WEIR - Report on the second assignment

Matej Bevec
Nejc Hirci
Rok Nikolič

I. Introduction

In this assignment we explore three approaches to structured data extraction from web pages: regular expressions, XPath, and a RoadRunner-like Automatic Web algorithm, which we apply to multiple examples of three different web page types. The following sections outline out implementations and present the obtained results.

II. Web pages

We consider three different web pages categories. These are *RTV.si* news articles and *Overstock.com* product listings, as described in the assignment instructions, and our own example: real estate listings from *Nepremicnine.com*. We implement one solution for each of the web page types, but test each on multiple examples of said page type with different data.

- **RTV.** Each considered page contains a news article from Slovenia's national media house RTV. It can be seen as a "detail view", form which we extract a single *title*, *subtitle*, *author*, *published time*, *lead* and *content*.
- **Overstock.** Each considered page contains a list view of products from the Overstock web store. For each product we extract the *title*, *list price*, (current) *price*, *saving*, *saving percent* and *content* (description).
- **Nepremičnine.** Each considered page contains a list view of real estate listings from the Slovene real estate marketplace *Nepremičnine.com*. For each listing we extract the (estate) *location*, *listing type*, *estate type*, (built) *year*, *price*, *area*, *description* and the *image URL* of the primary photo (see Figure 1).

III. Implementation

A. Regular expressions

The first and most rudimentary approach is to consider the HTML content as plain text and capture desired information with regular expressions. Here we rely on matching the desired element, usually by its class, and extracting the data within using a capture group. A single expression is written to capture a single data item, meaning list views return lists of the same data item type in different records.

Tables I, II and III list regular expressions to capture the desired data items in all three web page types.

B. XPath

Next we extract data by querying the we page's DOM tree with the XPath querying language. In our examples, the expressions generally translate one-to-one from regular expressions, although the syntax becomes much more readable.

Tables I, II and III list XPath queries to capture the desired data items in all three web page types.

C. RoadRunner-like

We base our implementation on the paper *Roadrunner: Towards automatic data extraction from large web sites* [1] which proposes an iterative algorithm for automatically approximating a page's wrapper (i.e. schema, tree). The algorithm progressively refines a template wrapper by resolving mismatches — non-compliances between the grammar defined by the wrapper and the sample page.

Figure 2 in the appendix outlines the algorithm in pseudo code.

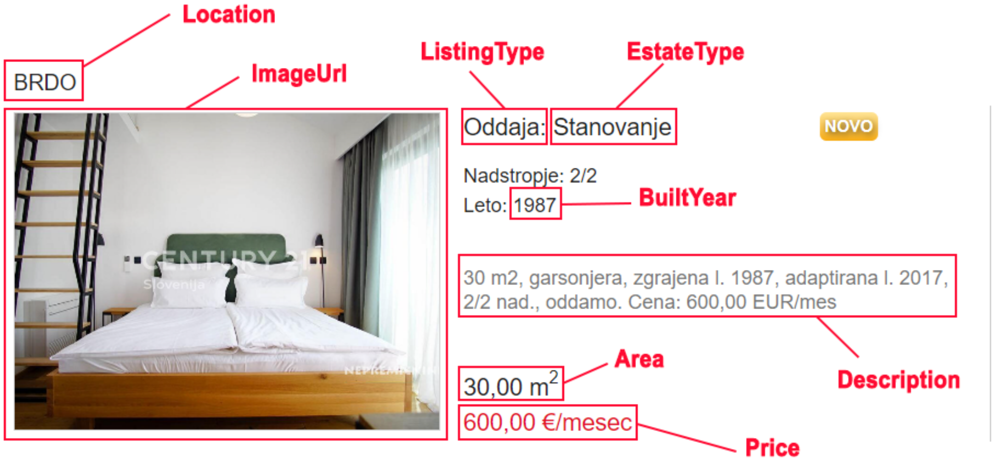
Figures 3, 4 and 5 in the appendix depict the constructed wrappers for the three considered page types. Due to space constraints, we only show the relecant region of the page, where our desired data items are contained.

