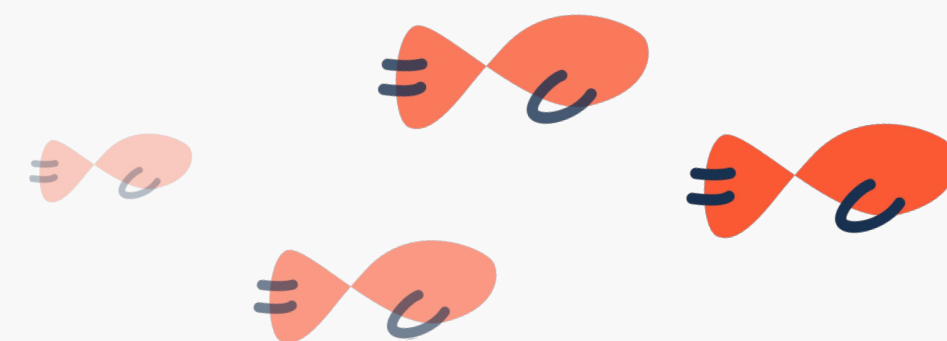




neue fische

School and Pool for Digital Talent

Intro to Web Scraping with BeautifulSoup



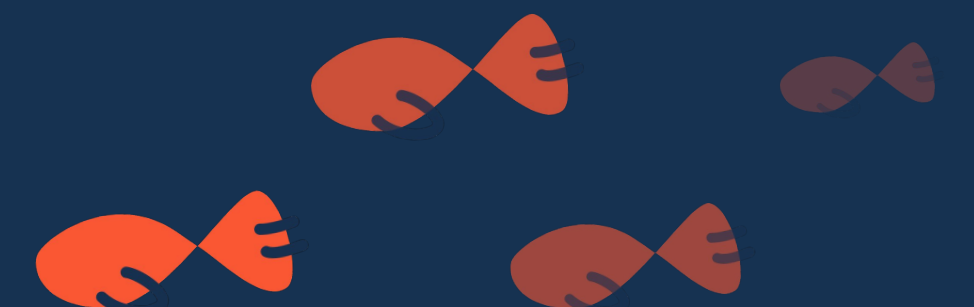
Code of Conduct

We expect all participants to our events and community to abide to this code of conduct:

LadyNerds Code of Conduct (<http://bit.ly/LadyNerds-CoC>).

We follow the **LadyNerds Code of Conduct** because we are dedicated to providing a safe, inclusive, welcoming, and harassment-free space and experience for all members and guests, regardless of gender identity and expression, sexual orientation, disability, physical appearance, socioeconomic status, body size, ethnicity, nationality, level of experience, age, or religion (or lack thereof).

The Code of Conduct exists because of that dedication. We do not tolerate harassment in any form and we prioritise marginalised people's safety over privileged people's comfort.



Team

Eike Rogall

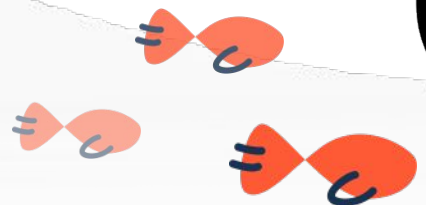
- Biologist
- Coach Data Science Bootcamp
- loves photography, football and his dog Knut



Team

Mia Reimer

- Marine Geologist
- Coach Data Science Bootcamp
- loves table soccer, knitting and eating ice cream



>> neue fische

School and Pool for Digital Talent

- founded 2018 by Dalia Das
- large alumni and partner network



12 Week Bootcamps:

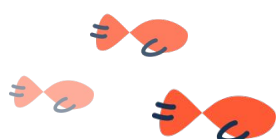
Web Development

Data Science

Data Analytics

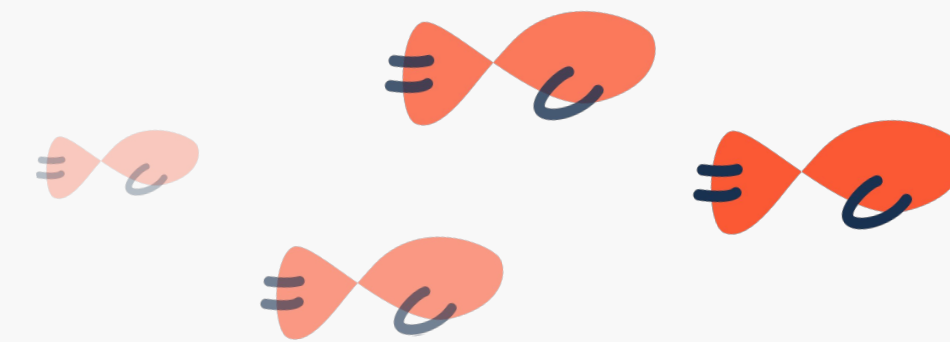
Java Development

Cloud Development



neuefische - upcoming Data Science Bootcamps

- Data Science Bootcamps:
 - 29.11. and January 2022
- for more details of our Bootcamps, checkout our website:
 - neuefische.de
- or get in touch with us:
 - studienberatung@neuefische.de



Agenda

About

At the beginning...

Who are we and
what is neufische?

HTML

**...shortly
afterwards...**

What is HTML and
how does it help us
in web scraping?

Beautiful Soup

**...roughly second
half...**

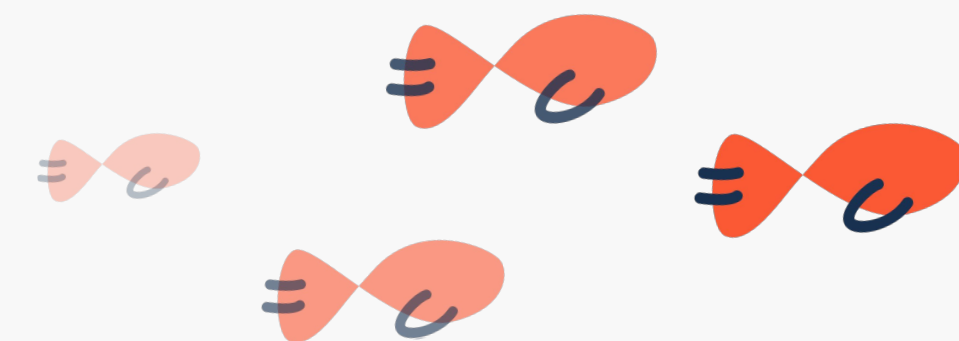
How is soup related
to web scraping and
why is it beautiful?

Q&A

**...when everyone is
enlightened.**

Any more questions?
Ask us anything.

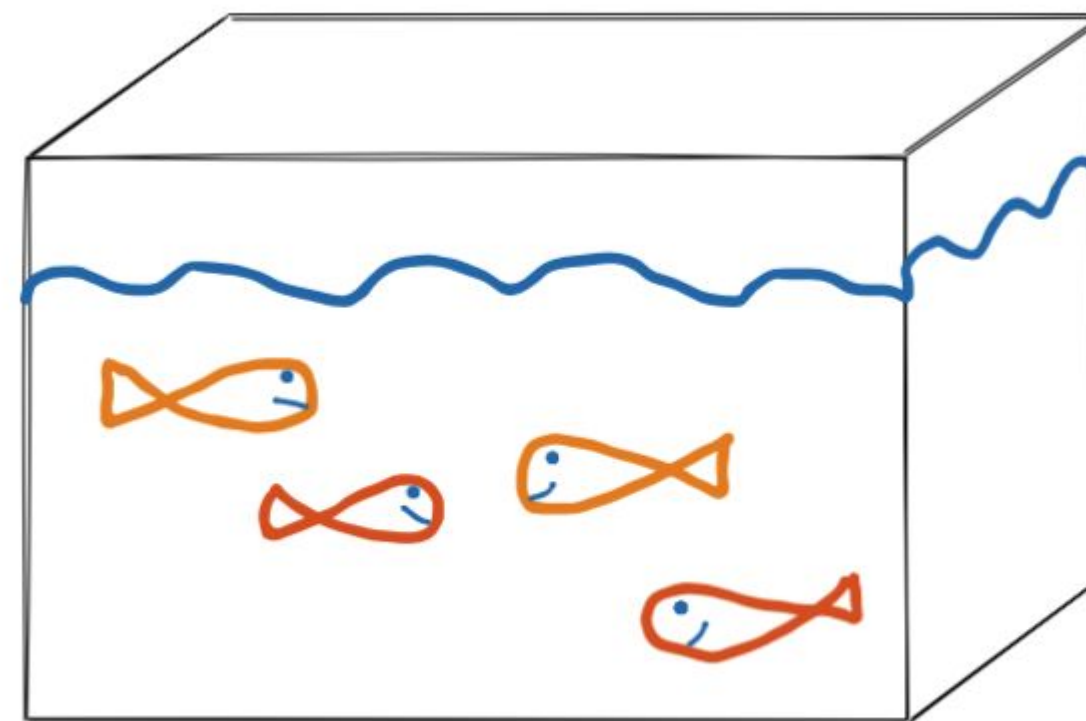
Once upon a time...



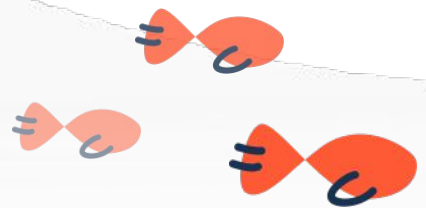
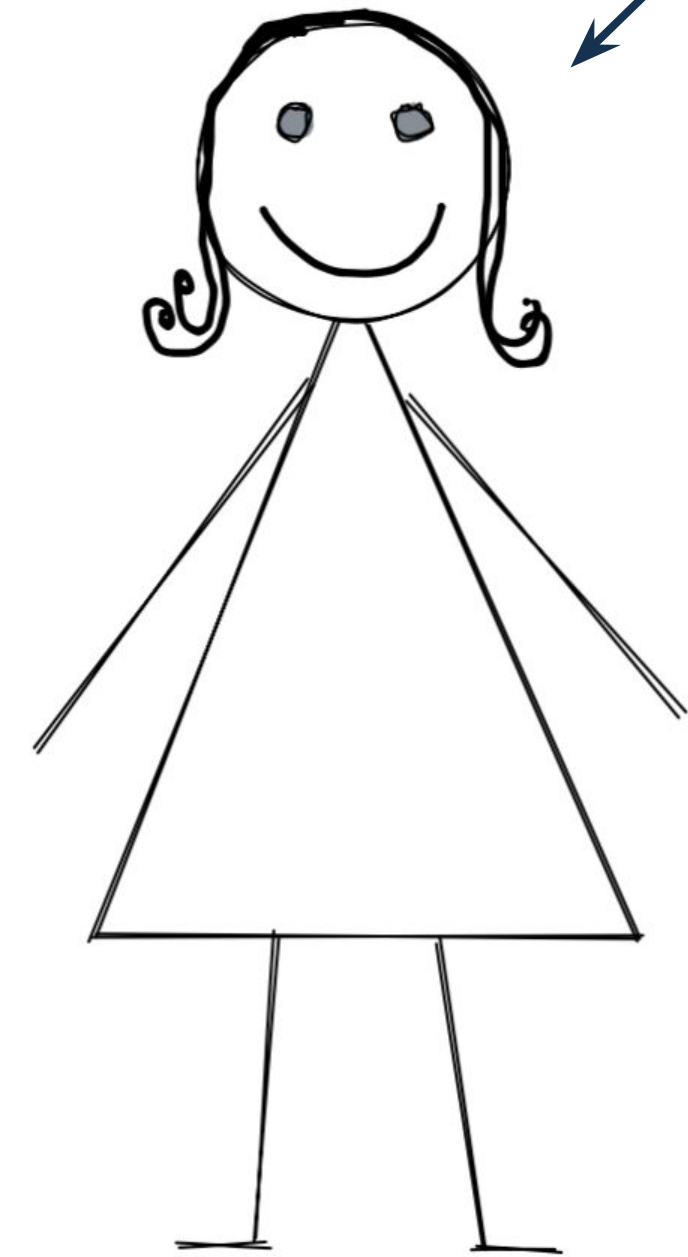
What happened before?

... there was a girl called Larissa.

