LXML tutorial

R10922192 許雅晴

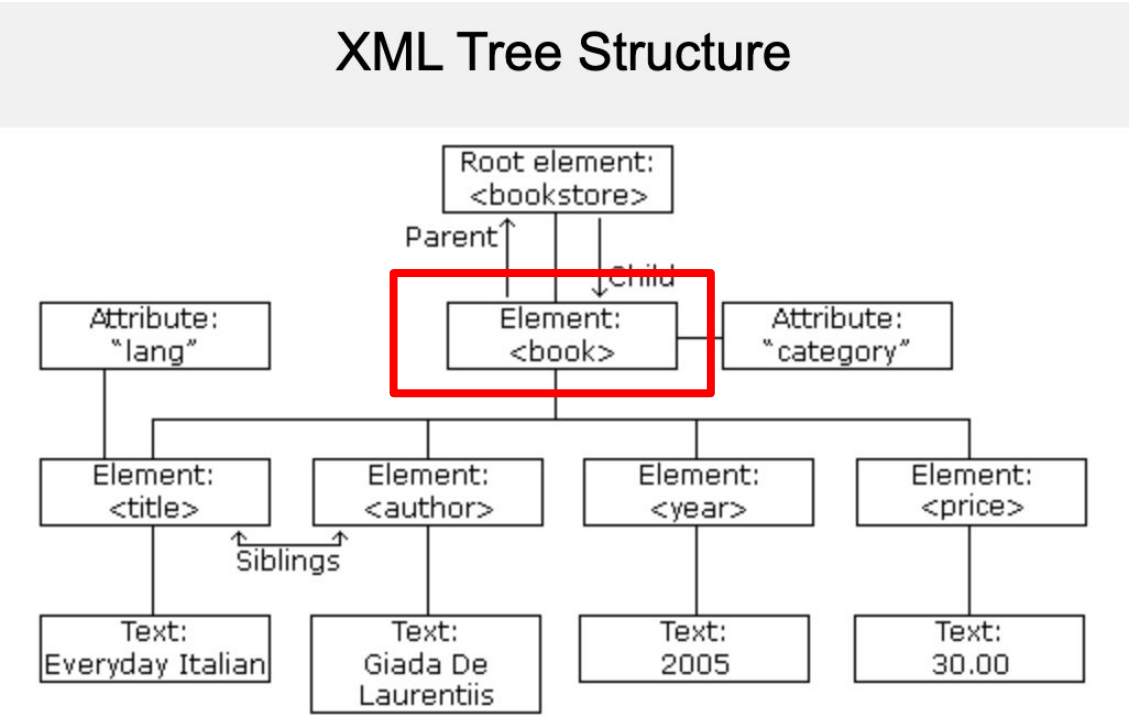
XML(eXtensible Markup Language)

- A XML document can be represented by a XML tree.
- Each node of the tree correspond to a XML tag.

