Teaching the Teacher: Python
Day 3 – Afternoon: »APIs and scraping«

Damian Trilling

d.c.trilling@uva.nl
@damian0604
www.damiantrilling.net

30 June 2021

Afdeling Communicatiewetenschap
Universiteit van Amsterdam

Today

A first intro to parsing webpages

Modern approaches to web scraping

- XPATH vs CSS selector

Scaling up

Next steps

Everything clear from this morning?

A first intro to parsing webpages

Let's have a look of a webpage with comments and try to understand the underlying structure.

Websites change constantly!

The examples on these slides are meant to illustrate the principles and approaches and are *not* meant as a practical guide for scraping the specific websites mentioned here. Websites change their structure quite regularly, and you cannot assume that scraping code written once keeps working in the future.

Let's have a look of a webpage with comments and try to understand the underlying structure.

Websites change constantly!

The examples on these slides are meant to illustrate the principles and approaches and are *not* meant as a practical guide for scraping the specific websites mentioned here. Websites change their structure quite regularly, and you cannot assume that scraping code written once keeps working in the future.

GeenStijl : Vreemde vogel vogelvrij bij dodenherdenking Dam

http://www.geenstijl.nl/mt/archieven/2014/05/ik_ben_niet_gek_ik_ben_een_ehm_sja_eeeh_wie_het_weet_mag_het

Reader

Google

Broen s01e0... - YouTube | upgrade - L... Ask Ubuntu | Old MS-DOS ...r download | RP account | Apple | Yahoo! | Nieuws | Populair | Import to Mendeley

Twitter ... | Anatom... | http://d... | Beautifu... | Vreemd... | Gezocht... | Web Scr... | Regular ... | regex - ... | python ... | Facebook

GEENSTIJL

Tendentieus. Ongefundeerd & Nodeloos kwetsend

184 files - 236147 leden / 194 km 5076 stijfjozen

Tik hier je zoe | ZOI TIP DE REDACTIE

Vreemde vogel vogelvrij bij dodenherdenking Dam

Vreemde vogel bij dodenherdenking

0:00 / 4:38

Filmpje uit het rauwrealistische archief van Amsterdamse stadscineast Frank Buis. En met 'rauwrealistisch' bedoelen we: de beelden die u niet ziet op de stream-met-nietszeggend-commentaar-in-stemmige-fluistertoon van de Staatsomroep. Vlak voor de Dodenherdenking dook bovenstaande dodo op de Dam op. Plastic Hannibal Lecter maskertje voor de mond. Rare oranje klomp aan de voet. Kogelhulzen in de kroon. Daar wilde de politie wel even mee babbelen, zo vlak voor de aankomst van de koninklijke nazi-nazaat en zijn Junta-meisje. Want doden herdenken op je eigen manier is 1 ding. Maar jezelf daarbij koddig kleden valt buiten de voorschriften van de vrijheid. Wat moesten die 69 kindjes in het buitenlandse gasten-vak wel niet denken? Daarom. Goed dat de politie deze malle meneer en zijn aluminium lunchpakketje eventjes naar een veilige afstand van de vrijheidsviering voerde. (Video: Frank Buis/Royal Press Amsterdam)

Van Rossem | 05-05-14 | 11:11 | Link | 100 reacties |

HEADLINES

05-05

Daar is pathologische leugenaar Rehwinkel weer!

Kijk. Zo leren kindjes in Tsjechie over de EU

Goedkoopste Sony Samsung en iPhone GENOEG = GENOEG www.uwonrede.nl

Gezocht. Dief met tatoeage en motorfiets

Wientjes: 'Hey paupers, de EU is fantastisch hoor'

Anarchie! Wildplassen is ook vrijheid

Vreemde vogel vogelvrij bij dodenherdenking Dam

Aangifte tegen GeenStijl - De Q en A Kloosried

Generaal van Uhm: 'Polderjihadisten zijn

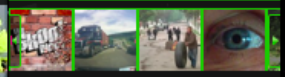
REAQUURSELS

Tsja, ik zou m ook niet in mn buurt willen hebben, die vreemde vogel.

Lepo | 05-05-14 | 11:13

Volgend jaar stropdascontrole voor de heren en hoedjescheck voor de dames. De volledige lijst van goedgekeurde kleding kunt u vinden op Postbus51.nl.

DUMPERT



POWNEWS

POWNEWS - ELKE WERKDAG, 22:45, NED3

Eerder: Delft strijdt tegen Volkert

16:48 Jongens melden zich na mishandeling

16:27 Polio gaat wereld slopen

16:24 Verkiezingsposters plakken niet

15:48 Stemfie = celstraf in Zuid-Afrika

15:36 Update: ontvoeringen bevestigd

15:26 Timmermans sipt op rijm over Roda

15:16 Tientallen migranten verdronken

15:13 Natte vrouwen op supergijbaan

GEENSTIJL TV

DASKAPITAL

16:58 Pensioenpolder jaagt fondsen naar België

REAGUURSELS

Tsja, ik zou m ook niet in mn buurt willen hebben, die vreemde vogel.

Lepo | 05-05-14 | 11:13

Volgend jaar stropdascontrole voor de heren en hoedjescheck voor de dames. De volledige lijst van goedgekeurde kleding kunt u vinden op Postbus51.nl.

Uiteraard bent u extra welkom als u Abercrombie & Fitch draagt.

rara | 05-05-14 | 11:15

Gewoon even de afdeling psychiatrie bellen, wie ze missen: Klaar!

Mazzeltov | 05-05-14 | 11:16

netjes opgelost toch?

--sql error-- | 05-05-14 | 11:16

view-source:https://www.geenstijl.nl/3945571/das_toch_niet_normaal/

Aan de slag | Press This | Atlassian Cloud

```
an class="divider">|</span>
an class="datetime">05-05-14 | 23:34</span>


class="reportcomment" title="Deze reactie is in overtreding met de huisregels."></a>
er>


e>


article class="comment col-xs-12 no-y-padding"
commentid="192631041"
92631041">
ss="anchor-pos" id="cid_192631041"></div>

class="cmt-content"><p>Een VZ800 Marauder, wat een giga kutmotor is dat. Na 10 km. heb je al


r>


pan class="username">tiswat</span>
```

Let's make a plan!

Which elements from the page do we need?

- What do they mean?
- How are they represented in the source code?

How should our output look like?

- What *lists* do we want?
- ...

And how can we achieve this?