her beloved aquarium



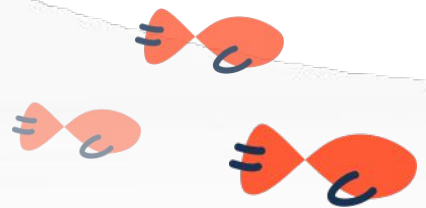
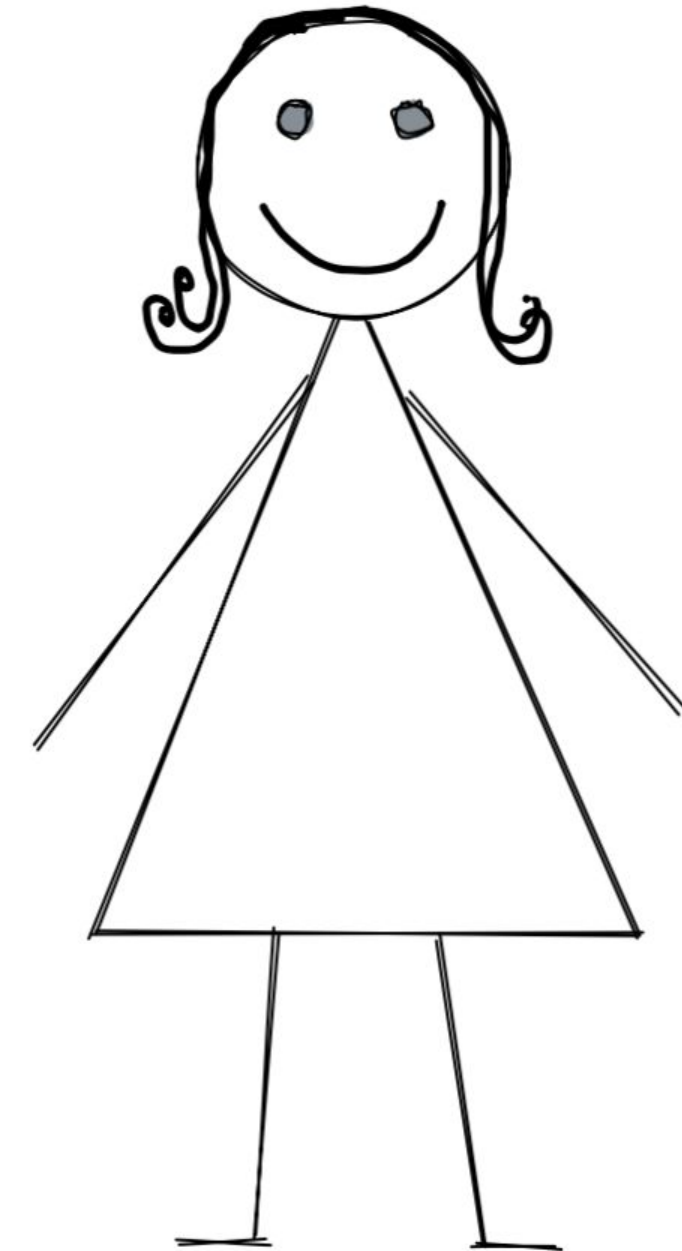
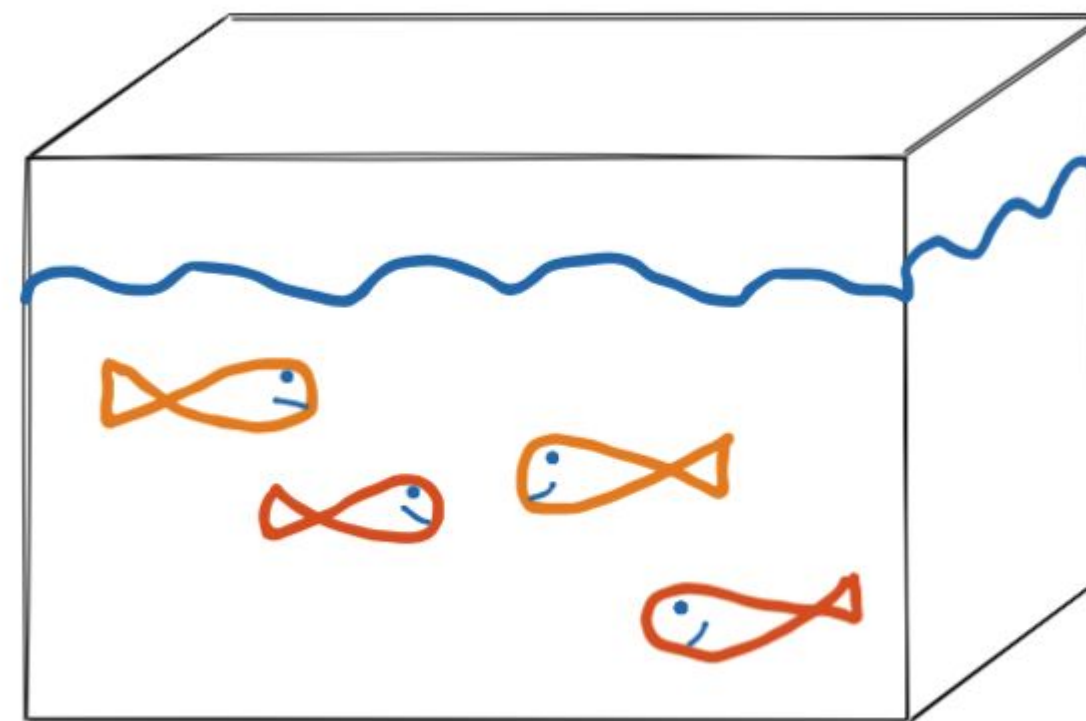
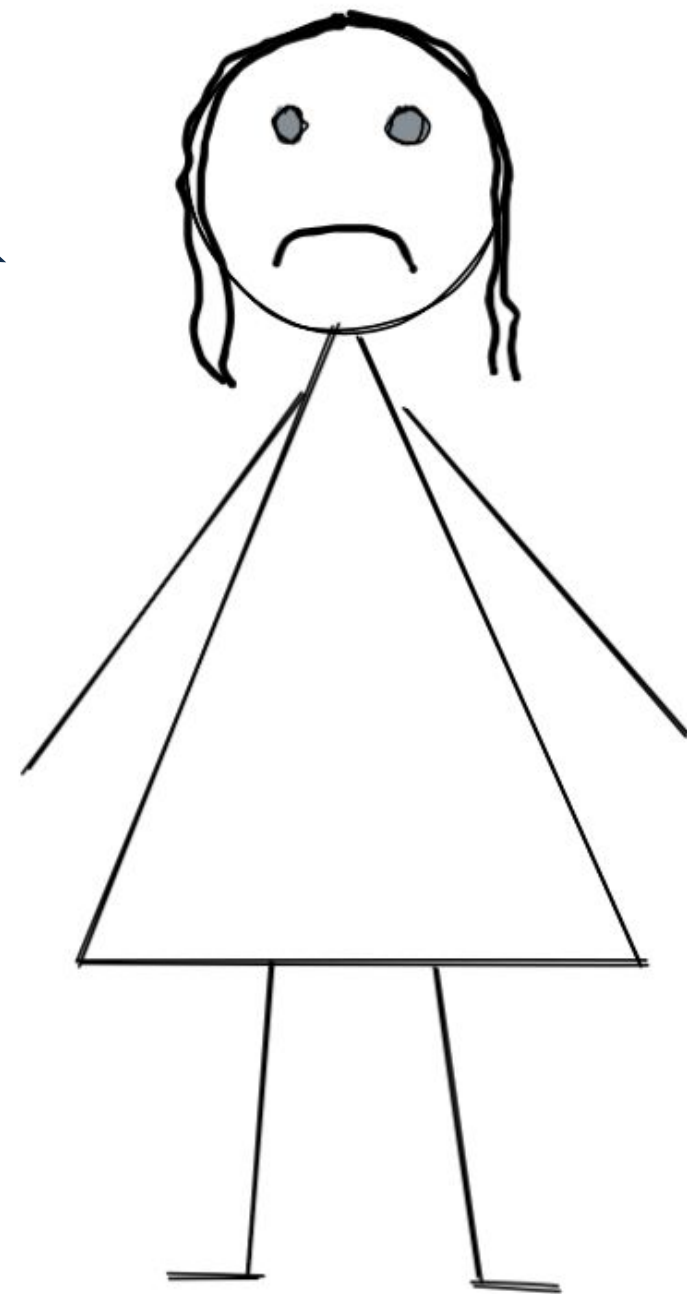
our colleague
Larissa



What happened before?

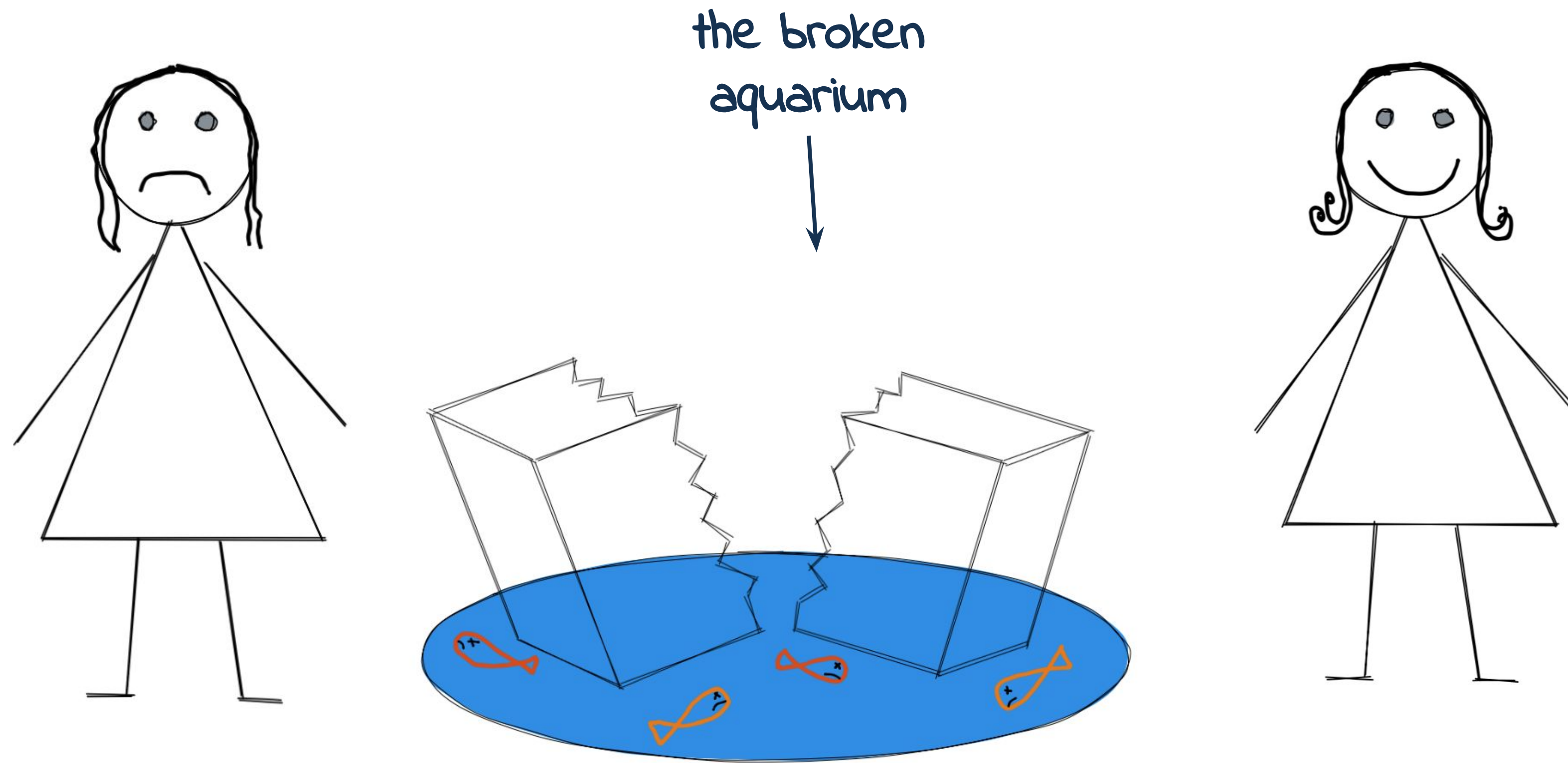
... there was a girl called Larissa.

her unhappy
sister



What happened before?

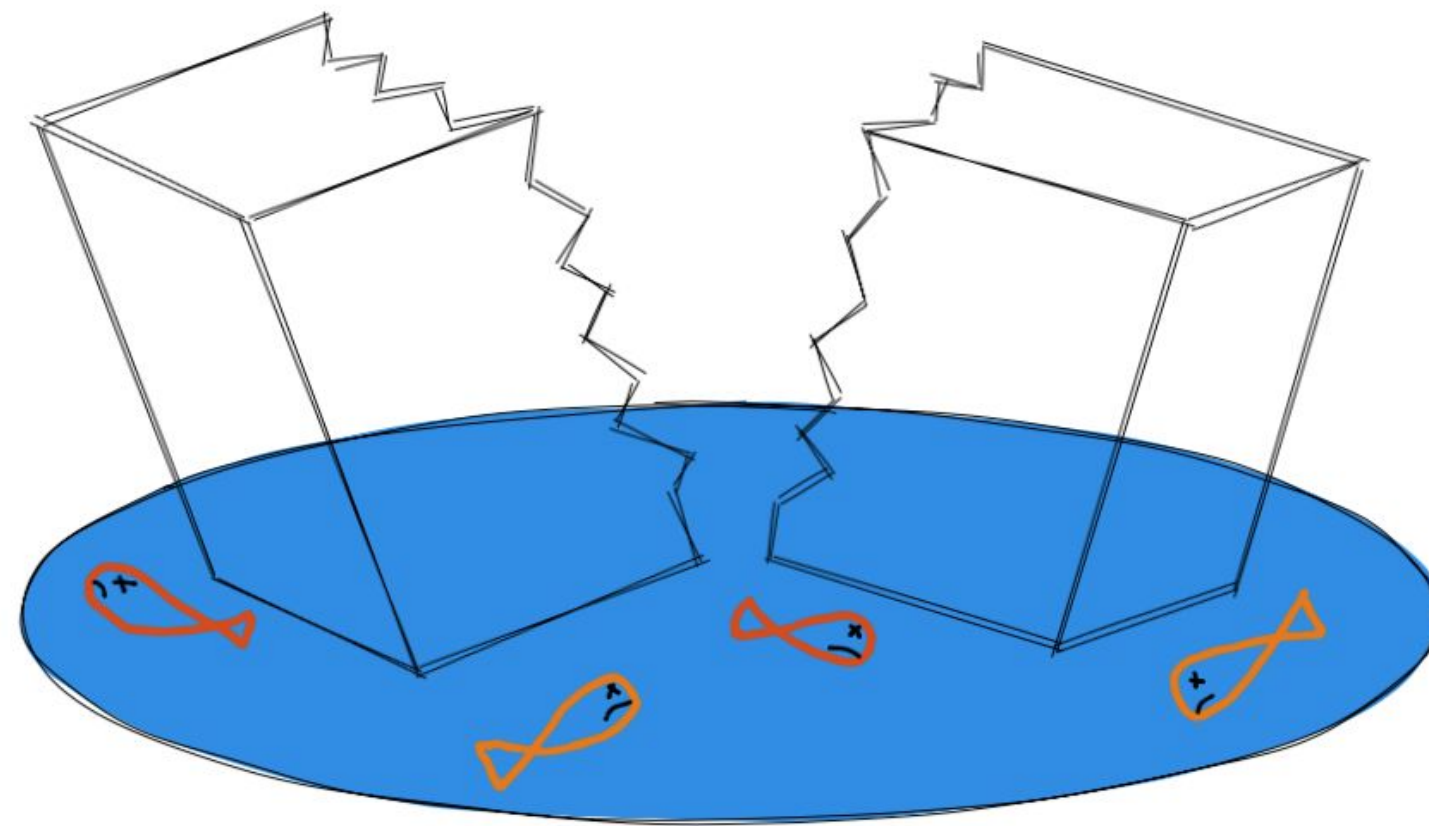
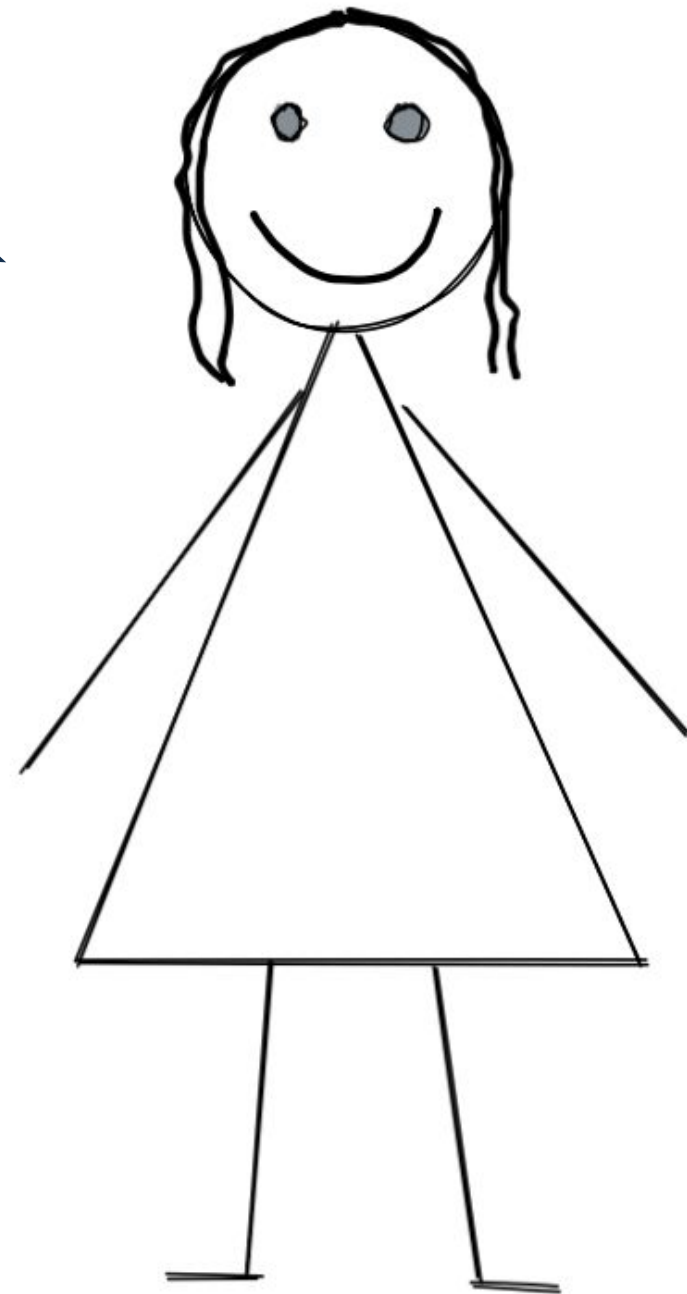
... there was a girl called Larissa.