<img> … </img>

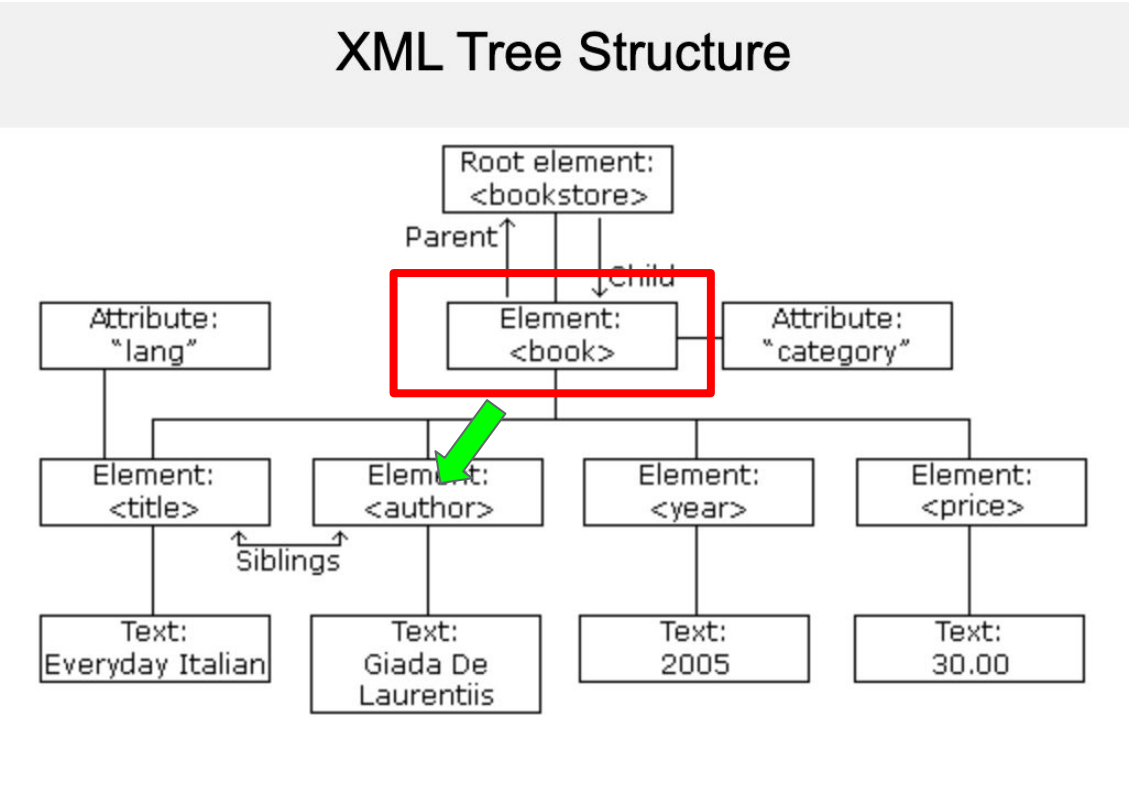
(a XML tag)