A first plan

1. Download the page
2. Try to isolate the comments
3. Hopefully, we have a list of comments now that we can work with.

A first plan

1. Download the page
    - Possibly taking measures to deal with cookie walls, being blocked, etc.
2. Try to isolate the comments
3. Hopefully, we have a list of comments now that we can work with.

A first plan

1. Download the page
    - Possibly taking measures to deal with cookie walls, being blocked, etc.
2. Try to isolate the comments
3. Hopefully, we have a list of comments now that we can work with.

A first plan

1. Download the page
    - Possibly taking measures to deal with cookie walls, being blocked, etc.
2. Try to isolate the comments
    - Do we see any pattern in the source code? $\Rightarrow$ *two weeks ago: if we can see a pattern, we can describe it with a regular expression*
3. Hopefully, we have a list of comments now that we can work with.

A first plan

1. Download the page
    - Possibly taking measures to deal with cookie walls, being blocked, etc.
2. Try to isolate the comments
    - Do we see any pattern in the source code? $\Rightarrow$ *two weeks ago: if we can see a pattern, we can describe it with a regular expression*
3. Hopefully, we have a list of comments now that we can work with.

```
import requests
import re

URL = 'http://www.geenstijl.nl/mt/archieven/2014/05/
    das_toch_niet_normaal.html'

# ugly workaround to circumvent cookie wall, not of interest for today
client = requests.session()
r = client.get(URL)
cookies = client.cookies.items()
cookies.append(('cpc','10'))
response = client.get(URL,cookies=dict(cookies))
# end workaround

# remove line breaks and tabs (for regexp matching later on)
tekst=response.text.replace("\n"," ").replace("\t"," ")

comments=re.findall(r'<div class="cmt-content">(.*?)</div>',tekst)
print("There are",len(comments),"comments")
print("These are the first two:")
print(comments[:2])
```

Some remarks

The regexp

- `.*?` instead of `.*` means *lazy* matching. As `.*` matches everything, the part where the regexp should stop would not be analyzed (*greedy* matching) – we would get the whole rest of the document (or the line, but we removed all line breaks).
- The parentheses in `(.*?)` make sure that the function only returns what's between them and not the surrounding stuff (like `<div>` and `</div>`)

Optimization

- Parse usernames, date, time, ...
- Replace `<p>` tags

Some remarks

The regexp

- `.*?` instead of `.*` means *lazy* matching. As `.*` matches everything, the part where the regexp should stop would not be analyzed (*greedy* matching) – we would get the whole rest of the document (or the line, but we removed all line breaks).
- The parentheses in `(.*?)` make sure that the function only returns what's between them and not the surrounding stuff (like `<div>` and `</div>`)

Optimization

- Parse usernames, date, time, ...
- Replace `<p>` tags

Some remarks

The regexp

- `.*?` instead of `.*` means *lazy* matching. As `.*` matches everything, the part where the regexp should stop would not be analyzed (*greedy* matching) – we would get the whole rest of the document (or the line, but we removed all line breaks).
- The parentheses in `(.*?)` make sure that the function only returns what's between them and not the surrounding stuff (like `<div>` and `</div>`)

Optimization

- Parse usernames, date, time, ...
- Replace `<p>` tags

Doing this with other sites?

- It's basically puzzling with regular expressions.
- Look at the source code of the website to see how well-structured it is.

Modern approaches to web scraping

OK, but this surely can be doe more elegantly? Yes!

Others have written these regular expressions for you!

Very few edge cases aside (broken pages, for instance), you do not write these (low-level) regular expressions yourself but use existing packages that let you describe the position of some content within a HTML file with an easier (high-level) syntax, so-called CSS Selectors and/or XPATHs (two new languages next to regexp, yeah![1])

[1] I promise they are easier!

OK, but this surely can be doe more elegantly? Yes!

Others have written these regular expressions for you!

Very few edge cases aside (broken pages, for instance), you do not write these (low-level) regular expressions yourself but use existing packages that let you describe the position of some content within a HTML file with an easier (high-level) syntax, so-called CSS Selectors and/or XPATHs (two new languages next to regexp, yeah![1])

[1] I promise they are easier!

Scraping

The Geenstijl-example

- Worked well (and we could do it with the knowledge we already had)
- But we can also use existing parsers (that can interpret the structure of the html page)
- especially when the structure of the site is more complex

Scraping

The Geenstijl-example

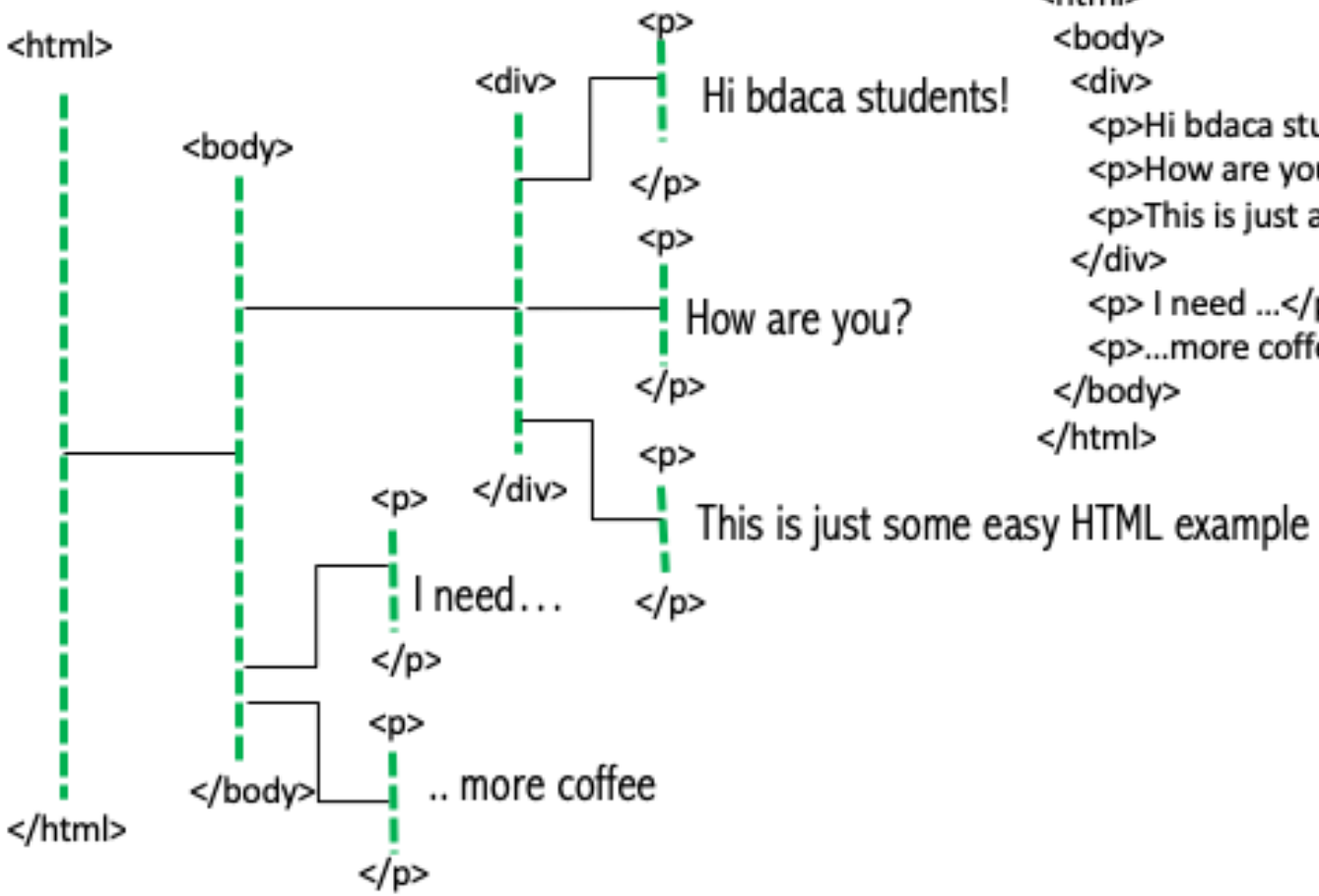
- Worked well (and we could do it with the knowledge we already had)
- But we can also use existing parsers (that can interpret the structure of the html page)
- especially when the structure of the site is more complex