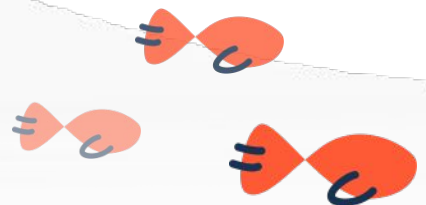
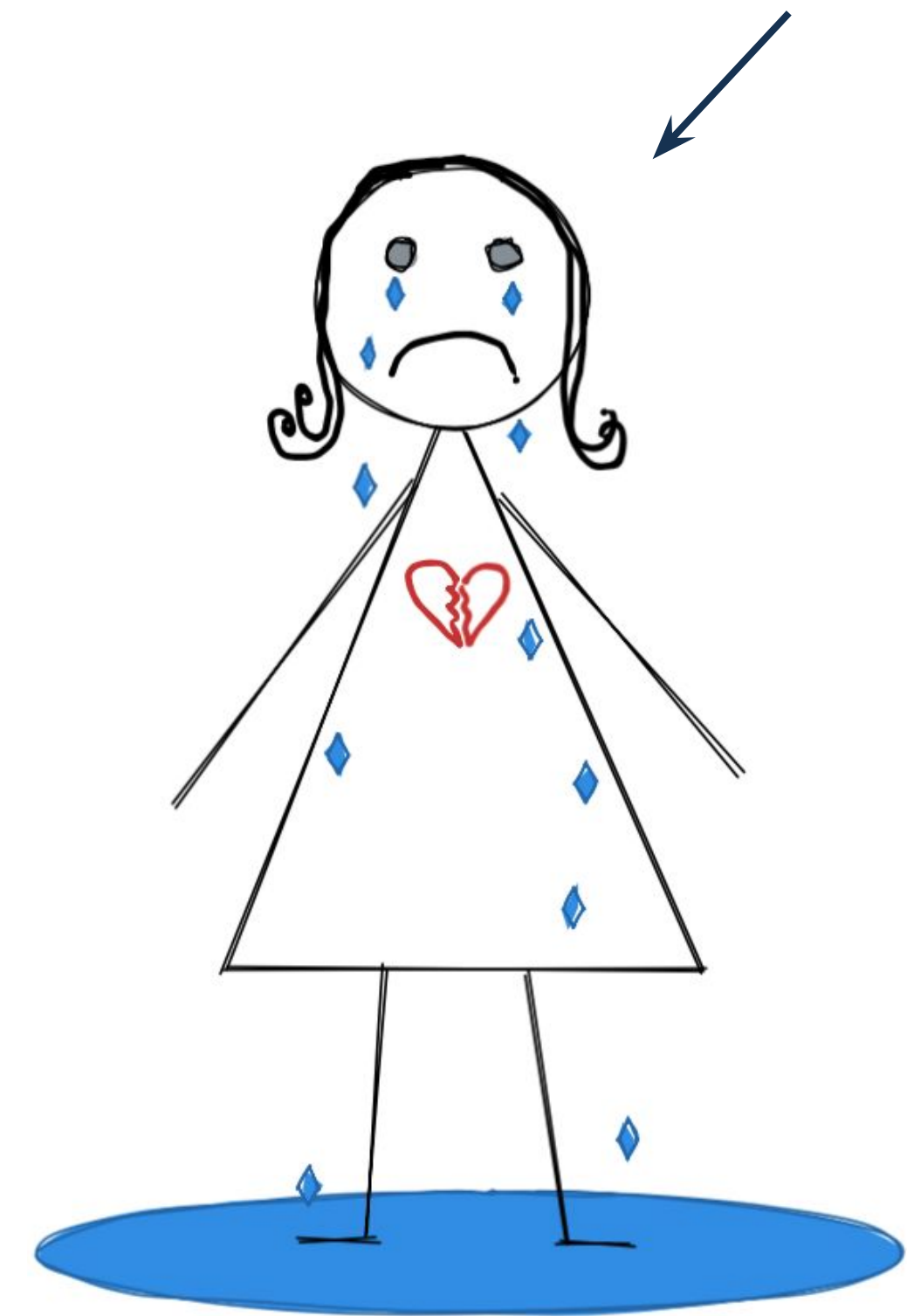
What happened before?

... there was a girl called Larissa.

her happy
sister

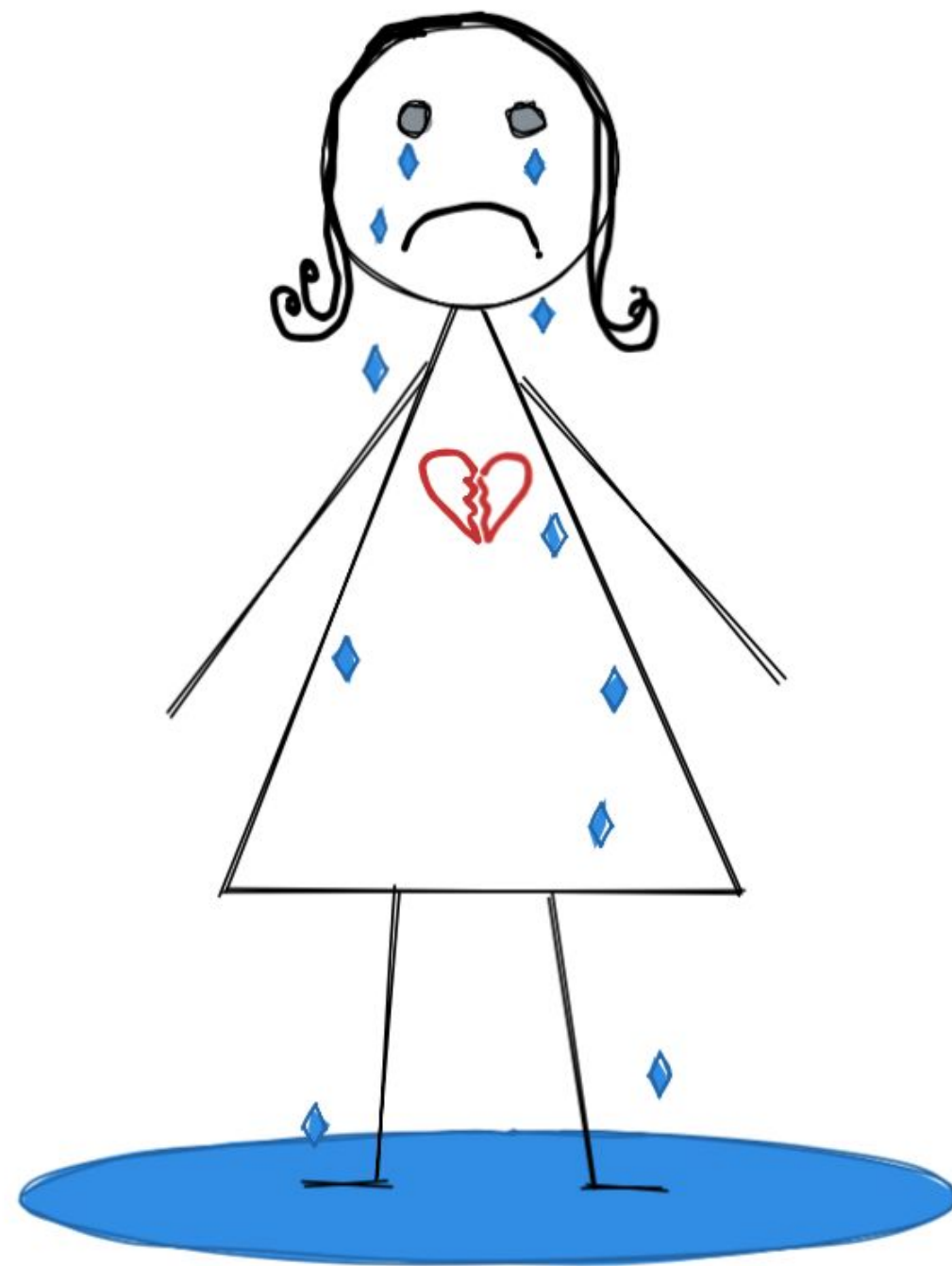


a heartbroken
Larissa

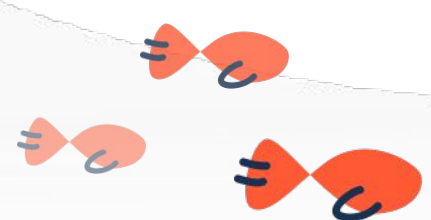
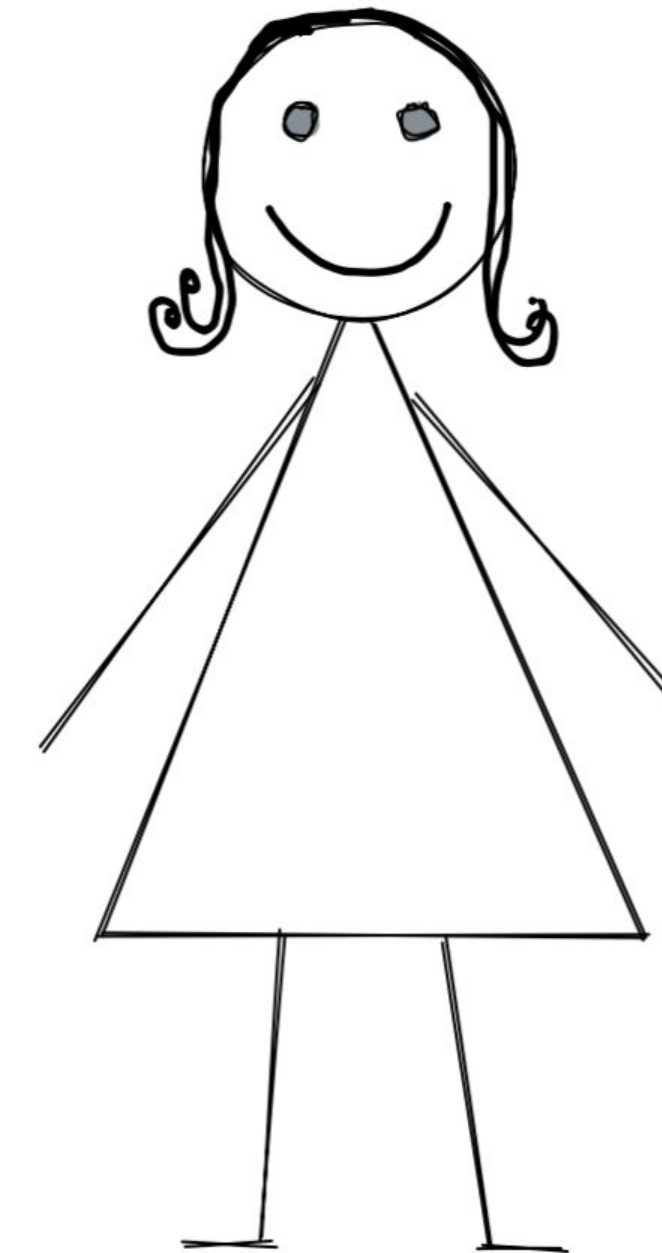


Our mission

How can we make Larissa happy again?



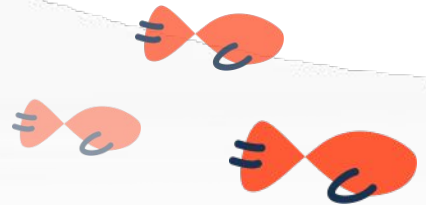
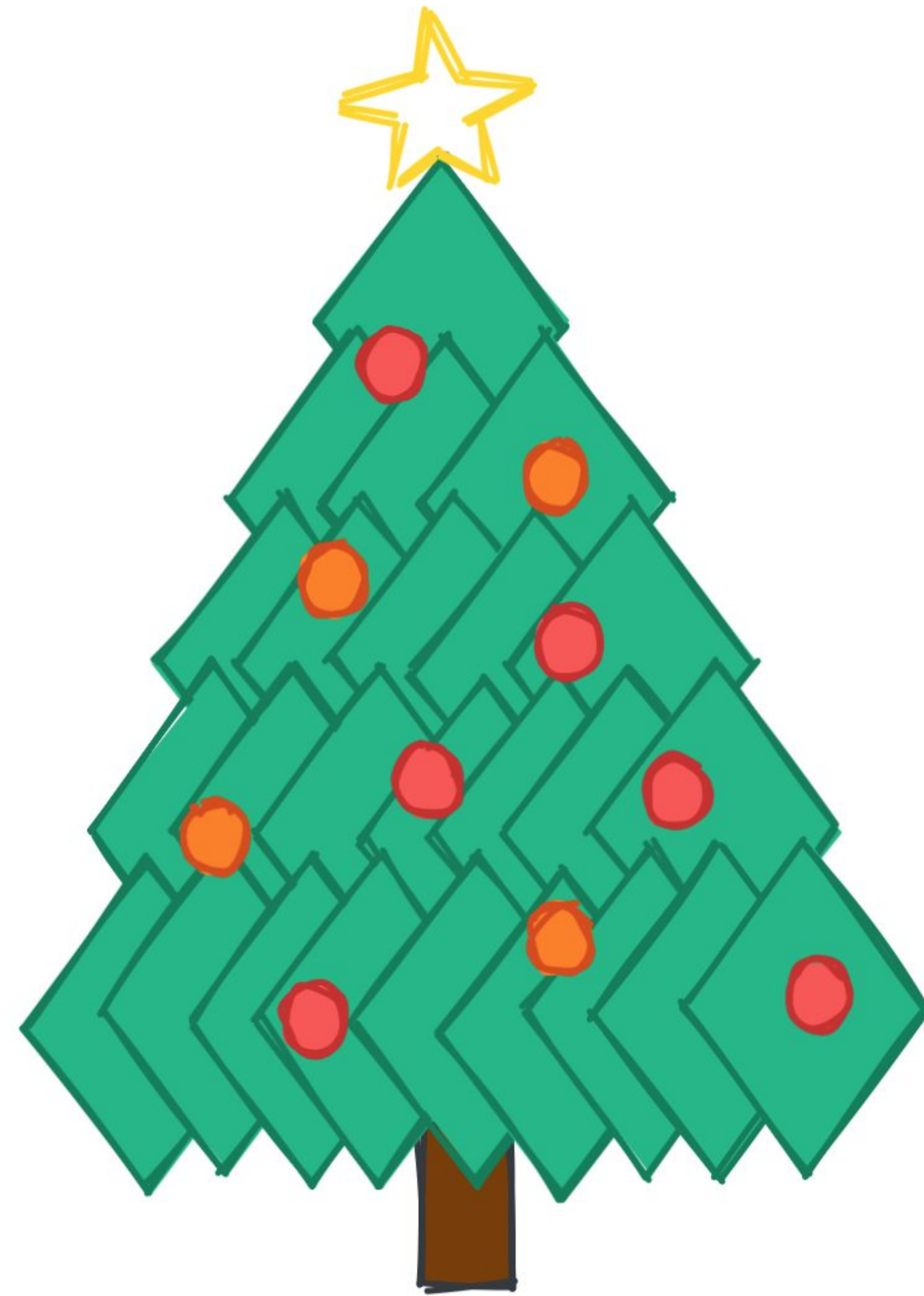
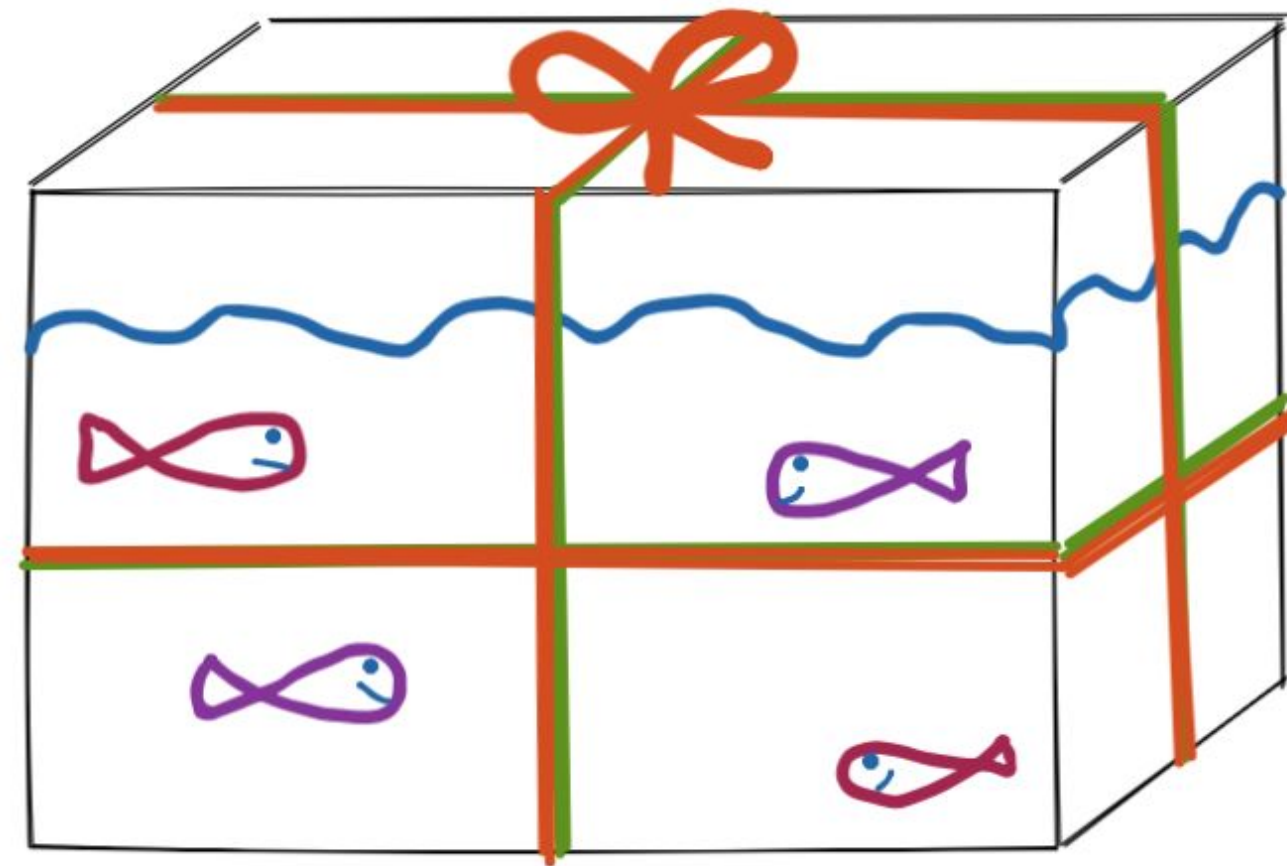
our mission →



Our mission...