IV. Conclusion

In the preceding assignment we implemented three methods structured data (wrapper) extraction: regular expressions, XPath and RoadRunner and applied them to different website types. Even the simple regular expression approached proved to be sufficient for our example task, while XPath, next to being more powerful, provided a much more intuitive interface. We additionally explored automatic wrapper extraction with a RoadRunner-like algorithm. While most versatile in theory, we found this approach to be somewhat cumbersome and overly complicated for our particular task.

References

[1] Valter Crescenzi, Giansalvatore Mecca, Paolo Merialdo, et al. 2001. Roadrunner: towards automatic data extraction from large web sites. In *VLDB*. Vol. 1, 109–118.



30 m2, garsonjera, zgrajena l. 1987, adaptirana l. 2017, 2/2 nad., oddamo. Cena: 600,00 EUR/mes

Figure 1: An example of a *Nepremicnine.net* list record and the extracted data items.

Table I: Regular expressions and XPath queries for the RTV case.

Data Item	Regular Expression	XPath Query
title	`r'<h1>\s *(.+?)\s *</h1>'`	`'//h1/text()'`
subtitle	`r'<div class="subtitle">\s *(.+?)\s *</div>'`	`'//div[@class="subtitle"]/text()'`
author	`r'<div class="author-name">\s *(.+?)\s *</div>''`	`'//div[@class="author-name"]/text()'`
published time	`r'<div class="publish-meta">\s *(.+?)\s *<br>'`	`'//div[@class="publish-meta"]/text()'`
lead	`r'<p class="lead">\s *(.+?)\s *</p>'`	`'//p[@class="lead"]/text()'`
content	`r'<article class="article">.*?<p[^>]*>(.*?)</p>[\s \n \t ]*(?:\t |<figure class="mceNonEditable)'`	`'//article[@class="article"]//p/text()'`

Table II: Regular expressions and XPath queries for the Overstock case.

