniería Informática

Developing a Scraper that publishes data through an API

2º curso

Grado en Ciencia e Ingeniería de Datos

Desarrollo de Aplicaciones para Ciencia de Datos

Jaime Ballesteros Domínguez

12/01/2023

Tercera versión

Última revisión: 12/01/2023

Resumen

En este trabajo se nos pedía poner en práctica el desarrollo de un web service, el cual ofrecería datos obtenidos de una página web en HTML y los transformaría al formato JSON. En este caso, la página de la cual obtendríamos los datos sería booking.com.

Comencé el programa dividiendo mis extracciones en cuatro campos; la ubicación, los servicios que pudiese ofrecer el establecimiento, las notas dadas por los visitantes a ciertos aspectos de la estancia y los comentarios de los usuarios. Una vez establecida dicha división, procedí a crear una clase para cada campo de extracción, la cual contendría los diferentes elementos a mostrar en un futuro y comencé con la obtención de datos. Dichas operaciones estarían ubicadas en una clase con el nombre “Booking”.

Posteriormente, ya que la extracción de información la estaba realizando pasándole la URL del establecimiento como un parámetro más, decidí que era más recomendable si el programa buscase por internet la URL, pasándole el nombre del establecimiento más una string denominada “base” que se repite en todas las búsquedas de Booking. Cabe destacar que, a la hora de obtener los comentarios, no utilicé la propia página de Booking, sino su página de comentarios, la cual tiene una URL ligeramente distinta.

Ya para acabar, trasladé los distintos métodos de obtención y recolección de datos a una interfaz (HotelScrapper) y concreté los distintos métodos GET en la clase APICalls. En cada uno de los métodos conectados con el repositorio, al comenzar el programa, el usuario introduciría en dicho repositorio la String “http://localhost:4567/hotels/” junto con el nombre del establecimiento y una de las distintas variables requeridas en el enunciado del programa. Una vez mandada la petición, el programa recogería el nombre del alojamiento y comenzaría con la obtención de datos, devolviendo al final una respuesta en formato JSON.

Ejemplos:

- http://localhost:4567/hotels/riu_club_gran_canaria
- http://localhost:4567/hotels/riu_club_gran_canaria/comments
- http://localhost:4567/hotels/riu_club_gran_canaria/ratings
- http://localhost:4567/hotels/riu_club_gran_canaria/services

Índice

Resumen 3
Recursos utilizados 5
Diseño 5
Conclusiones 6
Líneas futuras 6
Bibliografía 6

Recursos utilizados

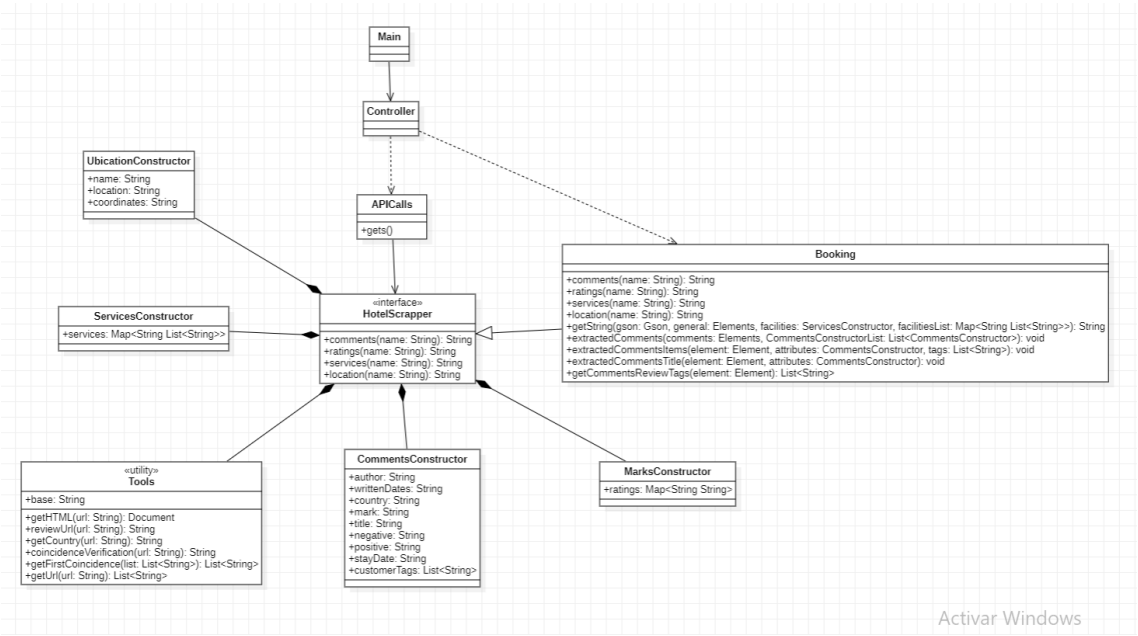
Entornos de desarrollo: El entorno de desarrollo utilizado para este trabajo fue IntelliJ IDEA desarrollado por JetBrains.