Christmas is coming...

Larissa's
new aquarium



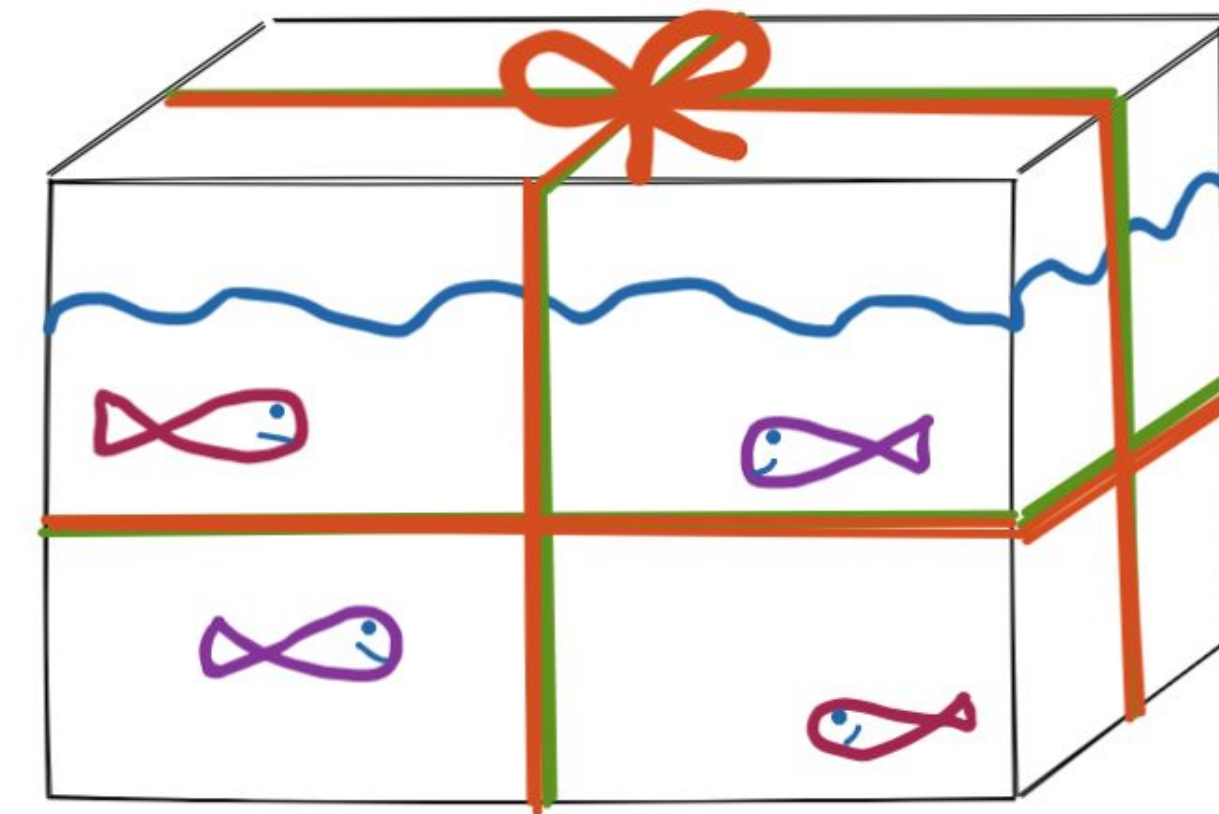
Our mission

What will we do today?

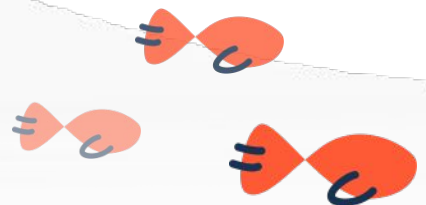
We need to find some new fish fast!

Of course we want an excellent selection...

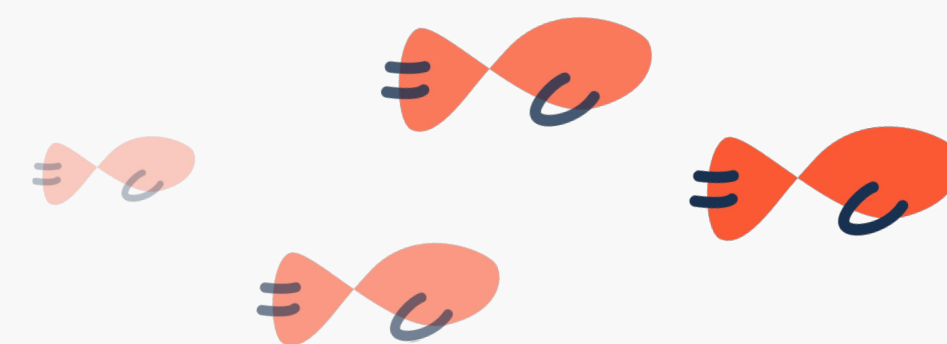
...and get some fish she has never had before.



Let's scrape the web with Beautiful Soup to find the perfect fish!



HTML



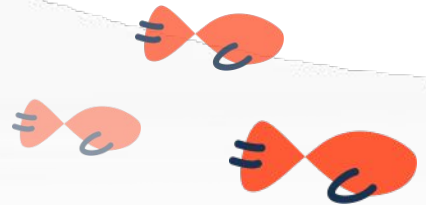
HTML

What is HTML?

- HyperText Markup Language
- standard markup language for Web pages
- describes structure of a Web page



But why do we need to know HTML to scrape the Web with BeautifulSoup?



HTML

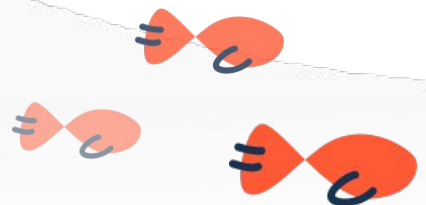
What is HTML?

- HyperText Markup Language
- standard markup language for Web pages
- describes structure of a Web page



But why do we need to know HTML to scrape the Web with BeautifulSoup?

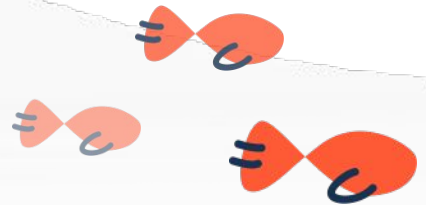
- BeautifulSoup let's us extract data/information out of an HTML file.
- To get the information we want we need to know which HTML element to access.



HTML

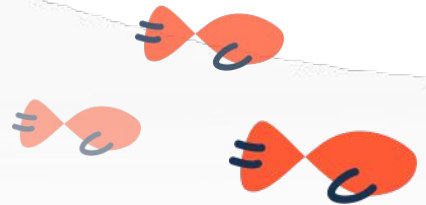
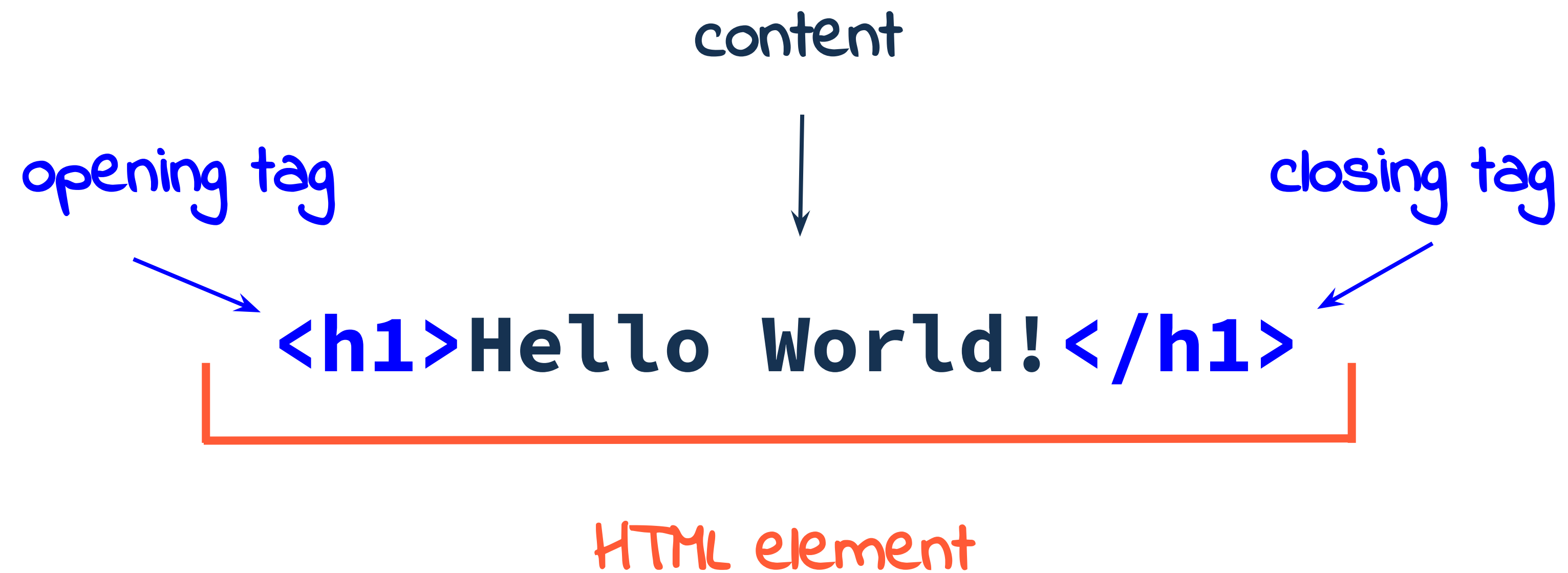
Syntax

```
<h1>Hello World! </h1>
```



HTML

Syntax



HTML

Example

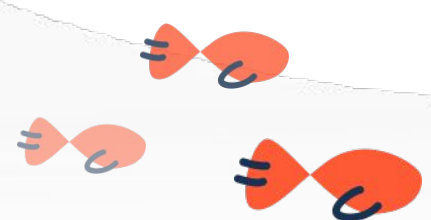
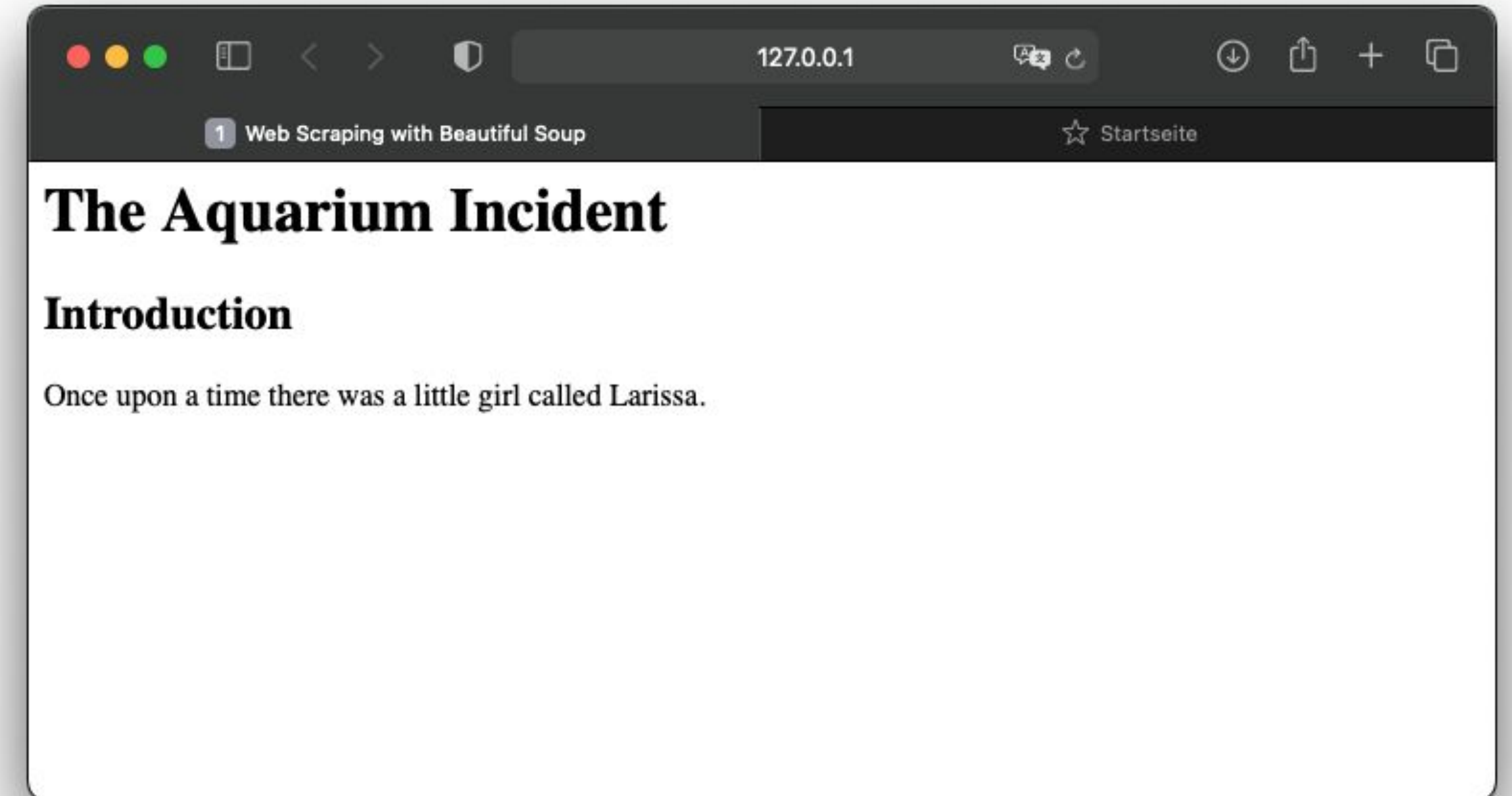


```
<html>

<head>
  <title>Web Scraping with Beautiful Soup</title>
</head>

<body>
  <h1>The Aquarium Incident</h1>
  <h2>Introduction</h2>
  <p>Once upon a time there was a little girl called
    Larissa.
  </p>
</body>

</html>
```