What is XML tree



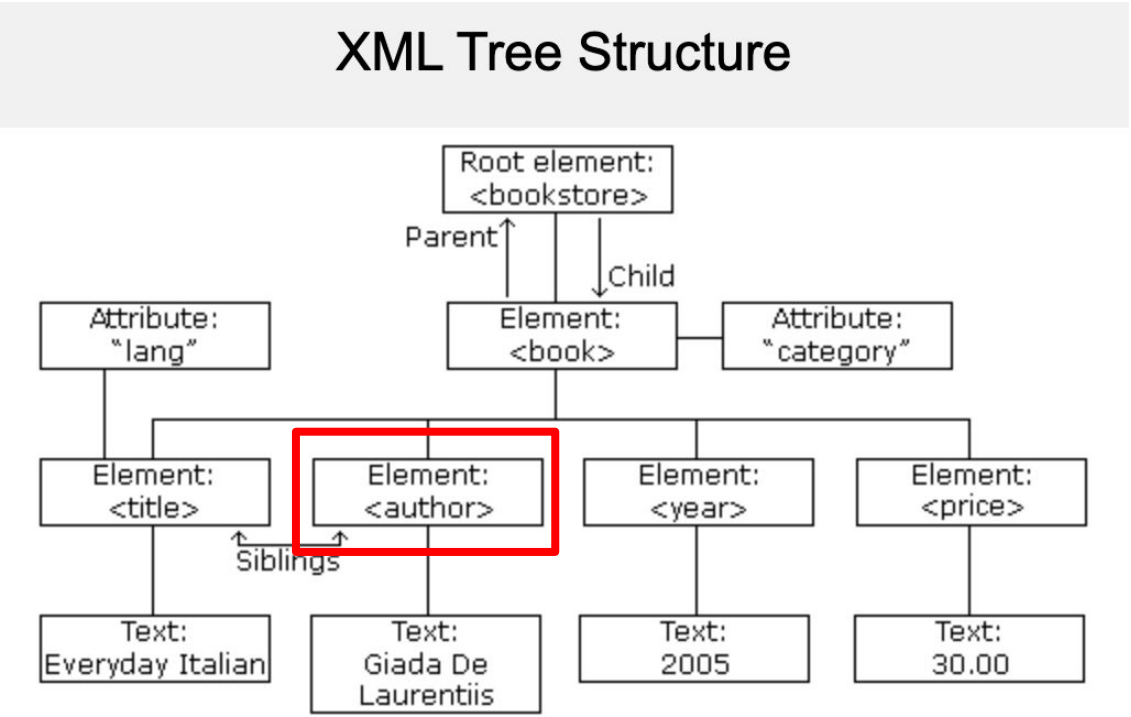

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

What is XML tree



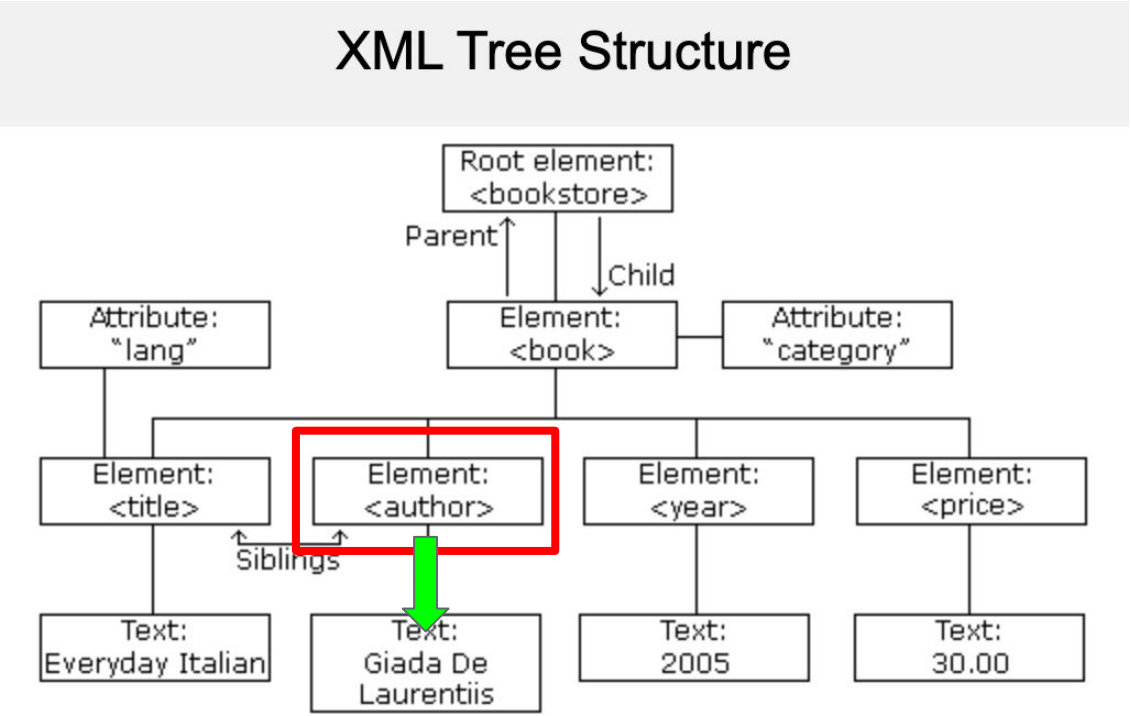

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

What is XML tree



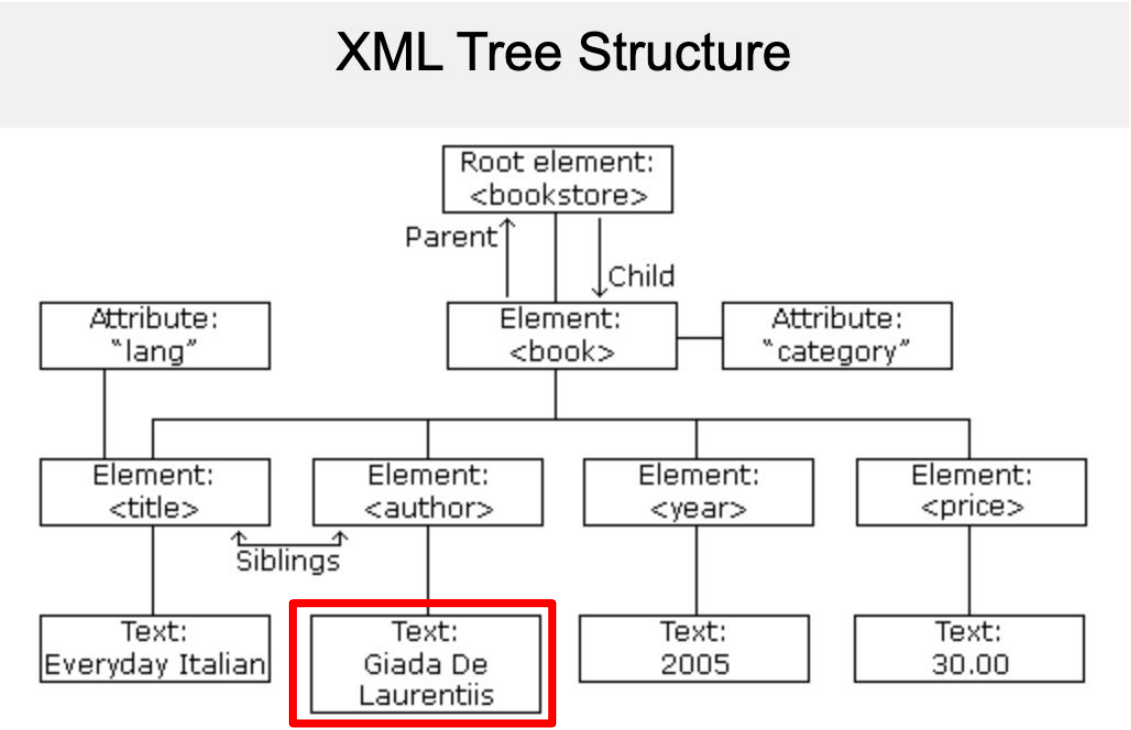