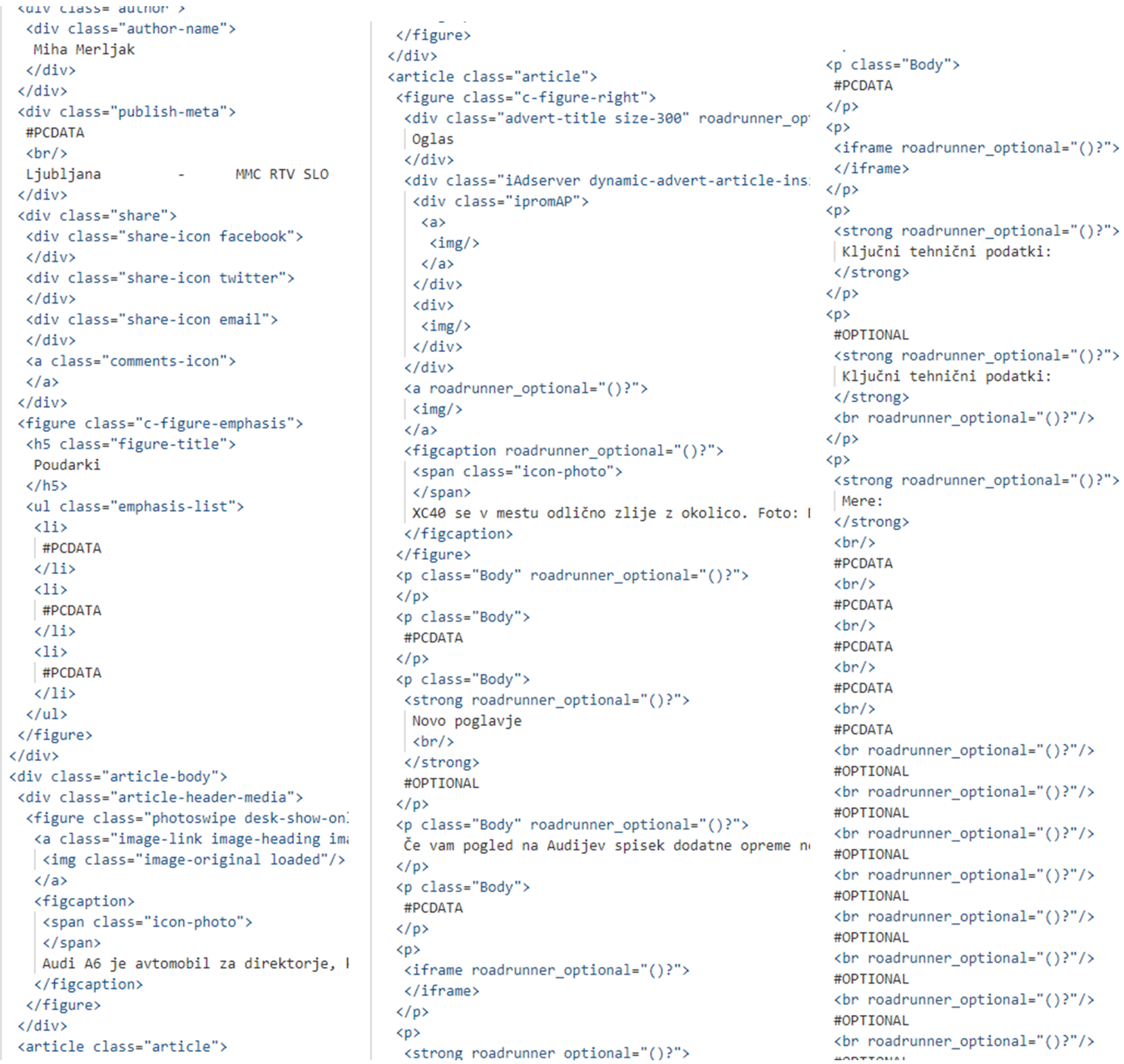
Data Item	Regular Expression	XPath Query
title	`r'<td valign="top">\s *<a href=\s *.+?\s *<b>\s *(.+?)\s *</b></a>'`	`'//td[@valign]/a/b/text()'`
list price	`r'<b>List Price:</b>\s *.+?\s *<s>\s *(.+?)\s *</s>'`	`'//td[@align="left" and @nowrap="nowrap"]/s/text()'`
price	`r'<b>Price:</b>\s *.+?\s *<b>\s *(.+?)\s *</b>'`	`'//td[@align="left" and @nowrap="nowrap"]/span[@class="bigred"]/b/text()'`
saving	`r'<b>You Save:</b>\s *.+?\s *<span class="littleorange">\s *(.+?\s )\s *.+?\s *</span>'`	`'//td[@align="left" and @nowrap="nowrap"]/span[@class="littleorange"]/text()'`
saving percent	`r'<b>You Save:</b>\s *.+?\s *<span class="littleorange">\s *.+?\s *(\s .+?)\s *</span>'`	`'//td[@align="left" and @nowrap="nowrap"]/span[@class="littleorange"]/text()'`
content	`r'<span class="normal">\s *(.+?)\s *<br>'`	`'//td[@valign="top"]/span[@class="normal"]/text()'`

Table III: Regular expressions and XPath queries for the Nepremicnine.net case.