HTML

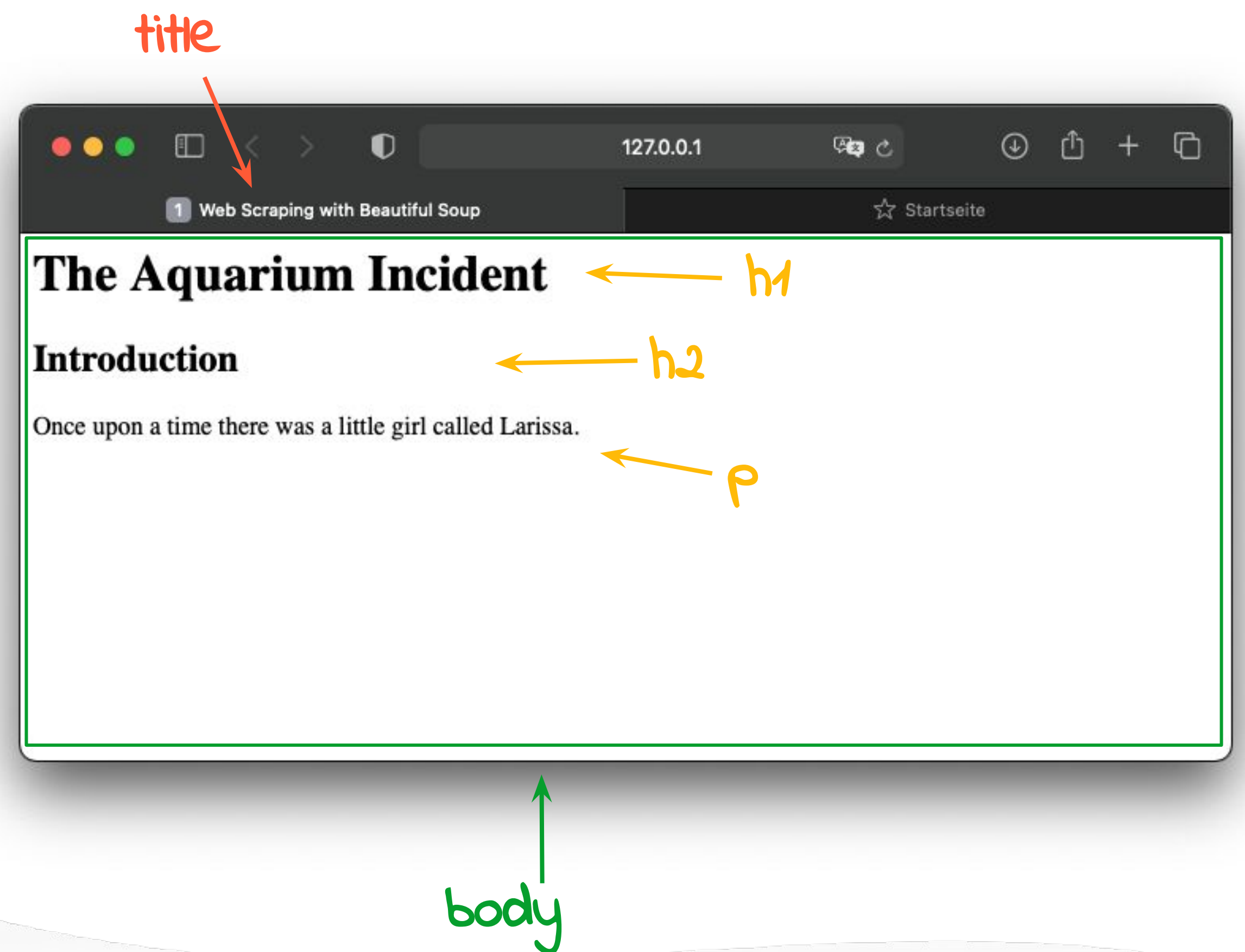
Example

```
<html>

<head>
  <title>Web Scraping with Beautiful Soup</title>
</head>

<body>
  <h1>The Aquarium Incident</h1>
  <h2>Introduction</h2>
  <p>Once upon a time there was a little girl called
    Larissa.
  </p>
</body>

</html>
```



Lists

ordered lists

```
<p>We need for a new  
aquarium:</p>  
<ol>  
  <li>an aquarium</li>  
  <li>some decoration</li>  
  <li>lots of fish</li>  
</ol>
```

list element

We need for a new aquarium:

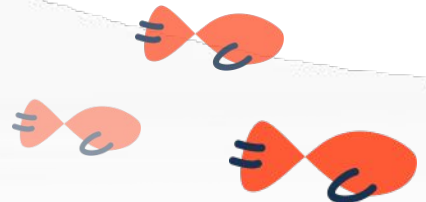
1. an aquarium
2. some decoration
3. lots of fish

unordered lists

```
<p>Possible new inhabitants:</p>  
<ul>  
  <li>piranha</li>  
  <li>hammerhead shark</li>  
  <li>manta ray</li>  
</ul>
```

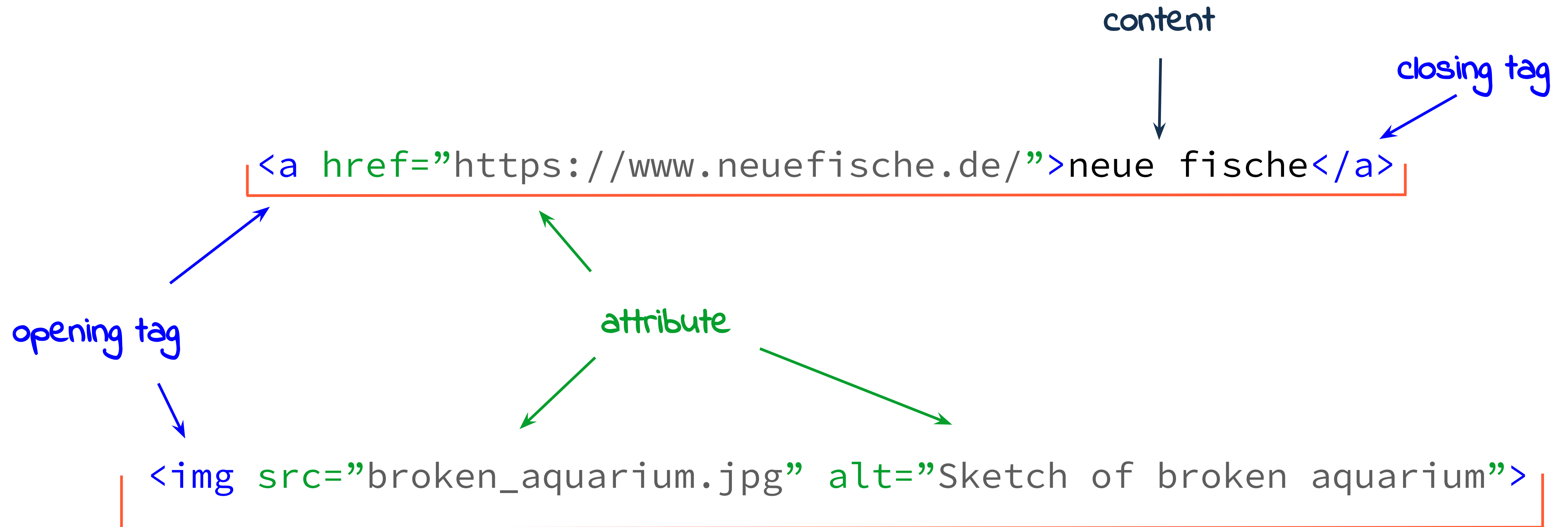
Possible new inhabitants:

- piranha
- hammerhead shark
- manta ray

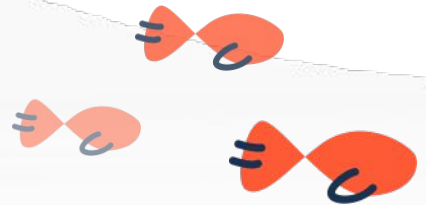


HTML

Links and Pictures



HTML element



HTML

Class and ID

```
<body>

  <h1>The Aquarium Incident</h1>

  <h2>Introduction</h2>

  <p>Once upon a time there was...</p>

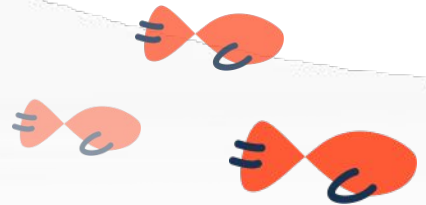
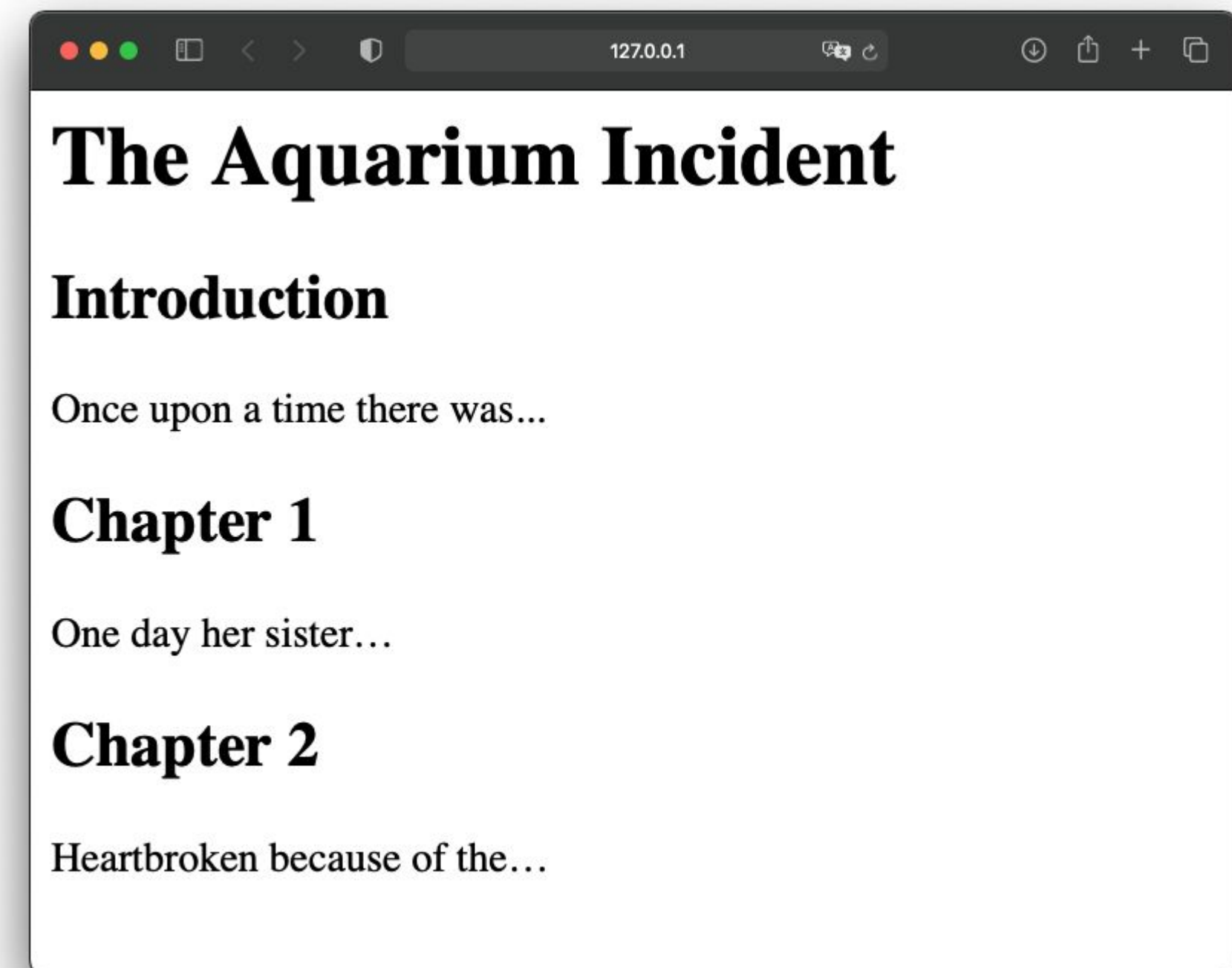
  <h2>Chapter 1</h2>

  <p>One day her sister...</p>

  <h2>Chapter 2</h2>

  <p>Heartbroken because of the...</p>

</body>
```



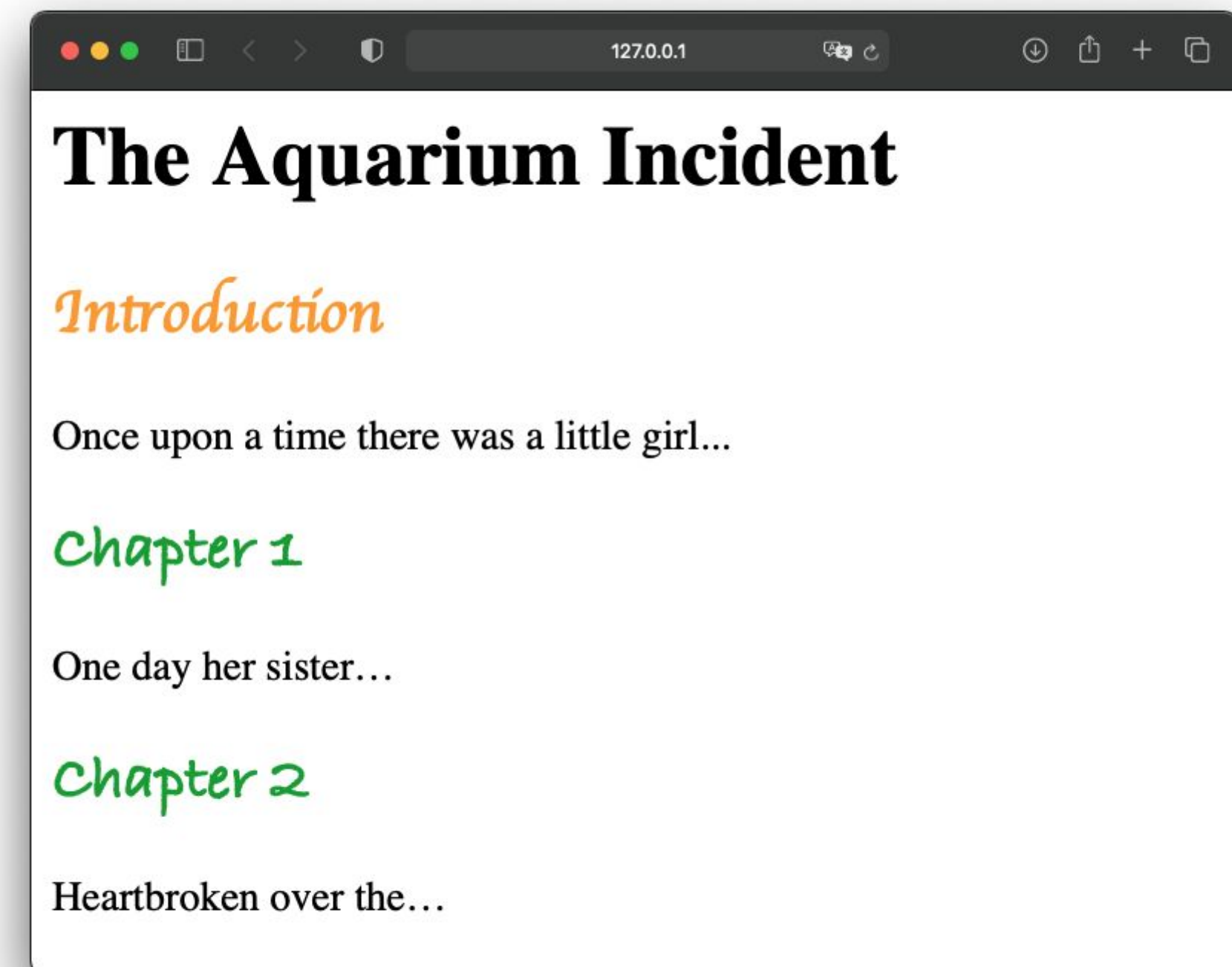
HTML

Class and ID

```
<body>
  <h1>The Aquarium Incident</h1>
  <h2 id="intro">Introduction</h2>
  <p>Once upon a time there was...</p>
  <h2 class="chapter">Chapter 1</h2>
  <p>One day her sister...</p>
  <h2 class="chapter">Chapter 2</h2>
  <p>Heartbroken because of the...</p>
</body>
```

id attribute

class attribute



HTML

Class and ID

```
<body>
  <h1>The Aquarium Incident</h1>
  <h2 id="intro">Introduction</h2>
  <p>Once upon a time there was...</p>
  <h2 class="chapter">Chapter 1</h2>
  <p>One day her sister...</p>
  <h2 class="chapter">Chapter 2</h2>
  <p>Heartbroken because of the...</p>
</body>
```

id attribute → `id="intro"`

class attribute → `class="chapter"`

id

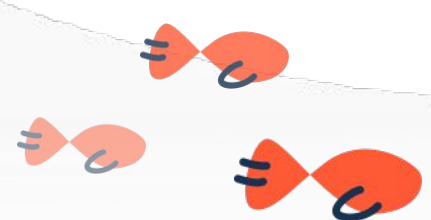
- unique identifier for one specific HTML element

class

- used to specify a class for an HTML element
- multiple HTML elements can share same class