Scraping

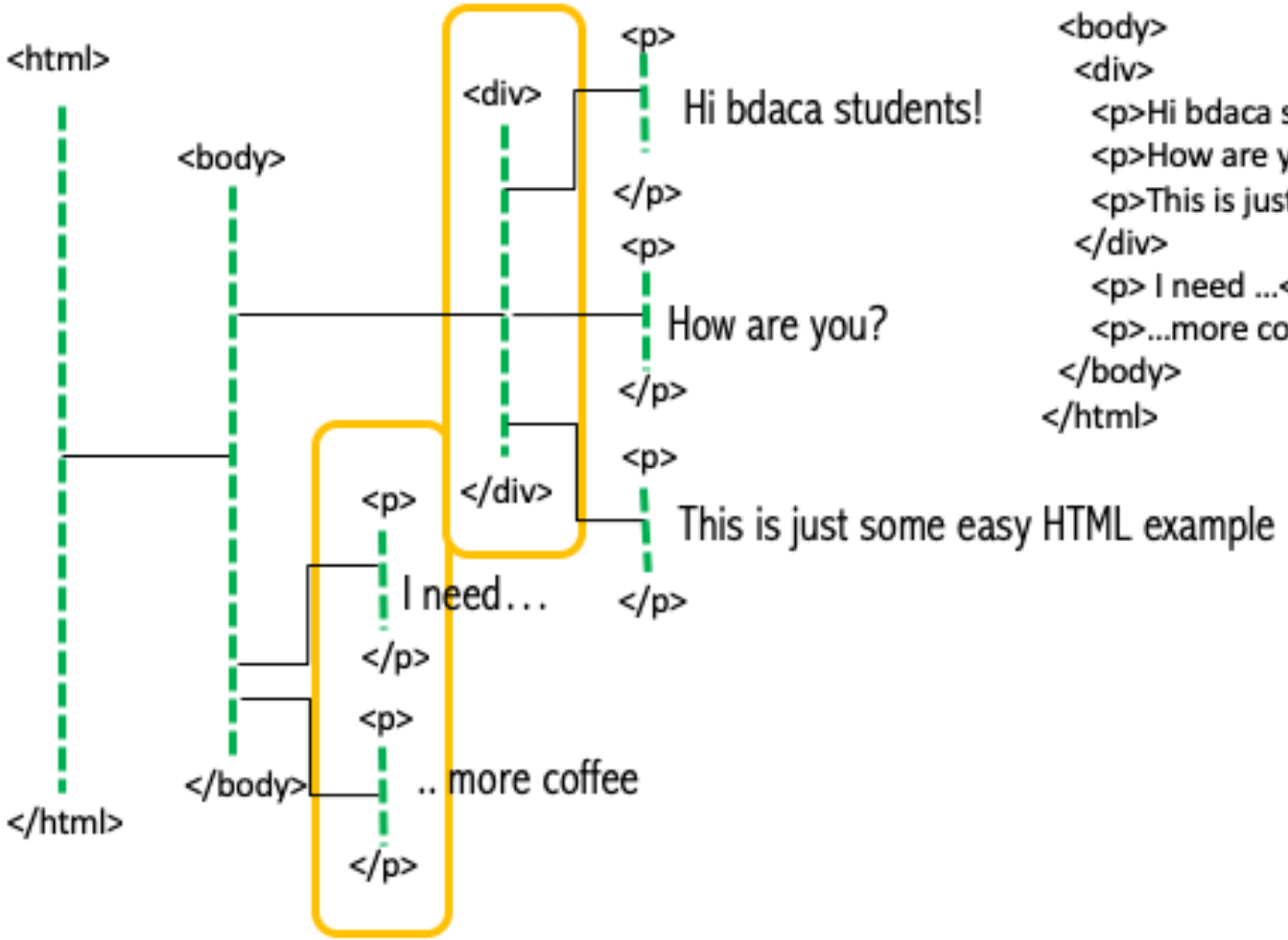
The Geenstijl-example

- Worked well (and we could do it with the knowledge we already had)
- But we can also use existing parsers (that can interpret the structure of the html page)
- especially when the structure of the site is more complex