Data Item	Regular Expression	XPath Query
location	`r'<h2.*?<span class="title".*?>(.*?)[<|,]'`	`'//h2//span[@class="title"]/text()'`
listing type	`r'<span class="posr.*?>(.+?):'`	`'//span[contains(@class, "posr")]/text()'`
estate type	`r'<span class="vrsta.*?>(.+?)<'`	`r'//span[contains(@class, "vrsta")]/text()'`
year	`r'<span class="atribut leto.*?<strong>(.+?)<'`	`'//span[contains(@class, "atribut leto")]/strong/text()'`
price	`r'<span class="cena.*?>(.+?)\s '`	`'//span[contains(@class, "cena")]/text()'`
area	`r'<span class="velikost.*?><span></span>(.+?)\s '`	`'//span[contains(@class, "velikost")]/text()'`
description	`r'<div class="kratek.*?itemprop="description.*?>(.+?)<'`	`'//div[contains(@class, "kratek") and contains(@itemprop, "description")]/text()'`
image url	`r'<a.*?data-src="(.*?)"'`	`'//a//img[1][@data-src]/@data-src'`

```
def matching_tags(node_w, node_s):
    """Checks if two nodes are matching tags."""

def recursive_match(node_w, node_s):
    """Recursively matches nodes in the wrapper and sample."""

def discover_tag_iterators(node_list_w, node_list_s, i):
    """Searches for iterators in the wrapper and sample and generalizes
        them to iterators in the wrapper."""

def discover_tag_optionals(node_list_w, node_list_s, i):
    """Searches for optional tags in the wrapper and sample and generalizes
        them to optional tags in the wrapper."""


def run_roadrunner(node_wrapper, node_sample):

        children_w = [el for el in node_wrapper.children]
        children_s = [el for el in node_sample.children]

        i = 0
        while i < len(children_w) and i < len(children_s):
            child_w = children_w[i]
            child_s = children_s[i]

            if matching_tags(child_w, child_s):
                # Tags are the same (node already in wrapper), recursively run roadrunner
                run_roadrunner(child_w, child_s)

            elif "child_w and child_s are both strings":
                if child_w == child_s:
                    pass
                else:
                    children_w[i] = "#PCDATA" # Mark with special token

            else:
                # Tags are not the same

                # Find iterators
                found_iterator, new_children_w = discover_tag_iterators(children_w, children_s, i)

                # If non are found, try optionals
                if not found_iterator:
                    j, children_w = discover_tag_optionals(children_w, children_s, i)
                    if j < 0: break
                    else: i = j
                else:
                    children_w = new_children_w
            i += 1

        # Update the children of the wrapper node
        node_wrapper.clear()
        for child in children_w:
            node_wrapper.append(child)
```