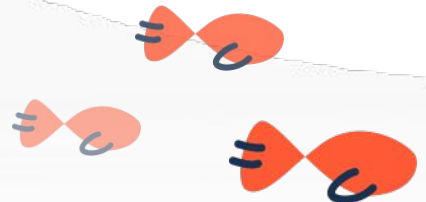
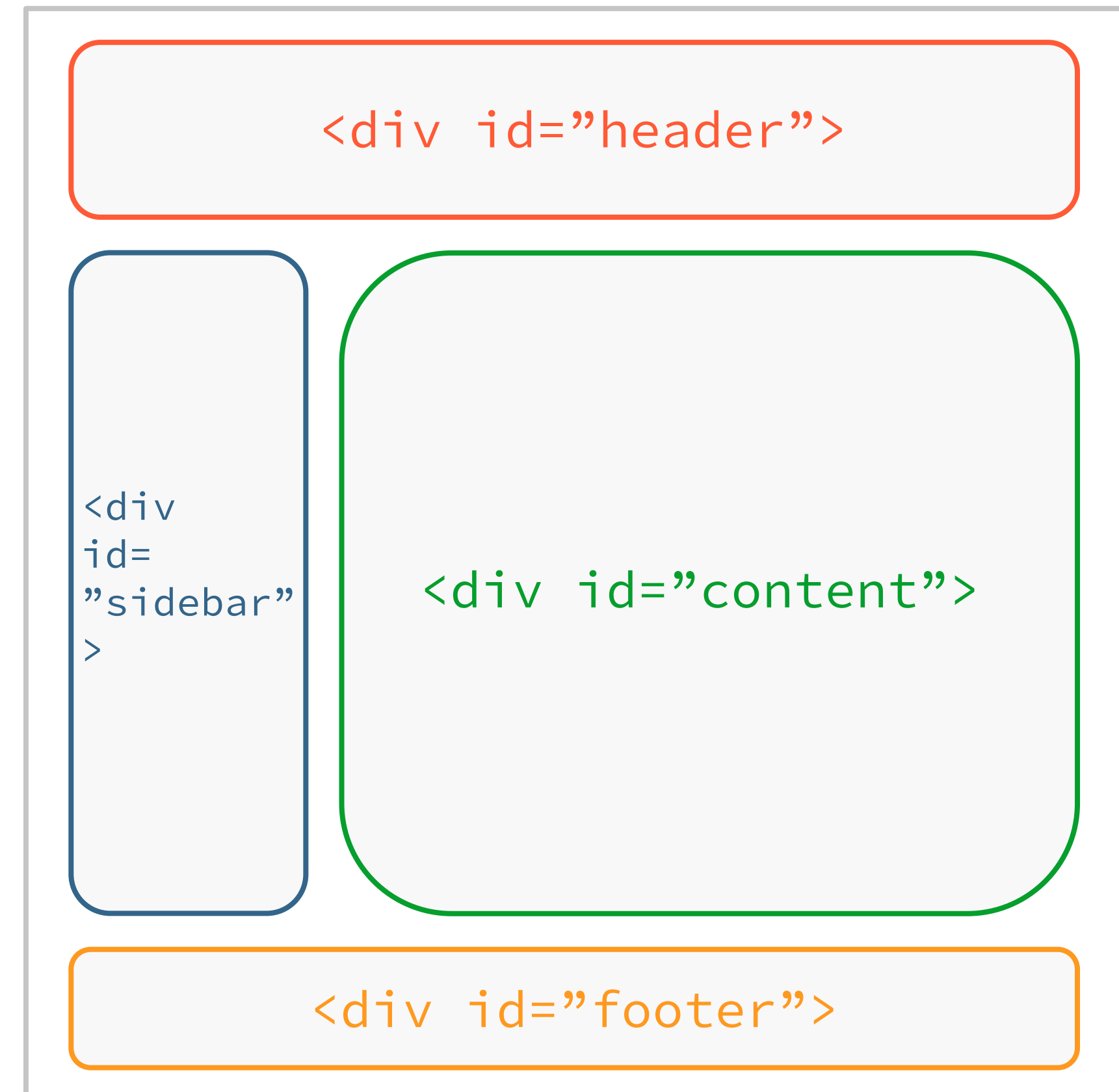
→ often used to style specific HTML elements

→ can be used to access information we want to scrape



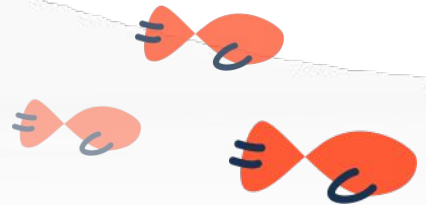
Blocks / Containers

- `<div>...</div>` tag
- divides the content of a web page
- groups other HTML elements into sections which can be styled independently

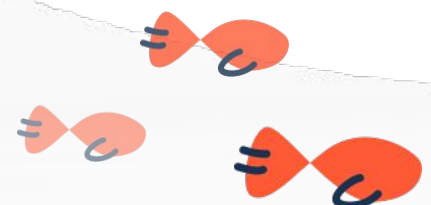


Time to checkout the first notebook!

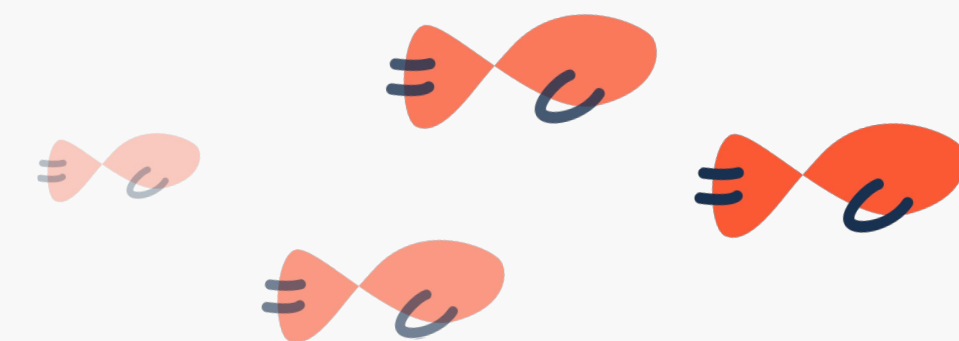
- We will open breakout rooms.
- The notebook can be opened in Google Colab by clicking on the link in the Meetup repository.
- We will meet again in 15 min in the main room.



Time for a short break...



BeautifulSoup



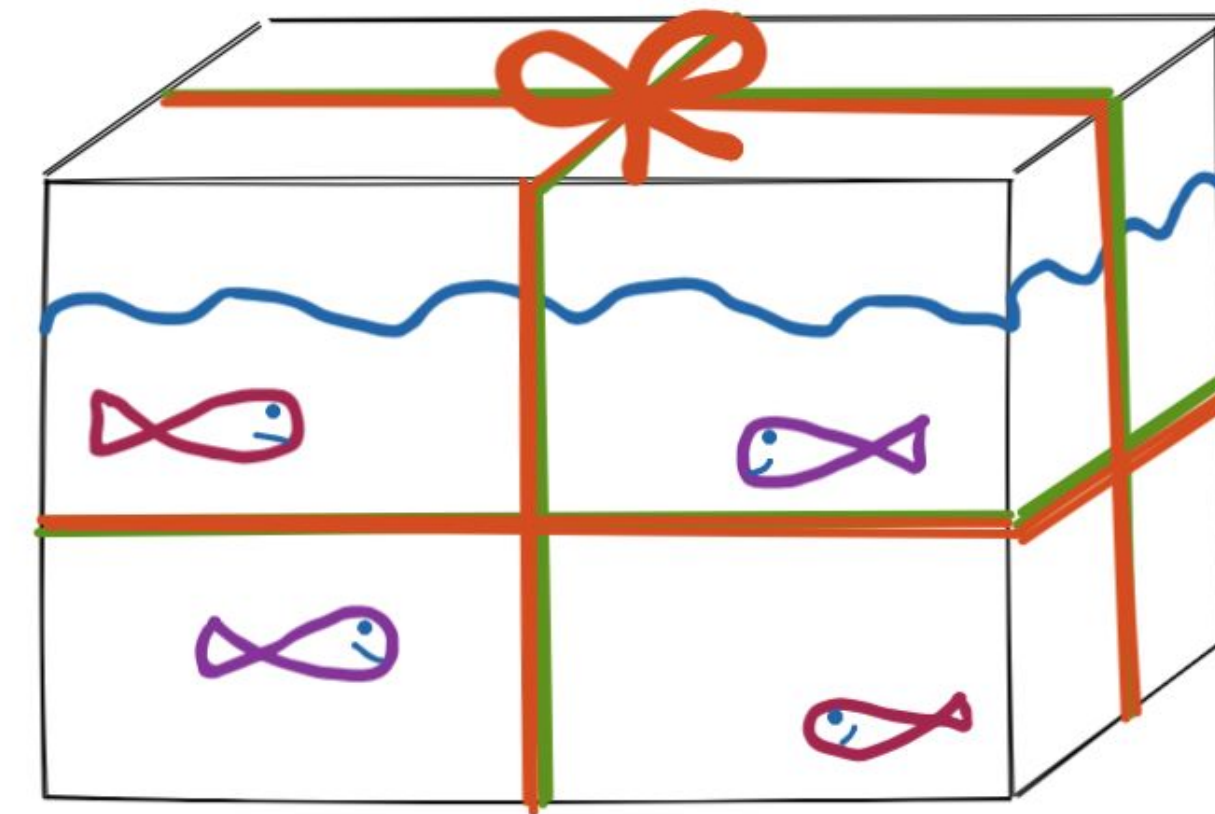
Our mission

A little reminder

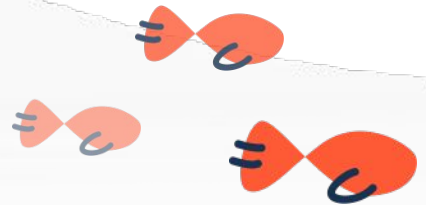
We need to find some new fish fast!

Of course we want an excellent selection...

...and get some fish she has never had before.



Let's scrape the web with BeautifulSoup to find the perfect fish!

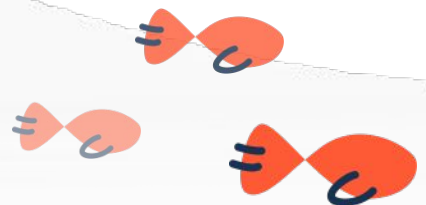


What is BeautifulSoup?

- Python Library to get data out of HTML and XML files
- It uses parsers to provide ways of navigating, searching, and modifying the parse tree.
- Quite easy to learn



But how do we start?

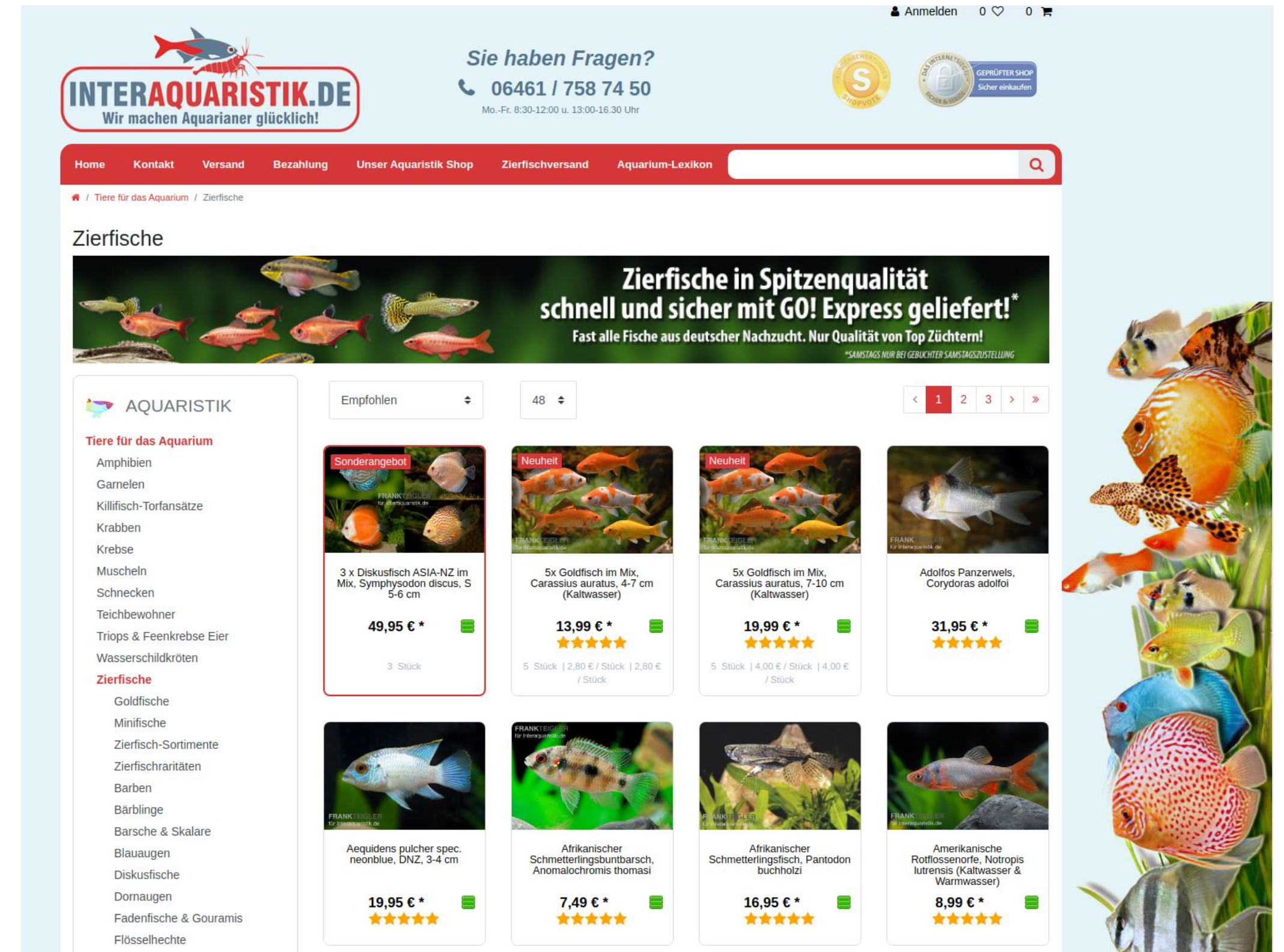


BeautifulSoup

Let's find a website that sells fish!

- <https://www.interaquaristik.de/tiere/zierfische>
- It is in German but you will see it sells fish ;)
- So there are fish with prices and “Neuheit (new item)” and “Sonderangebot (special offer)” marks

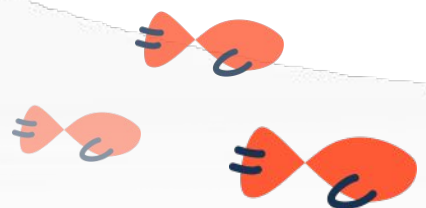
But wait are we allowed to scrape this site?



Is web scraping legal?