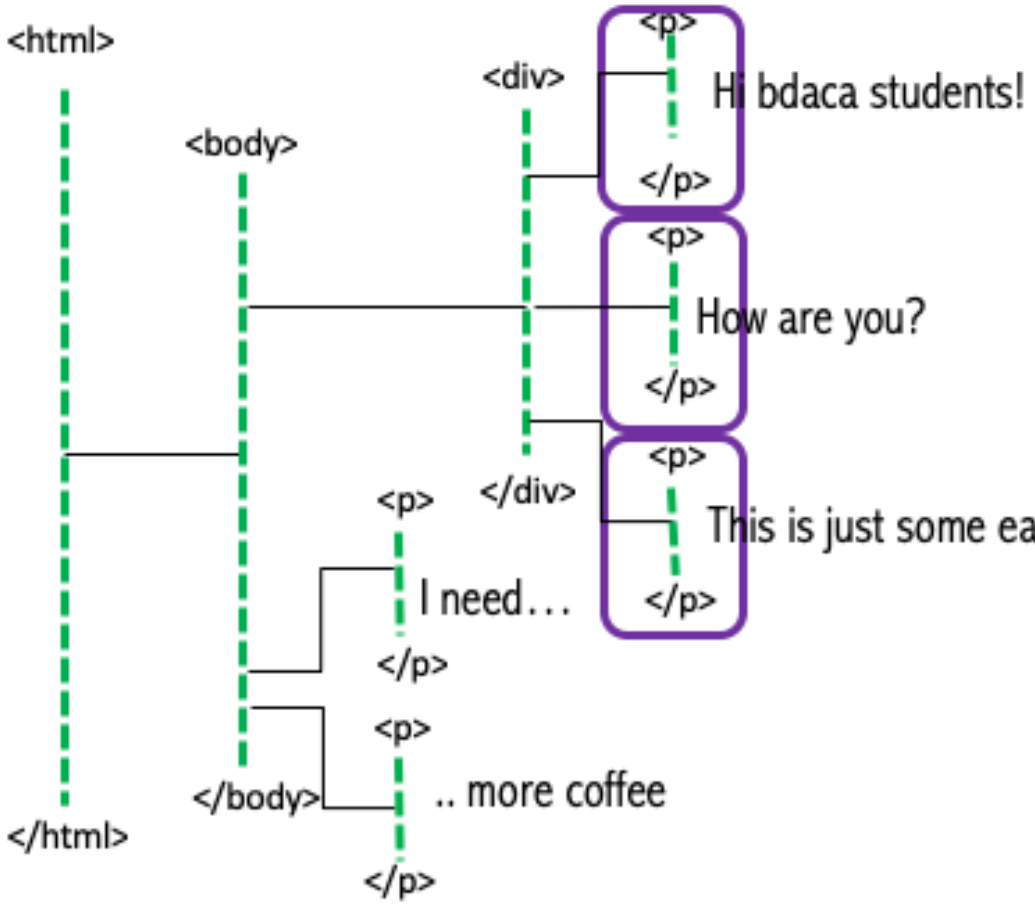
```
<html>
 <body>
  <div>
   <p>Hi bdaca students!</p>
   <p>How are you?</p>
   <p>This is just an easy HTML example</p>
  </div>
   <p> I need …</p>
   <p>…more coffee</p>
 </body>
</html>
```



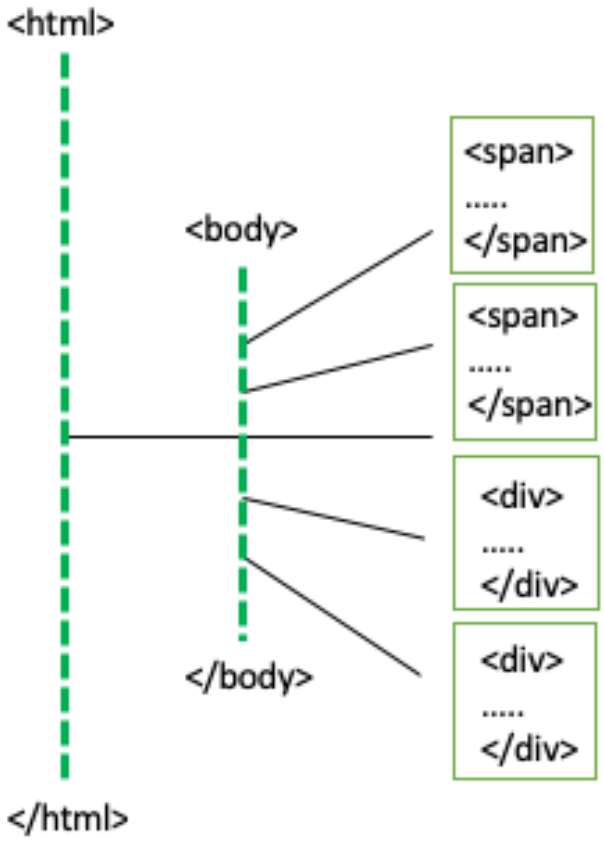

```
<html>
 <body>
  <div>
   <p>Hi bdaca students!</p>
   <p>How are you?</p>
   <p>This is just an easy HTML example</p>
  </div>
   <p> I need ...</p>
   <p>...more coffee</p>
 </body>
</html>
```



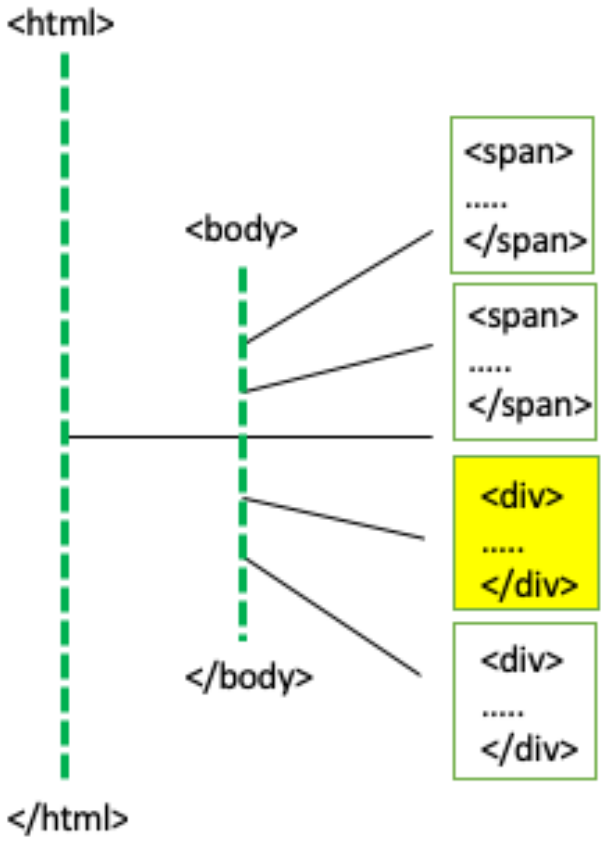

```
<html>
 <body>
  <div>
   <p>Hi bdaca students!</p>
   <p>How are you?</p>
   <p>This is just an easy HTML example</p>
  </div>
   <p> I need ...</p>
   <p>...more coffee</p>
 </body>
</html>
```




```
<html>
 <body>
  <div>
   <p>Hi bdaca students!</p>
   <p>How are you?</p>
   <p>This is just an easy HTML example</p>
  </div>
   <p> I need ...</p>
   <p>...more coffee</p>
 </body>
</html>
```



XPATH = “/html/body/div[1]" refers to which element?