Figure 2: Pseudo code for the RoadRunner algorithm

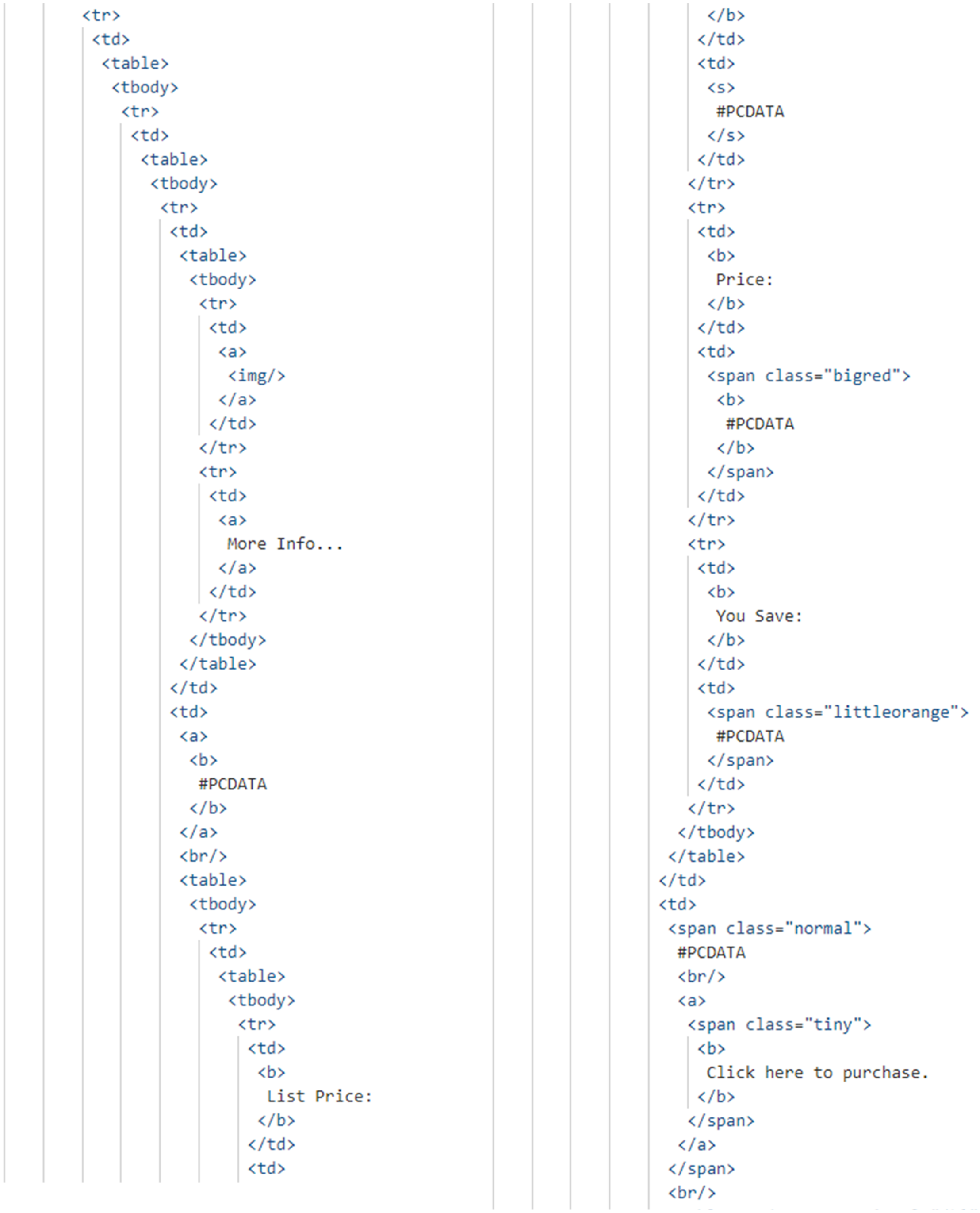


```
 <div class="author">
  <div class="author-name">
   Miha Merljak
  </div>
 </div>
 <div class="publish-meta">
  #PCDATA
  <br/>
  Ljubljana          -       MMC RTV SLO
 </div>
 <div class="share">
  <div class="share-icon facebook">
  </div>
  <div class="share-icon twitter">
  </div>
  <div class="share-icon email">
  </div>
  <a class="comments-icon">
  </a>
 </div>
 <figure class="c-figure-emphasis">
  <h5 class="figure-title">
   Poudarki
  </h5>
  <ul class="emphasis-list">
   <li>
    #PCDATA
   </li>
   <li>
    #PCDATA
   </li>
   <li>
    #PCDATA
   </li>
  </ul>
 </figure>
</div>
<div class="article-body">
 <div class="article-header-media">
  <figure class="photoswipe desk-show-onl
   <a class="image-link image-heading ima
    <img class="image-original loaded"/>
   </a>
   <figcaption>
    <span class="icon-photo">
    </span>
    Audi A6 je avtomobil za direktorje,
   </figcaption>
  </figure>
 </div>
 <article class="article">
```

```
 </figure>
</div>
<article class="article">
 <figure class="c-figure-right">
  <div class="advert-title size-300" roadrunner_op
   Oglas
  </div>
  <div class="iAdserver dynamic-advert-article-ins
   <div class="ipromAP">
    <a>
     <img/>
    </a>
   </div>
   <div>
    <img/>
   </div>
  </div>
  <a roadrunner_optional="()?">
   <img/>
  </a>
  <figcaption roadrunner_optional="()?">
   <span class="icon-photo">
   </span>
   XC40 se v mestu odlično zlije z okolico. Foto:
  </figcaption>
 </figure>
 <p class="Body" roadrunner_optional="()?">
 </p>
 <p class="Body">
  #PCDATA
 </p>
 <p class="Body">
  <strong roadrunner_optional="()?">
   Novo poglavje
   <br/>
  </strong>
  #OPTIONAL
 </p>
 <p class="Body" roadrunner_optional="()?">
  Če vam pogled na Audijev spisek dodatne opreme n
 </p>
 <p class="Body">
  #PCDATA
 </p>
 <p>
  <iframe roadrunner_optional="()?">
  </iframe>
 </p>
 <p>
  <strong roadrunner_optional="()?">
```

```
<p class="Body">
 #PCDATA
</p>
<p>
 <iframe roadrunner_optional="()?">
 </iframe>
</p>
<p>
 <strong roadrunner_optional="()?">
  Ključni tehnični podatki:
 </strong>
</p>
<p>
 #OPTIONAL
 <strong roadrunner_optional="()?">
  Ključni tehnični podatki:
 </strong>
 <br roadrunner_optional="()?"/>
</p>
<p>
 <strong roadrunner_optional="()?">
  Mere:
 </strong>
 <br/>
 #PCDATA
 <br/>
 #PCDATA
 <br/>
 #PCDATA
 <br/>
 #PCDATA
 <br/>
 #PCDATA
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
 #OPTIONAL
 <br roadrunner_optional="()?"/>
```