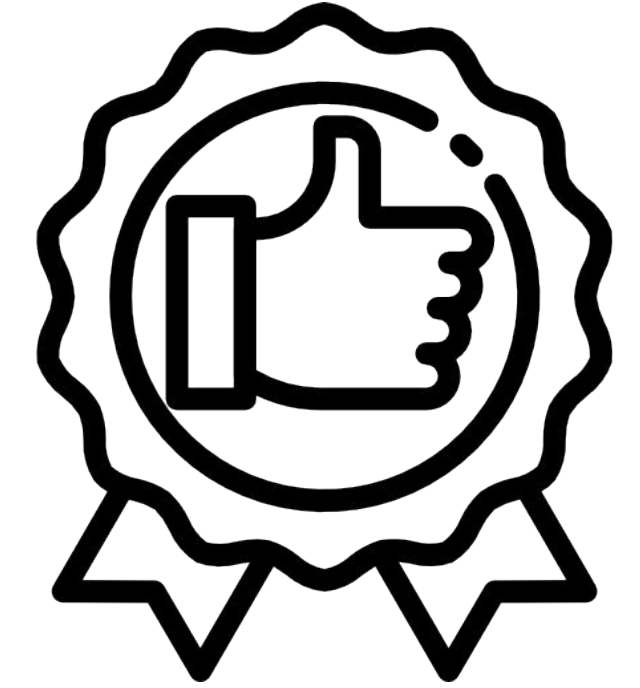
How do we find out what is allowed and what not?

- Unfortunately, there's not a cut-and-dry answer if web scraping is per se legal
- Some websites explicitly allow web scraping. Others explicitly forbid it. So have a look into the Terms of Service!
- If there are rules, we should follow them. If there are not, then it becomes more of a judgement call.
- Remember, though, that web scraping consumes server resources for the host website.



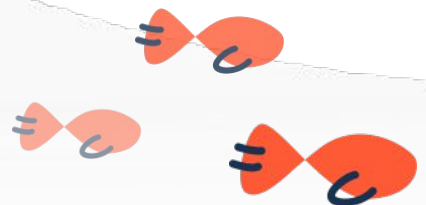
Is web scraping legal?

Web Scraping Best Practice



- Never scrape more frequently than you need to
- Consider caching the content you scrape so that it's only downloaded once
- Build pauses into your code using functions like `time.sleep()` to keep from overwhelming servers with too many requests too quickly.

How do we find out if there are rules on that website aside from the ToS?



Is web scraping legal?

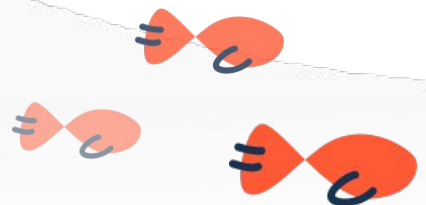
robots.txt

- Nearly every website has a robots.txt

“The robots exclusion standard, also known as the robots exclusion protocol or simply robots.txt, is a standard used by websites to communicate with web crawlers and other web robots. The standard specifies how to inform the web robot about which areas of the website should not be processed or scanned.”

- So the robots.txt tells us what is allowed and what not
- You can access it by adding /robots.txt to the main page address

But what does it really tell us?



Is web scraping legal?

robots.txt

```
User-agent: *  
Disallow:
```

For whom does the rule apply? * means every agent

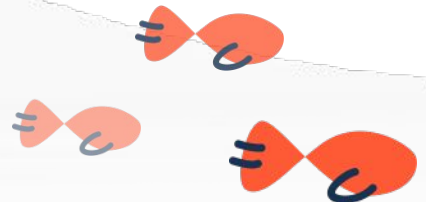
What directive are we allowed to scrape?
(in this case everything)

```
User-agent: *  
Disallow:/
```

Again every agent

But this time we are not allowed to scrape anything!

But you can also specify directories or files that are not allowed to be scraped!



Is web scraping legal?

Let's have a look on our target's robots.txt

<https://www.interaquaristik.de/robots.txt>

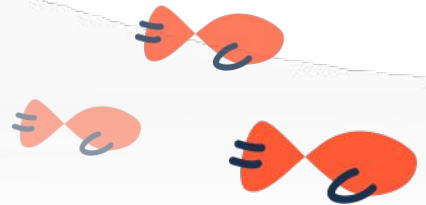
```
User-agent: *  
Disallow: /plenty/  
Allow: /plenty/api/external.php  
User-agent: ia_archiver  
Disallow: /
```

All user agents are not allowed to access the directory: plenty

But are allowed to access the file: /plenty/api/external.php

The user agent "ia_archiver" is not allowed to access anything

So we are allowed to scrape the website as long as we stay away from /plenty/ !



Web scraping

Now we can start with the scraping!

First we have to get the HTML:

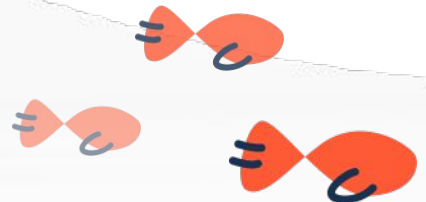
- We will use the python library `requests`

```
import requests  
page = requests.get("<link to website>")  
html = page.content
```

We save the HTML
content in a
variable



We send a request to the
website and saving the
response




Now we can start with the scraping!

We can use BeautifulSoup to parse this document and extract information from it:

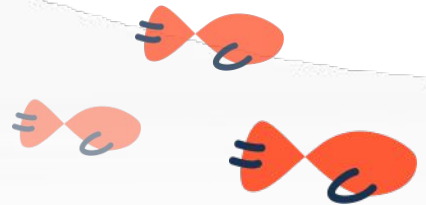
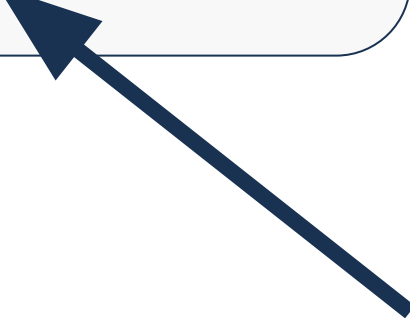
- We will create a BeautifulSoup instance bs

```
from bs4 import BeautifulSoup  
bs = BeautifulSoup(html, 'html.parser')
```

We pass the variable from the slide before to bs



We are defining the parser we want to use (in this case Python's built in html parser)



Now we can start with the scraping! But can we?

With this BeautifulSoup instance we can work and search for tags or attributes:

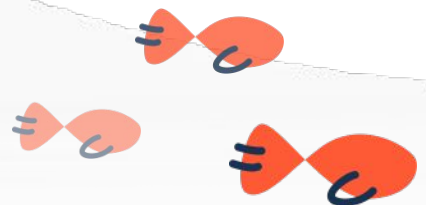
- What tags are we looking for?

```
print(bs.prettify())
```

← This will give back a
“prettified” version of
the HTML

- But in my opinion it is too incomprehensible for a good overview....

How else to get the tags or attributes we are looking for?

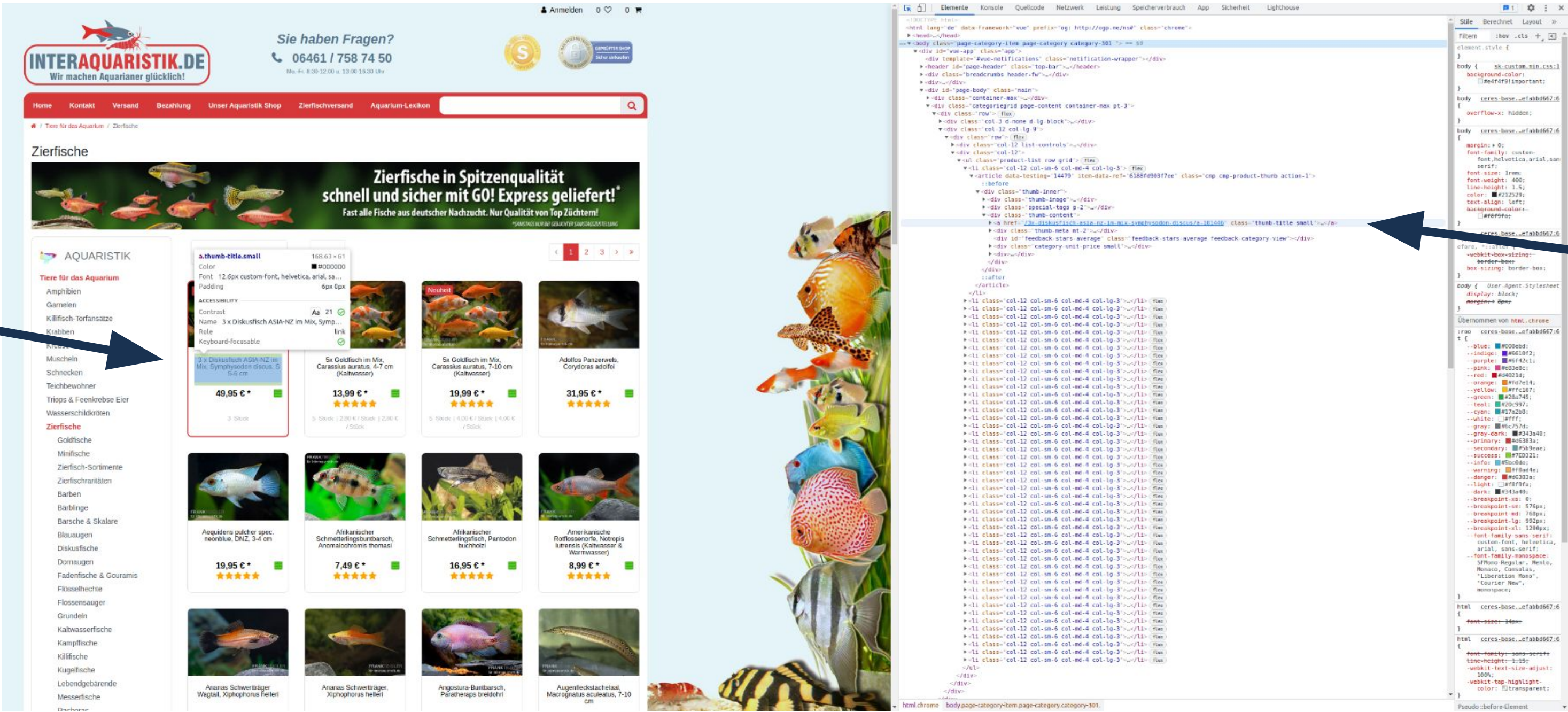
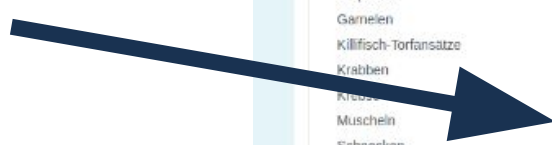


DevTools of your browser

Using the DevTools of your browser

You can access it by pressing ctrl + shift + c or command + shift + c for mac:

If you hover over items in your browser window



It will show you the corresponding line in the DevTool

Finally we can start with the scraping!

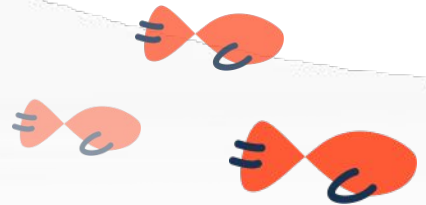
Let's start simple:

- What is the title of the website?

bs.title

<title>Zierfische kaufen, bestellen Sie Fische im Zierfisch-Versand Shop</title>

That was easy! But let's continue with solving our problem and find some fish!



Finally we can start with the scraping!

So let's start with the name of the fish:

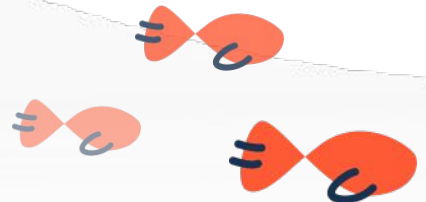
- In the DevTools we could identify that all the tag elements that include the fish names have the class: 'thumb-title small'

with .find or .find_all
we can search for
tags or attributes

```
bs.find(class_ = 'thumb-title small')
```

here we can define what
we are looking for

```
<a class="thumb-title small" href="/3x-diskusfisch-asia-nz-im-mix-symphysodon-discus/a-101446">  
    3 x Diskusfisch ASIA-NZ im Mix, Symphysodon discus, S 5-6 cm  
</a>
```



Finally we can start with the scraping!

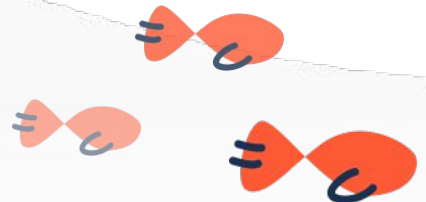
But we want all names so we will use `.find_all`:

- That will return all lines with that class including the tags but we are only interested in the text:

```
descriptions = bs.find_all(class_ = 'thumb-title small')  
descriptions_lst = [description.get_text() for description in descriptions]
```

We are saving the results in a list using list comprehension

With `.get_text()` we get back only the text that is between the tags



Finally we can start with the scraping!

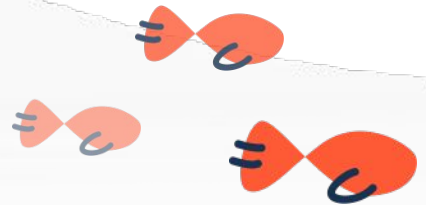
But the strings we get doesn't look nice:

- Because it will also extract spaces we have to do a little bit of cleaning:

```
descriptions_lst = [description.strip() for description in descriptions_lst]
```

We are saving the results in a list using list comprehension

With .strip() we are getting rid of spaces and paragraphs



Finally we can start with the scraping!

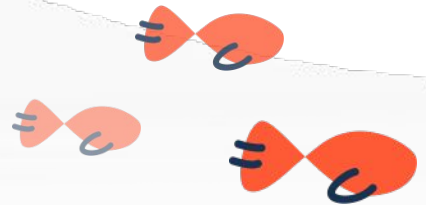
We will continue with the price:

- The prices are all in the class “price”:

```
prices = bs.find_all(class_ = 'price')  
prices_lst = [price.get_text() for price in prices]  
prices_lst = [price.strip() for price in prices_lst]
```

- Now we get something like this: '49,95\xa0€ *'

Wouldn't it be nice if we had floats and not strings for the price?



Finally we can start with the scraping!

We will continue with the price:

- So let's remove the letters:

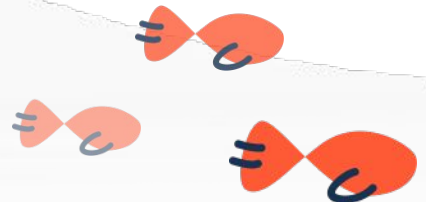
```
prices_lst = [price.replace('\xa0€ *', '') for price in prices_lst]
```

- To convert the numbers to floats we need points instead of commata:

```
prices_lst = [price.replace(',', '.') for price in prices_lst]
```

- And finally convert them:

```
prices_lst = [float(price) for price in prices_lst]
```

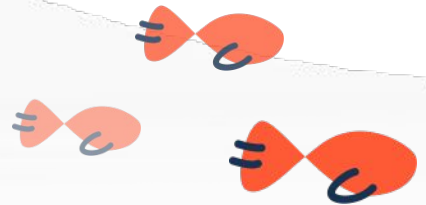


Finally we can start with the scraping!

And finally we will do the same for the specials:

- **Specials:**

```
<div class="special-tags p-2">  
  <span class="badge badge-offer badge-danger">  
    Sonderangebot  
  </span>  
</div>
```

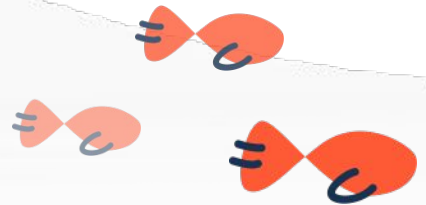


Finally we can start with the scraping!

And finally we will do the same for the specials:

- Specials:

```
specials = bs.find_all('div', {'class': 'special-tags p-2'})
special_lst = []
for special in specials:
    special_text = special.find("span").get_text().strip()
    special_lst.append(special_text)
```



Web scraping

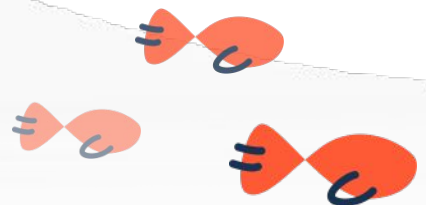
Finally we can start with the scraping!

Maybe you have recognized that we only scrape the first of 29 pages of fish... and i don't want to copy + paste 28 links into a list... So what now?