XPATH = “/html/body/div[1]" refers to which element?

What do we need?

- the URL (of course)
- the XPATH of the element we want to scrape (you'll see in a minute what this is)

The following example is based on https://www.kieskeurig.nl/smartphone/product/3518001-samsung-galaxy-a5-2017-goud/reviews. It uses the module `lxml`

What do we need?

- the URL (of course)
- the XPATH of the element we want to scrape (you'll see in a minute what this is)

The following example is based on https://www.kieskeurig.nl/smartphone/product/3518001-samsung-galaxy-a5-2017-goud/reviews. It uses the module `lxml`

What do we need?

- the URL (of course)
- the XPATH of the element we want to scrape (you'll see in a minute what this is)

The following example is based on https://www.kieskeurig.nl/smartphone/product/3518001-samsung-galaxy-a5-2017-goud/reviews. It uses the module `lxml`

Samsung Galaxy A5 (2017) zwart / 16 GB - Reviews

Reviewscore | 8,7 Lees 1049 reviews | Schrijf een review

Favorieten | Vergelijk | Prijsalert

6 Afbeeldingen

Testpanel score 8,2 - april 2017

Laagste prijzen -1% prijsdaling

Midimedia	€ 267,00
Belsimpel.nl	€ 275,00
Mobiel.nl	€ 279,00

Specificaties

- 4.7 inch scherm
- 16 GB opslag
- Android

Bekijk 3 andere uitvoeringen

Samsung Galaxy A5 (2017) zwart / 16 GB

Prijzen (9) | Overzicht | Abonnementen | Specificaties | Reviews (1049)

Review type: **Consumenten reviews (1049)**

Gemiddelde beoordeling

8,7

De gemiddelde beoordeling is gebaseerd op de 1049 reviews die zijn achtergelaten op Kieskeurig.nl.

Help anderen en geef jouw mening over dit product. We verloten iedere maand onder alle reviewers 2 x € 250,-.

Schrijf een review

Kieskeurig.nl review criteria:

Geluidskwaliteit	Goed
Mogelijkheden	Goed
Gebruiksgemak	Uitstekend
Vormgeving	Uitstekend
Degelijkheid	Goed

Echte reviews van echte gebruikers, door ons gecontroleerd op betrouwbaarheid.

Sorteer op: Meest nuttig eerst +

BartTilburg
21-03-2017

· Product paar weken in bezit

Verliefd op de Samsung Galaxy A5 2017

✓ Ja, ik beveel dit product aan

Wat een fenomenale smartphone is de nieuwe Samsung Galaxy A5 2017 geworden, lees hier waarom.

Bij het openen van de verpakking valt direct op dat dit toestel niet onder doet voor een smartphone uit het duurste segment. Ik heb gekozen voor de kleur sand/gold, een mooie champagneachtige gouden kleur die erg opvalt. Het toestel heeft een aluminium behuizing en zowel de voorkant als de achterkant van de Samsung Galaxy A5 2017 zijn geheel van glas en er is zelfs een

Algemene score

Reviewscore | 10,0

Review criteria:

Geluidskwaliteit	Uitstekend
Mogelijkheden	Uitstekend
Gebruiksgemak	Uitstekend

Verliefd op de Samsung Galaxy A5 2017

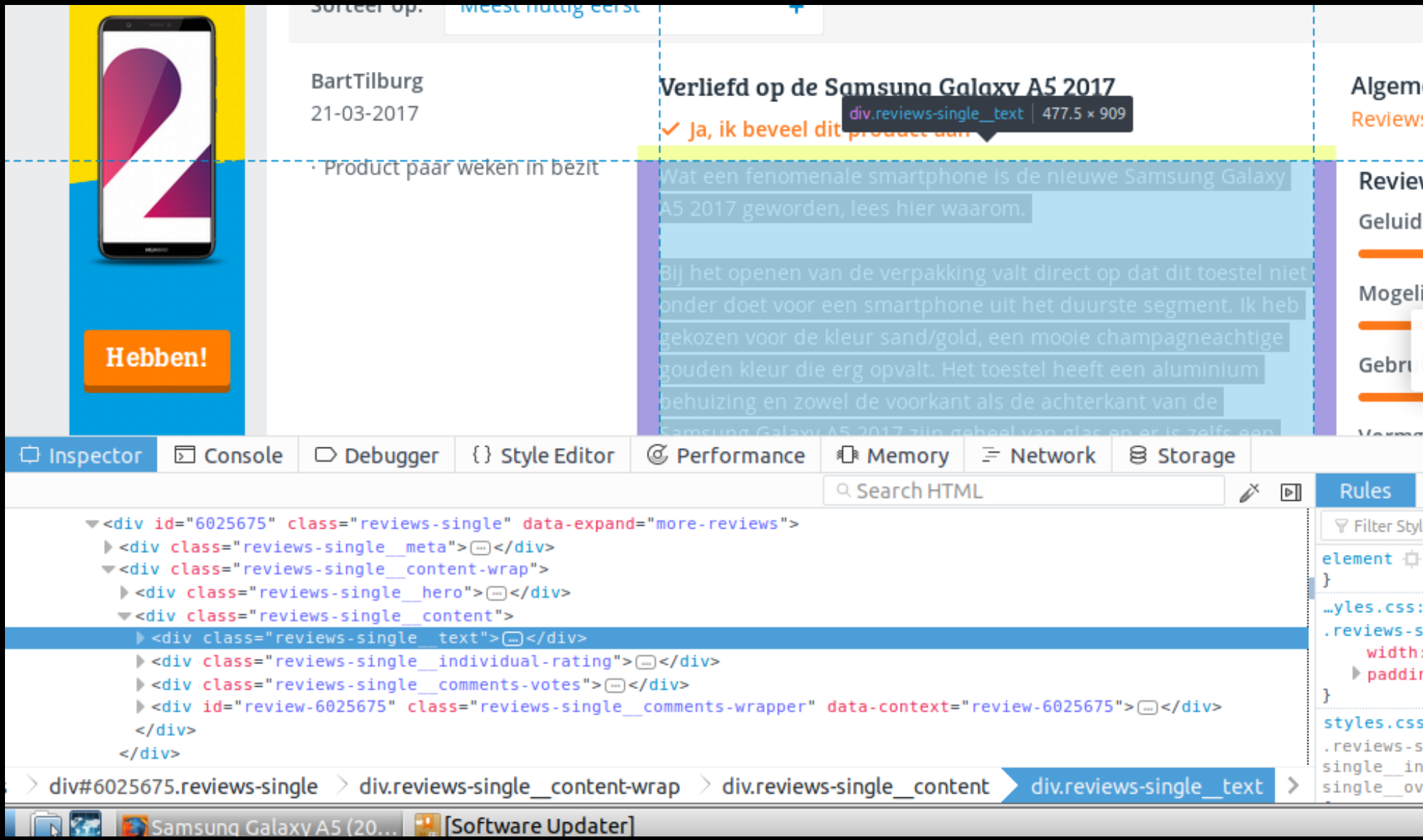
✓ Ja, ik beveel dit product aan

Wat een fenomenale smartphone is de nieuwe Samsung Galaxy A5 2017 geworden, lees hier waarom.

Bij het openen van de verpakking valt direct op dat dit toestel niet onder doet voor een smartphone uit het duurste segment. Ik heb gekozen voor de kleur sand/gold, een mooie champagneachtige gouden kleur die erg opvalt. Het toestel heeft een aluminium behuizing en zowel de voorkant als de achterkant van de Samsung Galaxy A5 2017 zijn gehee
vingerafdruksensor aanwezig. Dit a
premium look and feel, prachtig! In
accessoires die je nodig hebt zoals
hoofdtelefoon én de bijzondere USI
USB-C kabel kan je dit toestel versn
toestel heeft echt alles aan boord wat je mag verwachten en zelfs

Algemene scor
Reviewscore | 10

Review criteria
Geluidskwaliteit
Mogelijkheden
Gebruiksgemak

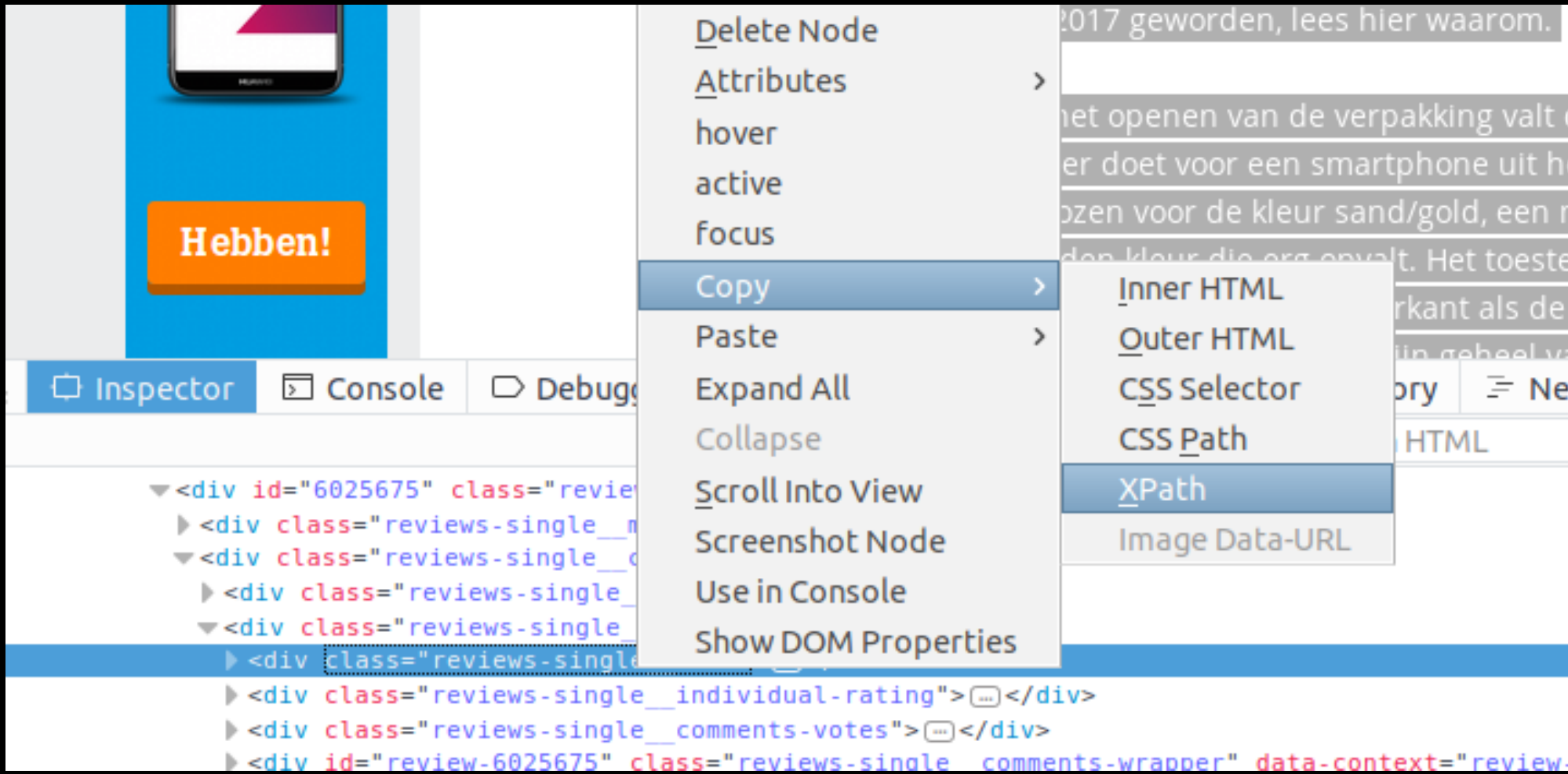

Hebben!
BartTilburg
21-03-2017
· Product paar weken in bezit
Verliefd op de Samsung Galaxy A5 2017
Ja, ik beveel dit product aan
div.reviews-single__text | 477.5 × 909
Wat een fenomenale smartphone is de nieuwe Samsung Galaxy A5 2017 geworden, lees hier waarom.
Bij het openen van de verpakking valt direct op dat dit toestel niet onder doet voor een smartphone uit het duurste segment. Ik heb gekozen voor de kleur sand/gold, een mooie champagneachtige gouden kleur die erg opvalt. Het toestel heeft een aluminium behuizing en zowel de voorkant als de achterkant van de
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Search HTML
Rules

