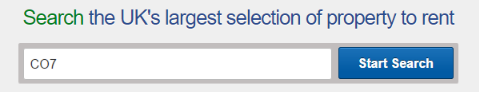
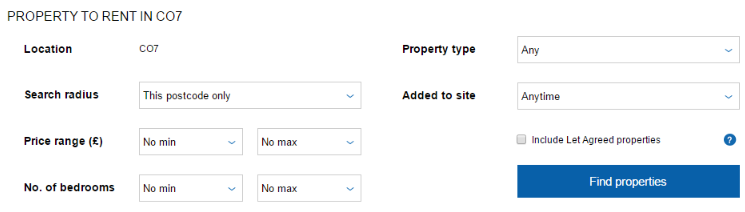
**Navigation**

Choose an outcode:



You will be presented with a search screen:



Choose not to enter any filters. Do not check ‘Let Agreed’ properties as this will introduce some bias as we don’t know what RightMove will choose to show and not show here. We limit ourselves to properties that are not yet let agreed, i.e. those that are still available.

Upon clicking ‘Find Properties’, a user is navigated to the following URL:

<http://www.rightmove.co.uk/property-to-rent/find.html?searchType=RENT&locationIdentifier=OUTCODE%5E520&insId=3&radius=0.0&minPrice=&maxPrice=&minBedrooms=&maxBedrooms=&displayPropertyType=&maxDaysSinceAdded=&sortByPriceDescending=&_includeLetAgreed=on&primaryDisplayPropertyType=&secondaryDisplayPropertyType=&oldDisplayPropertyType=&oldPrimaryDisplayPropertyType=&letType=&letFurnishType=&houseFlatShare=false>

Manual experimentation tells us that the following simpler URL will have the same effect:

<http://www.rightmove.co.uk/property-to-rent/find.html?searchType=RENT&locationIdentifier=OUTCODE%5E520>

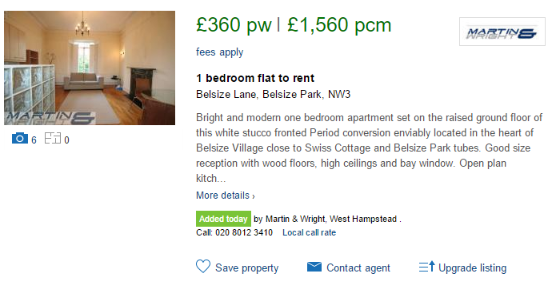
Note that the outcode has been converted by the website from CO7 to 520. 520 is RightMove’s internal ID for the outcode CO7, and the only ways to discover the mappings between outcodes and the internal IDs for the is manual discovery or an automated loop to try them all. We discovered that the internal IDs covering the whole of the UK ranged from 1 to 2917.

We decided to limit our scope for actual text extraction to the city of Edinburgh. This corresponds to the outcodes EH1 through EH17. The RightMove internal IDs for these outcodes, in order, are 793, 804, 815, 826, 837, 843, 844, 845, 846, 794, 795, 796, 797, 798, 799, 800, 801. We use these internal IDs to construct new URL’s to get lists of search results.

We further modify the URL to get as many results as permitted on a single page. The website offers users the option to see 10, 20, or 50 properties on a page. Unfortunately, we are not able to enter a number larger than 50.

<http://www.rightmove.co.uk/property-to-rent/find.html?locationIdentifier=OUTCODE%5E801&numberOfPropertiesPerPage=50>

As a user executing this query, we see a page of results that look like this:



At the bottom of the page, we see the paging controls:



The web scraper must simulate clicking through to the last page. This is done by further modifying the URL. The query element index specifies the starting index of the first ad on the page. So we request indexes 0, 50, 100, etc. In the URL below, we are requesting the third page of properties.

<http://www.rightmove.co.uk/property-to-rent/find.html?locationIdentifier=OUTCODE%5E801&numberOfPropertiesPerPage=50&index=100>

The first piece of actual “scraping” that is necessary is to extract the ID of each property on a page. This is embedded in the link under the image and the title of the property. The property shown above has an ID of 59453477. We must now “step into” every property on every page of every search query. To visit the page for the property above, we must use the following URL.

<http://www.rightmove.co.uk/property-to-rent/property-59453477.html>

The page for the property has the following components of interest to us:

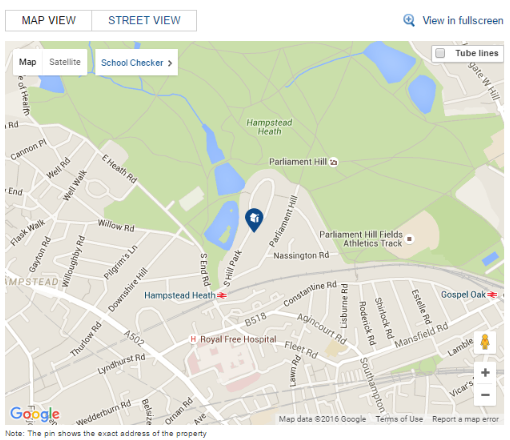
Title and cost:



Metadata:



Location:



To summarize the navigation challenges, here is pseudocode to describe the journey our web scraper takes:

For each outcode:

{

Index = 0

Navigate to the search results using RightMove’s internal ID for the outcode starting with  
ad index Index.

While the search results are not empty:

{

For each ad on the search results page:

{

Extract the property ID.

Navigate to the property page.

Extract description text.

Extract price.

Extract location.

Extract other metadata.

}

}

Index = Index + 50

}

**Robots.txt**

The only robot we that we match is the user agent ‘\*’. Under the rules for this robot, a large number of paths are disallowed. We scrape only the following paths:

/property-to-rent/property-\*.html

/property-to-rent/find.html

These do not match any of the disallowed paths. Therefore, we have not violated any of the rules in the robots.txt file. No features were specifically implemented to adhere to these rules.