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

What is XML tree




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

What is XML tree




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

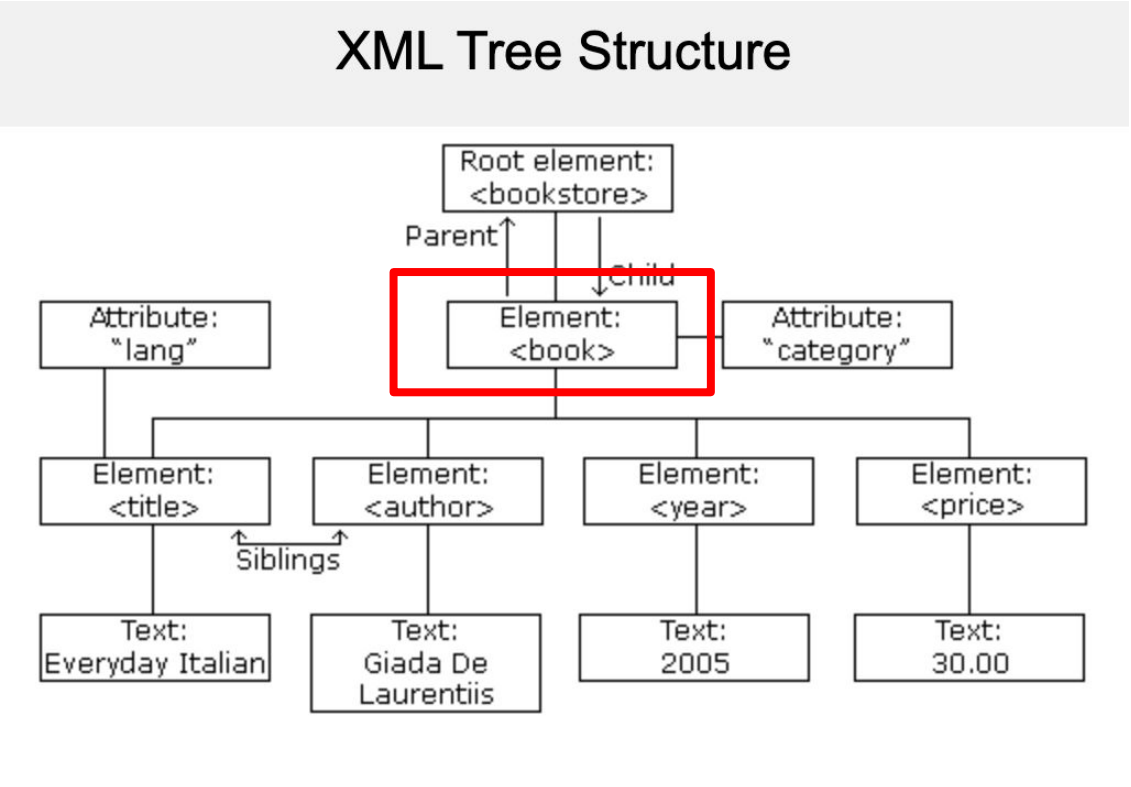
XPath syntax

A simple yet powerful tool to select node-set in XML document.