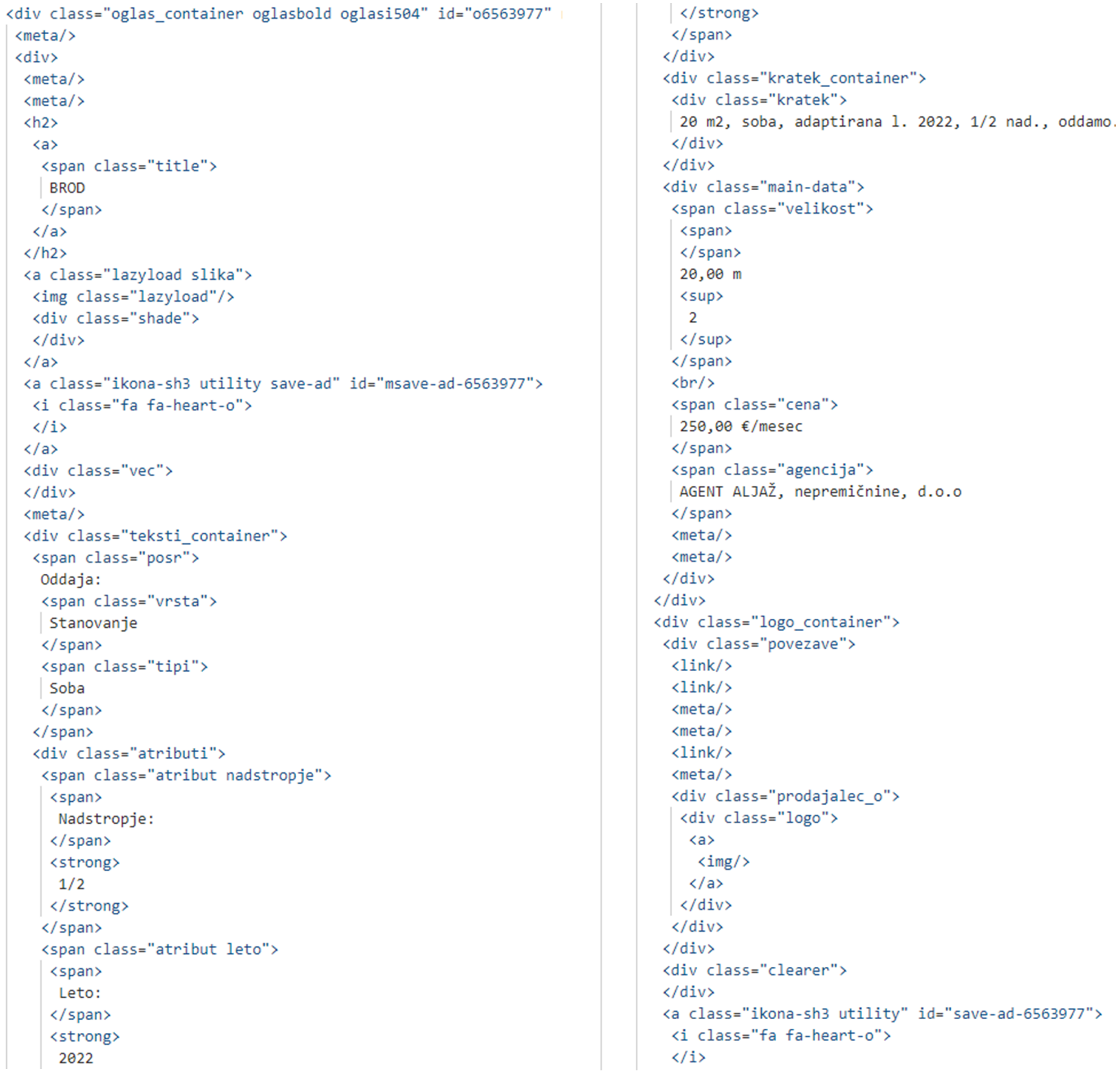
Figure 3: RoadRunner wrapper for the news article section of an RTV page.



```
<tr>
 <td>
  <table>
   <tbody>
    <tr>
     <td>
      <table>
       <tbody>
        <tr>
         <td>
          <table>
           <tbody>
            <tr>
             <td>
              <a>
               <img/>
              </a>
             </td>
            </tr>
            <tr>
             <td>
              <a>
               More Info...
              </a>
             </td>
            </tr>
           </tbody>
          </table>
         </td>
         <td>
          <a>
           <b>
            #PCDATA
           </b>
          </a>
          <br/>
          <table>
           <tbody>
            <tr>
             <td>
              <table>
               <tbody>
                <tr>
                 <td>
                  <b>
                   List Price:
                  </b>
                 </td>
                 <td>
```

```
                  </b>
                 </td>
                 <td>
                  <s>
                   #PCDATA
                  </s>
                 </td>
                </tr>
                <tr>
                 <td>
                  <b>
                   Price:
                  </b>
                 </td>
                 <td>
                  <span class="bigred">
                   <b>
                    #PCDATA
                   </b>
                  </span>
                 </td>
                </tr>
                <tr>
                 <td>
                  <b>
                   You Save:
                  </b>
                 </td>
                 <td>
                  <span class="littleorange">
                   #PCDATA
                  </span>
                 </td>
                </tr>
               </tbody>
              </table>
             </td>
             <td>
              <span class="normal">
               #PCDATA
               <br/>
               <a>
                <span class="tiny">
                 <b>
                  Click here to purchase.
                 </b>
                </span>
               </a>
              </span>
              <br/>
```