…

…

…
…
…
…

element
}
.reviews-s
width:
paddir
}
styles.css
.reviews-s
single__in
single__ov
div#6025675.reviews-single
div.reviews-single__content-wrap
div.reviews-single__content
div.reviews-single__text
Samsung Galaxy A5 (20…
[Software Updater]


Hebben!
Delete Node
Attributes
hover
active
focus
Copy
Paste
Expand All
Collapse
Scroll Into View
Screenshot Node
Use in Console
Show DOM Properties
Inner HTML
Outer HTML
CSS Selector
CSS Path
XPath
Image Data-URL
Inspector
Console
HTML
<div id="6025675" class="revie
<div class="reviews-single__
<div class="reviews-single__
<div class="reviews-single_
<div class="reviews-single_
<div class="reviews-single
…
…
<div id="review-6025675" class="reviews-single__comments-wrapper" data-context="review

Note that each *tag* (such as `div`) can have *attributes* (such as `class` or `id`!)

But you can also create your XPATH yourself

There are multiple different XPATHs to address a specific element.

Some things to play around with:

- // means 'arbitrary depth' (=may be nested in many higher levels)
- * means 'anything'. (p[2] is the second paragraph, p[*] are all
- If you want to refer to a specific attribute of a HTML tag, you can use @. For example, every *[@id="reviews-container"] would grap a tag like <div id=reviews-container'' class=''user-content'
- Let the XPATH end with /text() to get all text
- Have a look at the source code (via 'inspect elements') of the web page to think of other possible XPATHs!

But you can also create your XPATH yourself

There are multiple different XPATHs to address a specific element.

Some things to play around with:

- `//` means 'arbitrary depth' (=may be nested in many higher levels)
- `*` means 'anything'. (`p[2]` is the second paragraph, `p[*]` are all
- If you want to refer to a specific attribute of a HTML tag, you can use `@`. For example, every `*[@id="reviews-container"]` would grap a tag like `<div id=reviews-container'' class=''user-content'`
- Let the XPATH end with `/text()` to get all text
- Have a look at the source code (via 'inspect elements') of the web page to think of other possible XPATHs!

But you can also create your XPATH yourself

There are multiple different XPATHs to address a specific element.

Some things to play around with:

- `//` means 'arbitrary depth' (=may be nested in many higher levels)
- `*` means 'anything'. (`p[2]` is the second paragraph, `p[*]` are all
- If you want to refer to a specific attribute of a HTML tag, you can use `@`. For example, every `*[@id="reviews-container"]` would grap a tag like `<div id=reviews-container'' class='''user-content'`
- Let the XPATH end with `/text()` to get all text
- Have a look at the source code (via 'inspect elements') of the web page to think of other possible XPATHs!

But you can also create your XPATH yourself

There are multiple different XPATHs to address a specific element.

Some things to play around with:

- `//` means 'arbitrary depth' (=may be nested in many higher levels)
- `*` means 'anything'. (`p[2]` is the second paragraph, `p[*]` are all
- If you want to refer to a specific attribute of a HTML tag, you can use `@`. For example, every `*[@id="reviews-container"]` would grap a tag like `<div id=reviews-container'' class='''user-content'`
- Let the XPATH end with `/text()` to get all text
- Have a look at the source code (via 'inspect elements') of the web page to think of other possible XPATHs!

But you can also create your XPATH yourself

There are multiple different XPATHs to address a specific element.

Some things to play around with:

- `//` means 'arbitrary depth' (=may be nested in many higher levels)
- `*` means 'anything'. (`p[2]` is the second paragraph, `p[*]` are all
- If you want to refer to a specific attribute of a HTML tag, you can use `@`. For example, every `*[@id="reviews-container"]` would grap a tag like `<div id=reviews-container'' class='''user-content'`
- Let the XPATH end with `/text()` to get all text
- Have a look at the source code (via 'inspect elements') of the web page to think of other possible XPATHs!

Let's test it!

https://www.kieskeurig.nl/wasmachine/product/2483630-siemens-wmn16t3471/reviews

Let's scrape them!

```
1  from lxml import html
2  import requests
3
4  response = requests.get('https://www.kieskeurig.nl/wasmachine/product
       /2483630-siemens-wmn16t3471/reviews')
5  tree = html.fromstring(response.text)
6
7  # we extract all relevant elements using their XPATH
8  reviews = tree.xpath('//div[@class="reviews-single__text"]')
9  # alternatively, we can use their CSS selector:
10 # reviews = tree.cssselect("div.reviews-single__text")
11
12 # but we don't want the elements, we want their text
13 review_texts = [e.text_content().strip() for e in reviews]
14
15 print (f"{len(reviews)} reviews scraped. Showing the first 60 characters
       :")
16 for i, review in enumerate(review_texts):
17     print(f"Review {i}: {review[:60]}")
```

The output – perfect!

```
20 reviews scraped. Showing the first 60 characters:
Review 0 : Siemens WMN16T3471 nu 4 maanden in gebruik in massagesalon.
Review 1 : Na een eerder positief review kort na aankoop nu een bijgest
Review 2 : Helaas ben ik teleurgesteld in dit product wegens de navolge
Review 3 : Ik ben heel blij met mijn nieuwe wasmachine:

Wat is hij st
Review 4 : Ik heb de wasmachine nu net een paar dagen in huis en heb al
Review 5 : Na 25 jaar hebben we afscheid moeten nemen van onze degelijk
```

Modern approaches to web scraping

XPATH vs CSS selector

Two ways of expressing the same thing

```
# we extract all relevant elements using their XPATH
reviews = tree.xpath('//div[@class="reviews-single__text"]')
```

```
# alternatively, we can use their CSS selector:
reviews = tree.cssselect("div.reviews-single__text")
```

- partly a matter of personal preferences
- Table 12.1 in the book shows both
- CSS selectors are often easier to write (and more modern)
- XPATHs are more straight-forward for describing the hierarchical position of an object
- there are some cases that cannot be described as CSS selector

$\Rightarrow$ Many people us CSS selectors by default and restort to XPATHs if necessary

Two ways of expressing the same thing

```
# we extract all relevant elements using their XPATH
reviews = tree.xpath('//div[@class="reviews-single__text"]')
```

```
# alternatively, we can use their CSS selector:
reviews = tree.cssselect("div.reviews-single__text")
```

- partly a matter of personal preferences
- Table 12.1 in the book shows both
- CSS selectors are often easier to write (and more modern)
- XPATHs are more straight-forward for describing the hierarchical position of an object
- there are some cases that cannot be described as CSS selector

$\Rightarrow$ Many people us CSS selectors by default and restort to XPATHs if necessary

Scaling up

But this was on *one* page only, right?