Expression	Description
nodename	Selects all nodes with the name "nodename"
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
@	Selects attributes
nodename[n]	Selects n-th node with the name “nodename”
nodename[@attr]	Selects nodename with attribute: “attr”
nodename[@attr=”ibute”]	Selects nodename with attribute: “attr” and its value is “ibute”

XPath examples

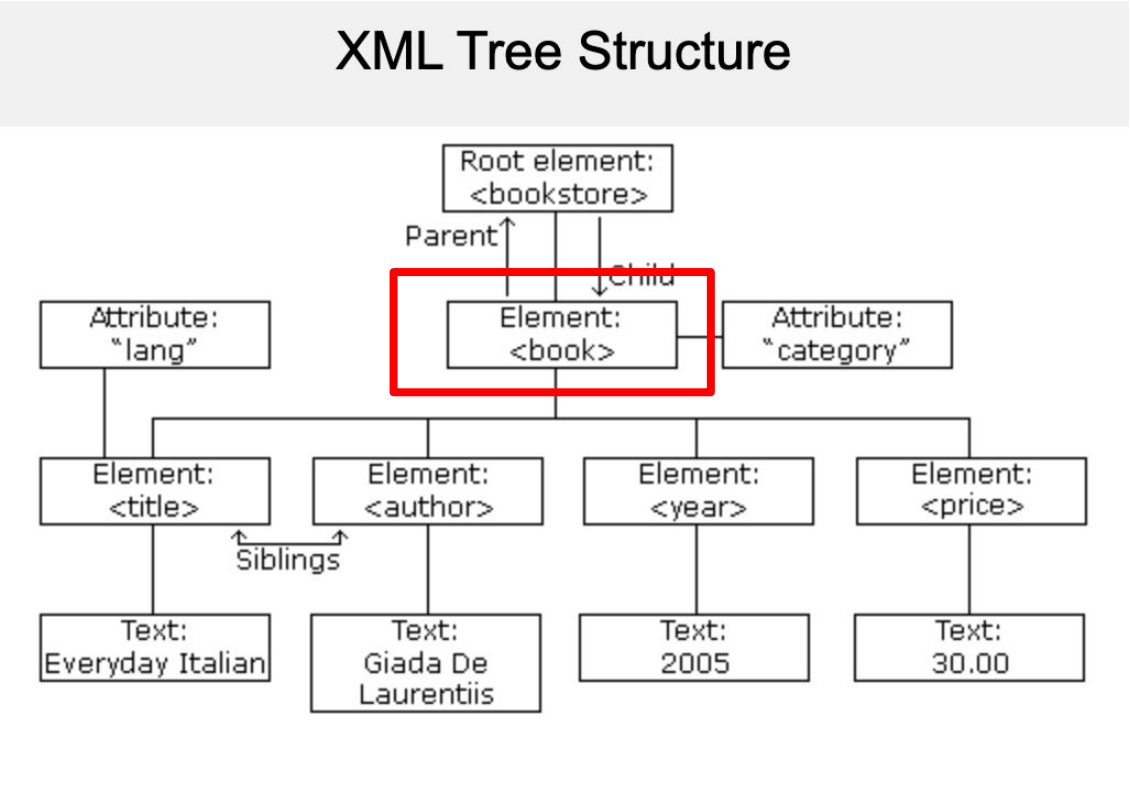
List books under all bookstore //bookstore/book