Figure 4: RoadRunner wrapper for a single list record in an Overstock page.



```
<div class="oglas_container oglasbold oglasi504" id="o6563977"
 <meta/>
 <div>
  <meta/>
  <meta/>
  <h2>
   <a>
    <span class="title">
     BROD
    </span>
   </a>
  </h2>
  <a class="lazyload slika">
   <img class="lazyload"/>
   <div class="shade">
   </div>
  </a>
  <a class="ikona-sh3 utility save-ad" id="msave-ad-6563977">
   <i class="fa fa-heart-o">
   </i>
  </a>
  <div class="vec">
  </div>
  <meta/>
  <div class="teksti_container">
   <span class="posr">
    Oddaja:
    <span class="vrsta">
     Stanovanje
    </span>
    <span class="tipi">
     Soba
    </span>
   </span>
   <div class="atributi">
    <span class="atribut nadstropje">
     <span>
      Nadstropje:
     </span>
     <strong>
      1/2
     </strong>
    </span>
    <span class="atribut leto">
     <span>
      Leto:
     </span>
     <strong>
      2022
     </strong>
    </span>
   </div>
   <div class="kratek_container">
    <div class="kratek">
     20 m2, soba, adaptirana l. 2022, 1/2 nad., oddamo.
    </div>
   </div>
   <div class="main-data">
    <span class="velikost">
     <span>
     </span>
     20,00 m
     <sup>
      2
     </sup>
    </span>
    <br/>
    <span class="cena">
     250,00 €/mesec
    </span>
    <span class="agencija">
     AGENT ALJAŽ, nepremičnine, d.o.o
    </span>
    <meta/>
    <meta/>
   </div>
  </div>
  <div class="logo_container">
   <div class="povezave">
    <link/>
    <link/>
    <meta/>
    <meta/>
    <link/>
    <meta/>
    <div class="prodajalec_o">
     <div class="logo">
      <a>
       <img/>
      </a>
     </div>
    </div>
   </div>
   <div class="clearer">
   </div>
   <a class="ikona-sh3 utility" id="save-ad-6563977">
    <i class="fa fa-heart-o">
    </i>
```

Figure 5: RoadRunner wrapper for a single list record in an *Nepremicnine.net* page.