Next step: Repeat for each relevant page.

Possibility 1: Based on url schemes

If the url of one review page is

https://www.hostelworld.com/hosteldetails.php/ClinkNOORD/Amsterdam/93919/reviews?page=2

...then the next one is probably?

⇒ you can construct a list of all possible URLs:

```
1 baseurl = 'https://www.hostelworld.com/hosteldetails.php/ClinkNOORD/
      Amsterdam/93919/reviews?page='
2
3 allurls = [f"baseurl{i+1}" for i in range(20)]
```

But this was on *one* page only, right?

Next step: Repeat for each relevant page.

Possibility 1: Based on url schemes

If the url of one review page is

https://www.hostelworld.com/hosteldetails.php/ClinkNOORD/Amsterdam/93919/reviews?page=2

...then the next one is probably?

$\Rightarrow$ you can construct a list of all possible URLs:

```
1 baseurl = 'https://www.hostelworld.com/hosteldetails.php/ClinkNOORD/
      Amsterdam/93919/reviews?page='
2
3 allurls = [f"baseurl{i+1}" for i in range(20)]
```

But this was on *one* page only, right?

Next step: Repeat for each relevant page.

Possibility 2: Based on XPATHs or CSS Selectors

Use XPATH to get the url of the next page (i.e., to get the link that you would click to get the next review)

Recap

General idea

1. Identify each element by its XPATH or CSS Selector (look it up in your browser)
2. Read the webpage into a (looooooong) string
3. Use the XPATH or CSS Selectors to extract the relevant text into a list (with a module like lxml)
4. Do something with the list (preprocess, analyze, save)
5. Repeat

Alternatives: scrapy, beautifulsoup, regular expressions, ...

Last remarks

There is often more than one way to specify an XPATH

1. Sometimes, you might want to use a different suggestion to be able to generalize better (e.g., using the *attributes* rather than the *tags*)
2. in that case, it makes sense to look deeper into the structure of the HTML code, for example with "Inspect Element" and use that information to play around with different possibilities

Console | Debugger | Style Editor | Performance | Memory | Network | Storage

Search HTML

```
id="6025675" class="reviews-single" data-expand="more-reviews">
 class="reviews-single__meta">…</div>
 class="reviews-single__content-wrap">
iv class="reviews-single__hero">…</div>
iv class="reviews-single__content">
<div class="reviews-single__text">…</div>
<div class="reviews-single__individual-rating">…</div>
<div class="reviews-single__comments-votes">…</div>
<div id="review-6025675" class="reviews-single__comments-wrapper" data-context="review-6025675">…</div>
div>
v>
```

75.reviews-single > div.reviews-single__content-wrap > div.reviews-single__content > div.reviews-single__text

Samsung Galaxy A5 (20... | [Software Updater]

Next steps

Friday

- Write a scraper for a website of your choice!
- Choose an easy site where you do not have to log on and where there is no dynamically generated content (if you want to do that, you need to use `selenium` – see book.)

Every scraper is different!

Scraping can be difficult, but it is also one of the most important data collection methods. Chances are very high you'll need it as a part of your final project. It can make sense to already start to write a scraper at home, so that you can ask specific questions on Friday.

Friday

- Write a scraper for a website of your choice!
- Choose an easy site where you do not have to log on and where there is no dynamically generated content (if you want to do that, you need to use `selenium` – see book.)

Every scraper is different!

Scraping can be difficult, but it is also one of the most important data collection methods. Chances are very high you'll need it as a part of your final project. It can make sense to already start to write a scraper at home, so that you can ask specific questions on Friday.

Shell Wijzig Weergave Venster Help

Za 11 mrt. 08:51 Damian T

Restaurants in Amsterdam ...

https://www.iens.nl/stad/amsterdam/19088

Aan de slag Press This Stan - Interfaces Atlassian Cloud

Yums! Verzilver ze hier

BEKIJK DE RESTAURANTS

Proef Italie!

BEKIJK DE RESTAURANTS

damian — ipython3 — 80×26

```
In [35]: ratings = tree.xpath("//*[@class='rating-ratingValue']/text()")

In [36]: restaurants = tree.xpath("//*[@class='restaurantSelection-name']/text()
")

In [37]: len(ratings)
Out[37]: 10

In [38]: len(restaurants)
Out[38]: 10

In [39]: for i in range(10):
    ...:     print(restaurants[i].strip(),ratings[i])
    ...:
Risto Enoteca PepeNero 9.3
Cafe Sage 9.2
The Lobby Fizeaustraat Restaurant & Bar (Hotel V) 9.1
Brasserie Ambassade 9.1
Tulsi Indian restaurant 9.1
Zaza's 9
Jacobsz 9
Bistrot des Alpes 9
Restaurant Jaspers 8.9
En Pluche 8.9

In [40]:
```

rdam Voor jou geselecteerd

De Vrienden Live

FRANS

8.5 /10

```
</a>
<div
</div>
</div>
<div class="restaurantSelection-rateContainer">
<div class="restaurantSelection-avgRate rating">
  <span class="rating-ratingValue">8.6</span>
  /10
</div>
```

Regels Berekend Animaties

Stijlen filteren

Pseudo-elementen

Dit element

```
element {
}
.restaurantSelection .restaurantSelection-avgRate {
    margin: 8px 0;
    text-align: right;
}
.rating {
    font-size: 11px;
```

per-column.lis... li.restaurantSelectionWrapper-item.resta... div.restaurantSelection-rateContainer div.restaurantSelection-avgRate.rating

CSS JS Beveiliging Logboekregistratie Server

Uitvoer filter

Mutation Events wordt niet meer ondersteund. Gebruik in plaats daarvan MutationObserver. ContentVe

ut: geen hoofdelement gevonden Locatie: https://api.realytics.io/event/track?cb=1489218342118 Regelnummer 1, kolom 1:

naar document.write() van een asynchroon geladen extern script is genegeerd. conversi