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

List book[1] under all bookstore

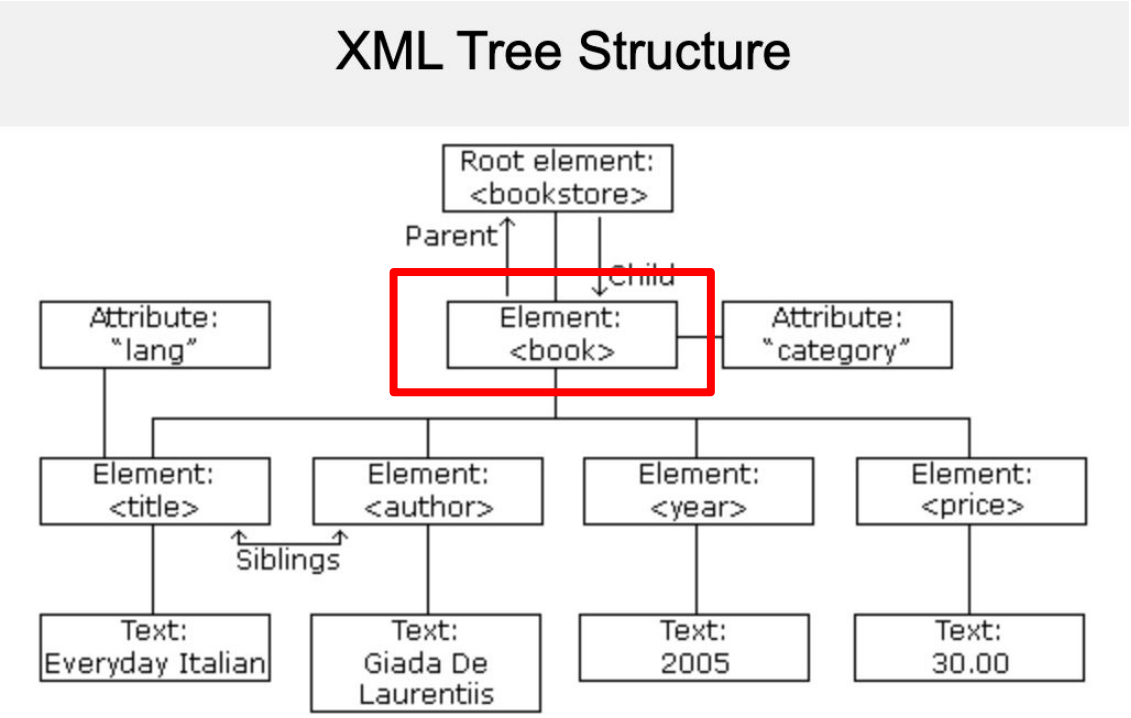
//bookstore/book[1]




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

All book with category=web

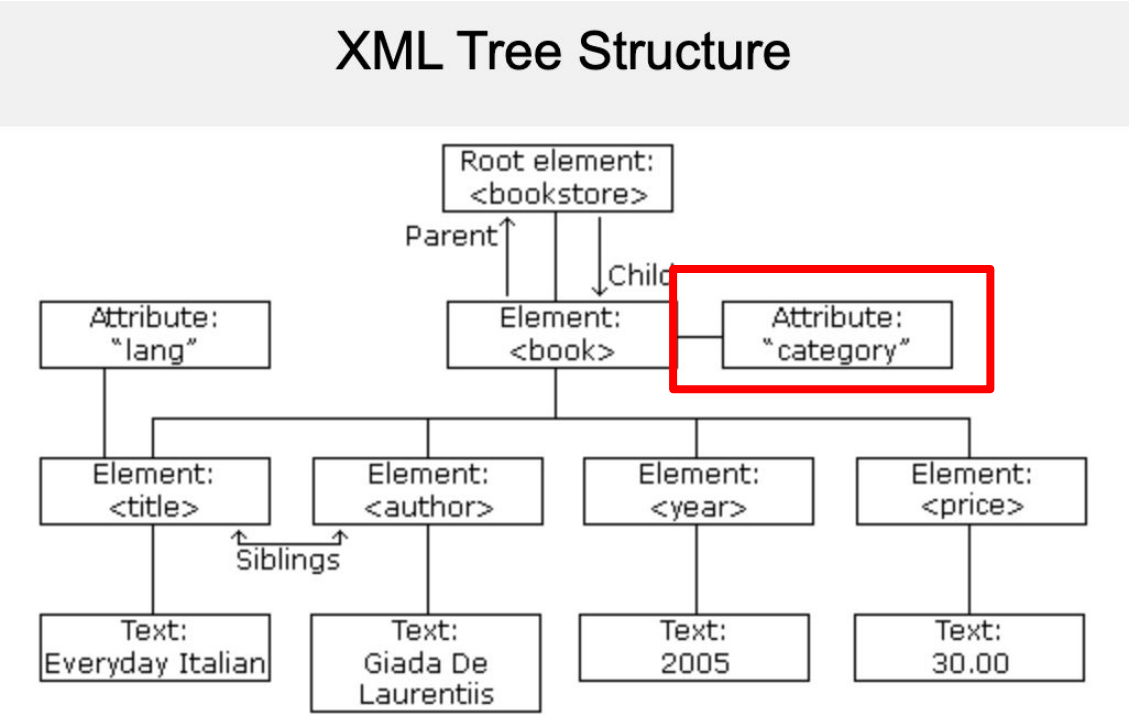
//book[@category="web"]