Herramientas de control de versiones: El software de control de versiones utilizados ha sido Git. Aquí se ha guardado tanto el código ejecutable como este pdf y el diagrama de clases.

Herramientas de documentación: Para documentar el trabajo realizado se ha utilizado el software de tratamiento de textos Microsoft Word.

Diseño

En este trabajo se intentó seguir el estilo de diseño Model View Controller (MVC), siendo las clases Booking y Tools, junto con la interfaz un ejemplo de Model al manejar los datos a obtener, APICalls estaría relacionado con View, al ser la encargada de representar la información obtenida al realizar la conexión con el repositorio y la clase Controller, como su nombre indica, recibiría las órdenes del usuario y se encargaría de estar en contacto con el Model y el View.



Aquí se puede apreciar el diagrama de clases resultante, el cual se puede ver con mayor calidad en el repositorio.

Conclusiones

En el proceso de realización de este trabajo he podido aprender tanto métodos como conceptos nuevos y formas de trabajar con ellos. Algunos ejemplos pueden ser:

- La obtención de datos HTML de una página web de maneras completamente distintas.
- El establecimiento de conexiones entre nuestro programa y el repositorio correspondiente.
- La ampliación de conocimientos sobre regex.

Líneas futuras

Para que el producto obtenido se pueda convertir en uno comercializable se podría añadir más información a representar, como podría ser el teléfono del alojamiento. También se podría añadir la función que se nos planteó de manera opcional, la cual al introducir el nombre de la ciudad en la que uno se va a quedar, devolverá los nombres de todos los posibles lugares en los que nos podemos alojar.

Por último, intentaría buscar la forma de reducir los tiempos de ejecución ya que hay algunas funciones como la de obtención del nombre y localización que tardan bastante en mostrar los datos resultantes a pesar de pedir menos información que en otras llamadas.

Bibliografía

https://stackoverflow.com/questions/17907812/tojsonstring-is-undefined-for-the-type-jsonobject

https://stackoverflow.com/questions/25358485/how-to-create-multiple-json-object-dynamically

https://www.baeldung.com/json-multiple-fields-single-java-field

https://www.w3schools.com/java/java_regex.asp

https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html

https://stackoverflow.com/questions/18838906/java-regex-first-match-only

https://stackoverflow.com/questions/43270019/java-regex-extract-string-between-two-strings

https://www.techiedelight.com/es/concatenate-two-strings-java/

https://stackoverflow.com/questions/20366408/keeping-multiple-values-under-a-single-key-in-dictionary

https://www.baeldung.com/spark-framework-rest-api

https://howtodoinjava.com/java/collections/hashmap/merge-two-hashmaps/

https://www.programiz.com/java-programming/library/hashmap/merge

https://www.youtube.com/watch?v=kRWiUsHkIts