Automation!

- The starting link looks like this: <https://www.interaquaristik.de/tiere/zierfische>
- And the link to the second page: <https://www.interaquaristik.de/tiere/zierfische?page=2>
- The only thing that changed was the last part: [?page=2](#)

Let's use this!



Finally we can start with the scraping!

We will write a for-loop that iterates over all sites:

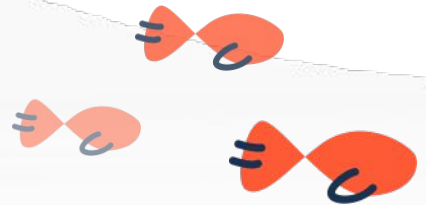
```
link = 'https://www.interaquaristik.de/tiere/zierfische'
for _ in range(30):
    time.sleep(3)
    if _ == 0:
        page = requests.get(link)
        html = page.content
    else:
        page = requests.get(link + f'?page={_}')
        html = page.content
```

← 29 iterations

← adding a sleep timer to reduce the traffic

← if-statement for the main page

← else-statement for the following pages using a f-string



Web scraping

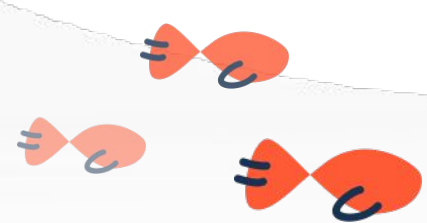
We did it!

df

	fish_names	fish_prices in EUR	special
0	3 x Diskusfisch ASIA-NZ im Mix, Symphysodon di...	49.95	Sonderangebot
1	5x Goldfisch im Mix, Carassius auratus, 4-7 cm...	13.99	Neuheit
2	5x Goldfisch im Mix, Carassius auratus, 7-10 c...	19.99	Neuheit
3	Adolfos Panzerwels, Corydoras adolfoi	31.95	NaN
4	Aequidens pulcher spec. neonblue, DNZ, 3-4 cm	19.95	NaN
...
1379	Zierfisch-Sortiment Amazonas für 100 cm	109.99	Neuheit
1380	Zierfisch-Sortiment Asien für 120 cm	99.99	Neuheit
1381	Zierfisch-Sortiment Einsteiger für 60 cm	69.99	Neuheit
1382	Zierfisch-Sortiment Minifische für Becken ab 6...	79.99	Neuheit
1383	Zierfisch-Sortiment Salmler rot/blau ab 100 cm	84.99	Neuheit

1384 rows x 3 columns

There are 1384 fish in our dataframe!



Web scraping

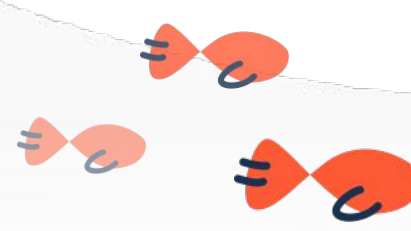
We did it!

```
df_special = df.query('special == "Neuheit"')
```

	fish_names	fish_prices in EUR	special
1	5x Goldfisch im Mix, Carassius auratus, 4-7 cm...	13.99	Neuheit
2	5x Goldfisch im Mix, Carassius auratus, 7-10 c...	19.99	Neuheit
13	Aulonocara hueseri "Likoma", DNZ	29.95	Neuheit
16	Aulonocara spec. Dragon Blood albino, DNZ	24.95	Neuheit
18	Aulonocara spec. im MIX, Männchen, DNZ	21.95	Neuheit
...
1379	Zierfisch-Sortiment Amazonas für 100 cm	109.99	Neuheit
1380	Zierfisch-Sortiment Asien für 120 cm	99.99	Neuheit
1381	Zierfisch-Sortiment Einsteiger für 60 cm	69.99	Neuheit
1382	Zierfisch-Sortiment Minifische für Becken ab 6...	79.99	Neuheit
1383	Zierfisch-Sortiment Salmler rot/blau ab 100 cm	84.99	Neuheit

250 rows x 3 columns

There are 250 new items let's filter the too expensive fish out!



Web scraping

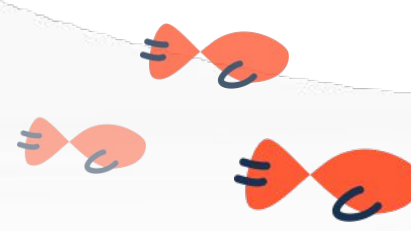
We did it!

```
df_final = df_special_offer[df_special_offer['fish_prices in EUR'] <= 25]
```

	fish_names	fish_prices in EUR	special
1	5x Goldfisch im Mix, Carassius auratus, 4-7 cm...	13.99	Neuheit
2	5x Goldfisch im Mix, Carassius auratus, 7-10 c...	19.99	Neuheit
16	Aulonocara spec. Dragon Blood albino, DNZ	24.95	Neuheit
18	Aulonocara spec. im MIX, Männchen, DNZ	21.95	Neuheit
26	Axelrods Zwergmaulbrüter, Cynotilapia axelrodi	19.95	Neuheit
...
1306	Sumatra Grundel, Mugilogobius sp Sumatra, 5er ...	19.95	Neuheit
1318	Tigerbarsch, Datnioides microlepis	18.99	Neuheit
1322	Towuti-Gelbflossengrundel, Glossogobius flavip...	12.99	Neuheit
1340	Vierstreifen-Lamprologus, Lepidiolamprologus n...	19.95	Neuheit
1368	White Peacock Endler Guppy, Poecilia wingei "W...	5.99	Neuheit

123 rows × 3 columns

Now there are only 123 fish below 25 € left!



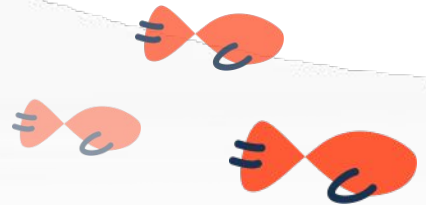
Web scraping

We did it!

So let's write some code to choose the fish:

```
BUDGET = 250
shopping_bag = []
price = 0
while price <= BUDGET:
    df_temp = df_final.sample(1)
    name = df_temp['fish_names'].values
    fish_price = df_temp['fish_prices in EUR'].values
    price += fish_price
    shopping_bag.append(name[0])
```

This gave me back a list with 19 fish for a price of 258.91 €.

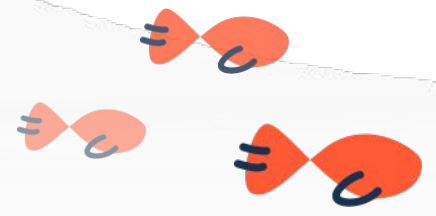
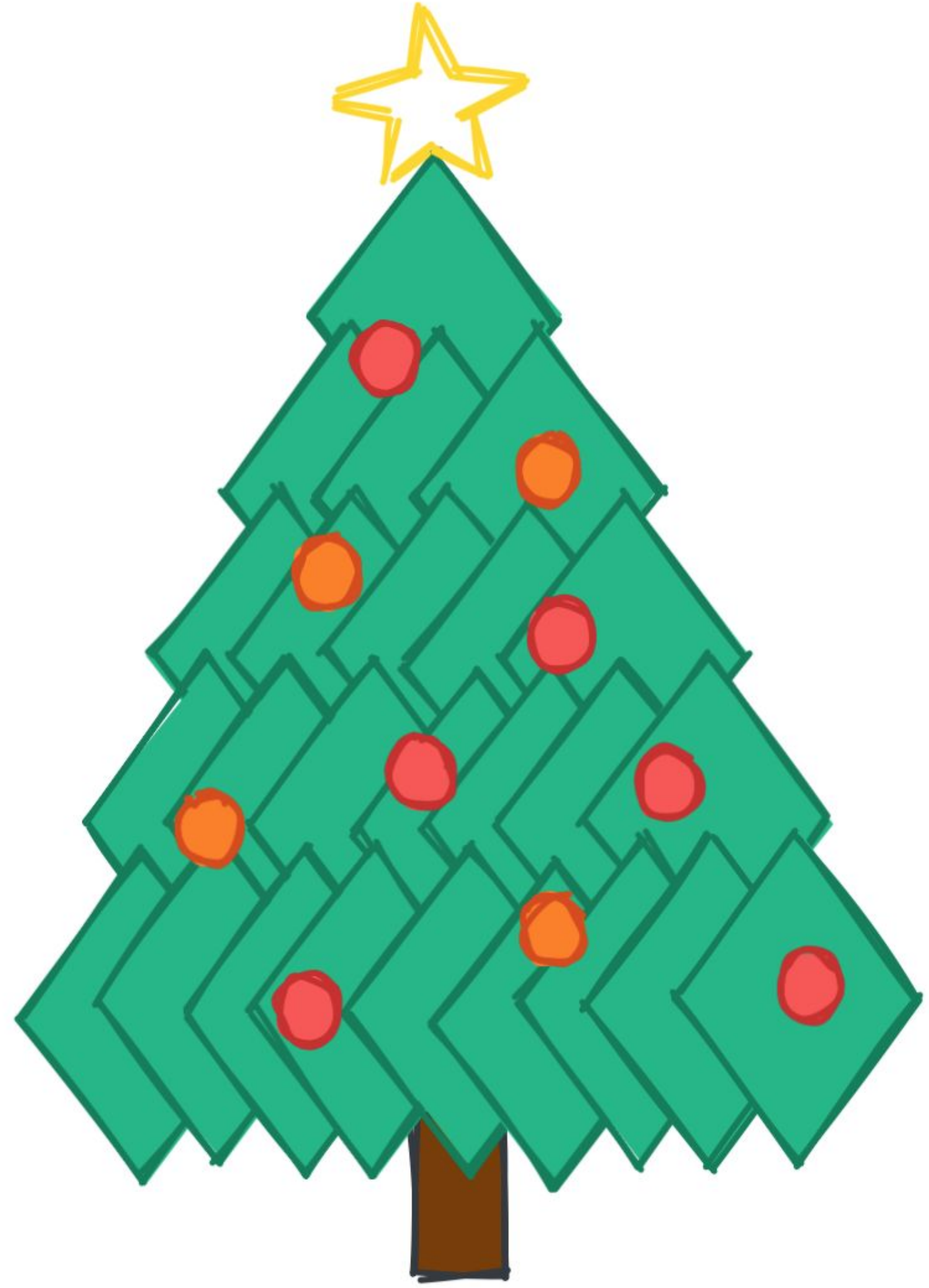
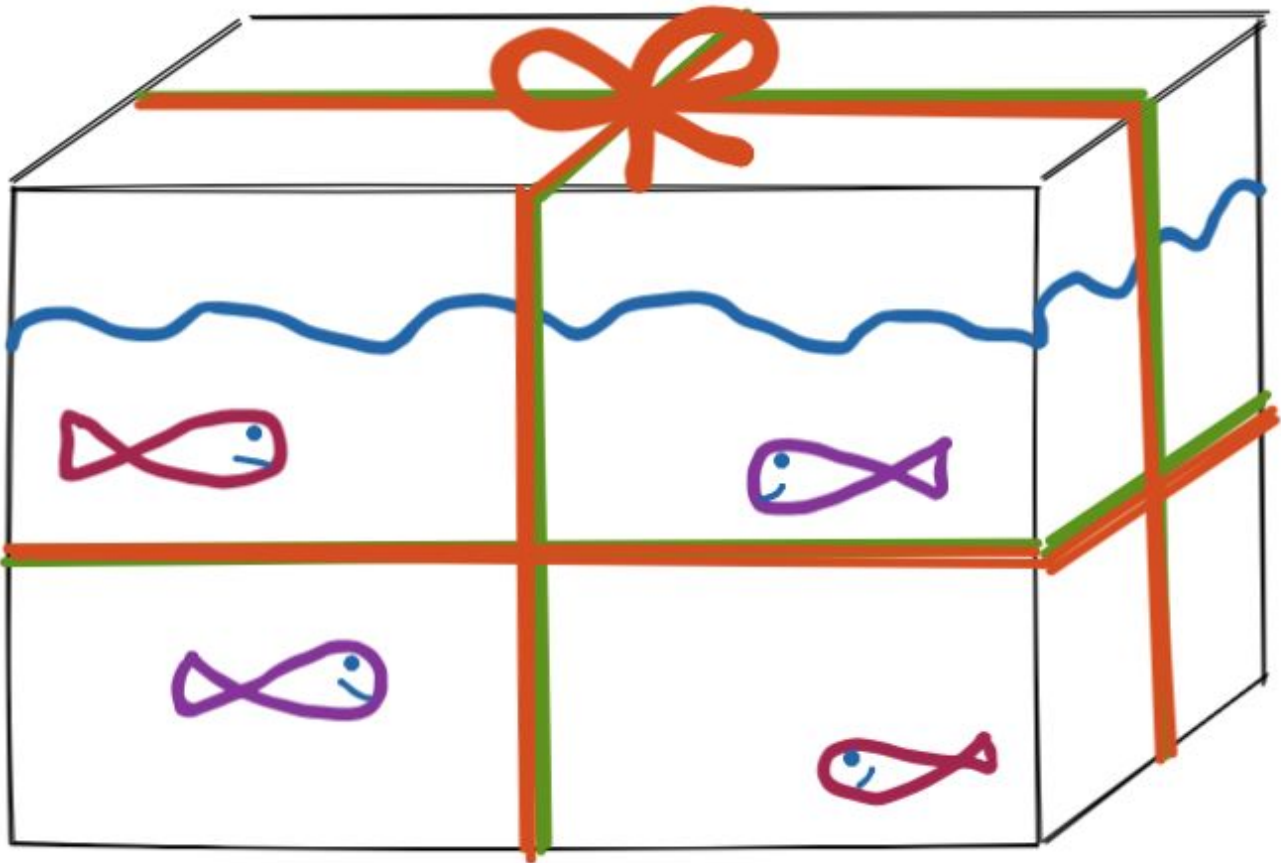
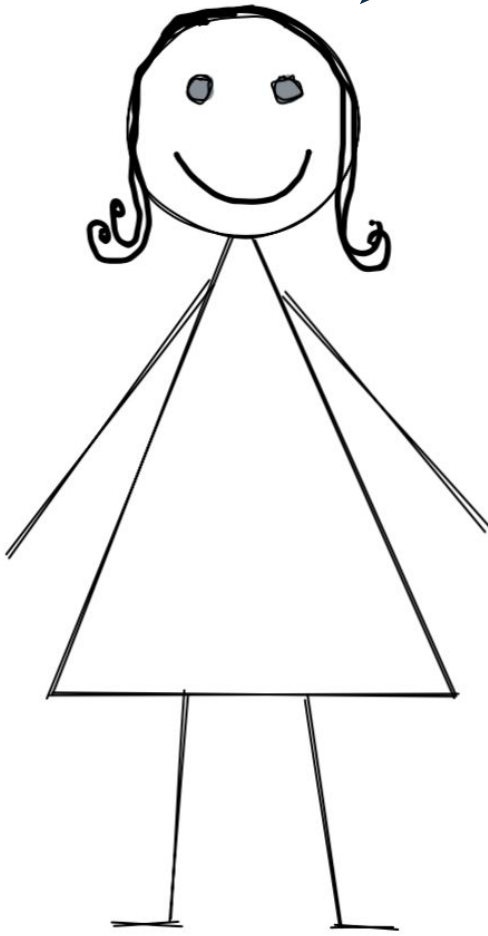


Our mission - Solved!

Christmas can come!

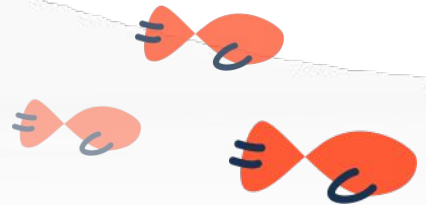
a happy
Larissa

Larissa's
new aquarium



Time to checkout the notebook!

- We will open breakout rooms.
- The notebook can be opened in Google Colab by clicking on the link in the Meetup repository.
- We will meet again in 35 min in the main room.
- Because we don't want to overwhelm the server with the requests of all participants in the meetup, we commented out a piece of code. Feel free to run it after the meetup!



Q&A