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

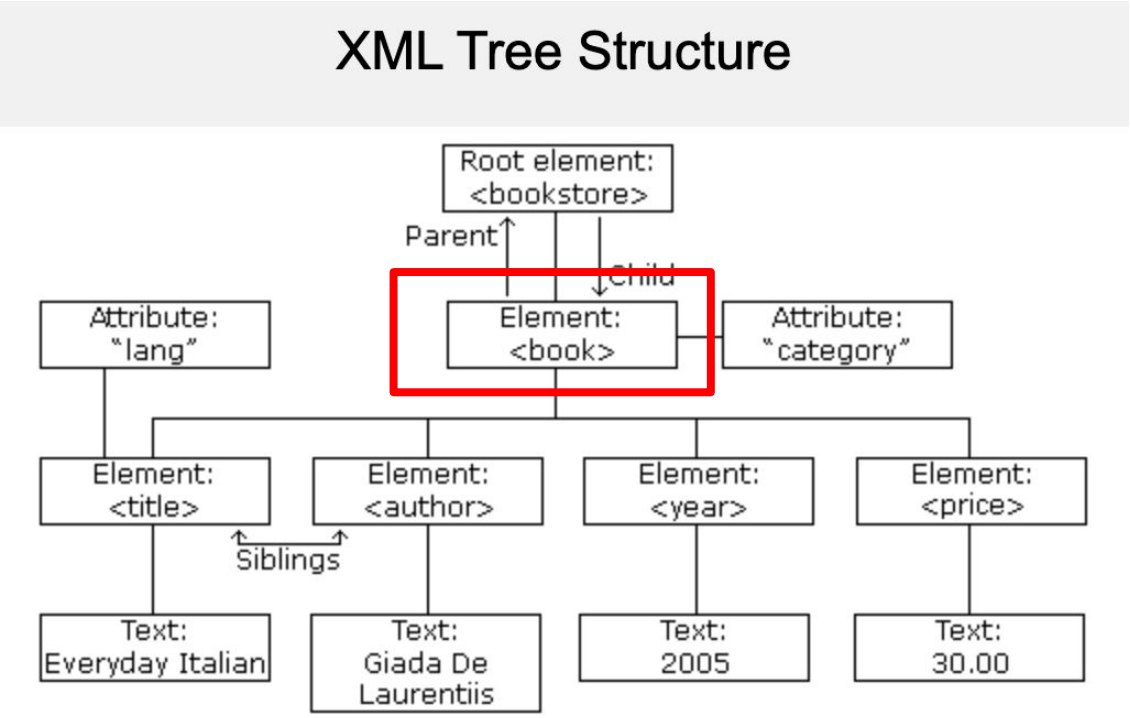
All category value of book

//book/@category




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

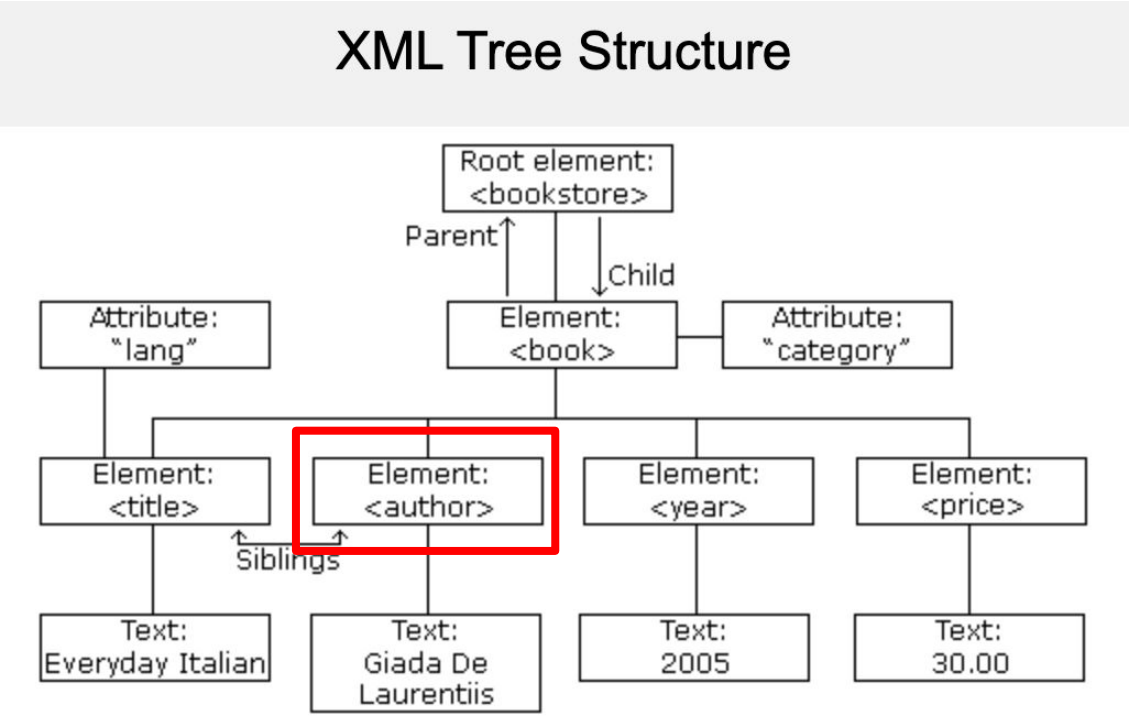
List book[2] under all bookstore //bookstore/book[2]




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Author under book

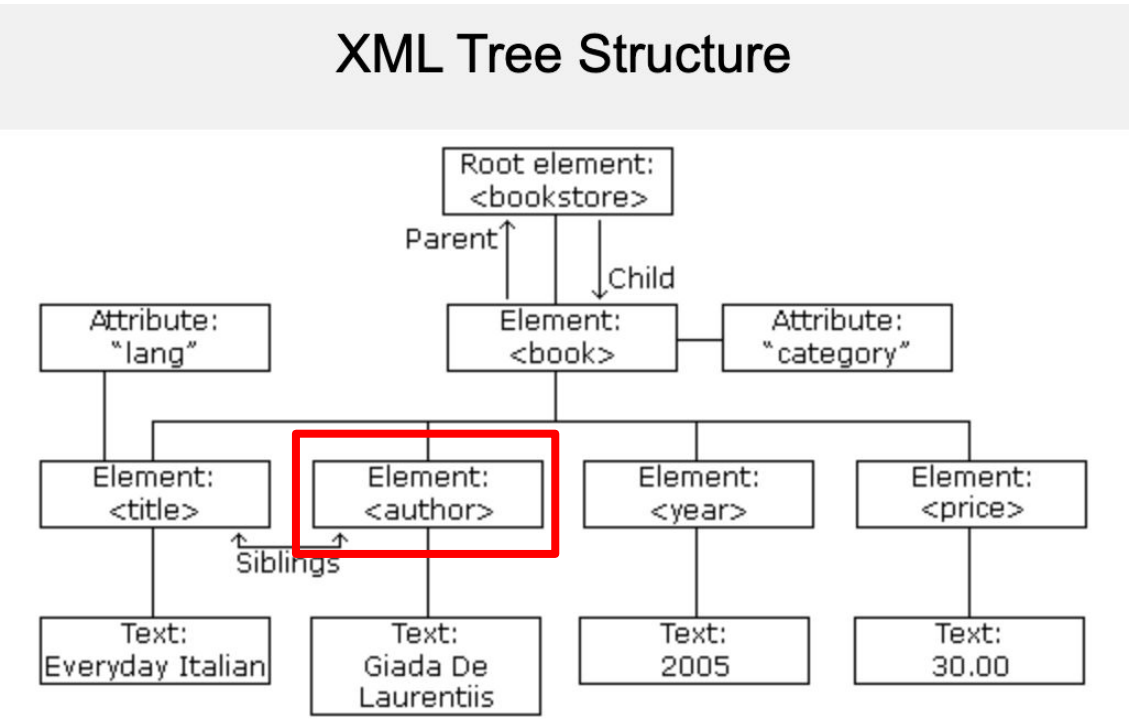
//bookstore/book/author




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Author under book[2]

//bookstore/book[2]/author

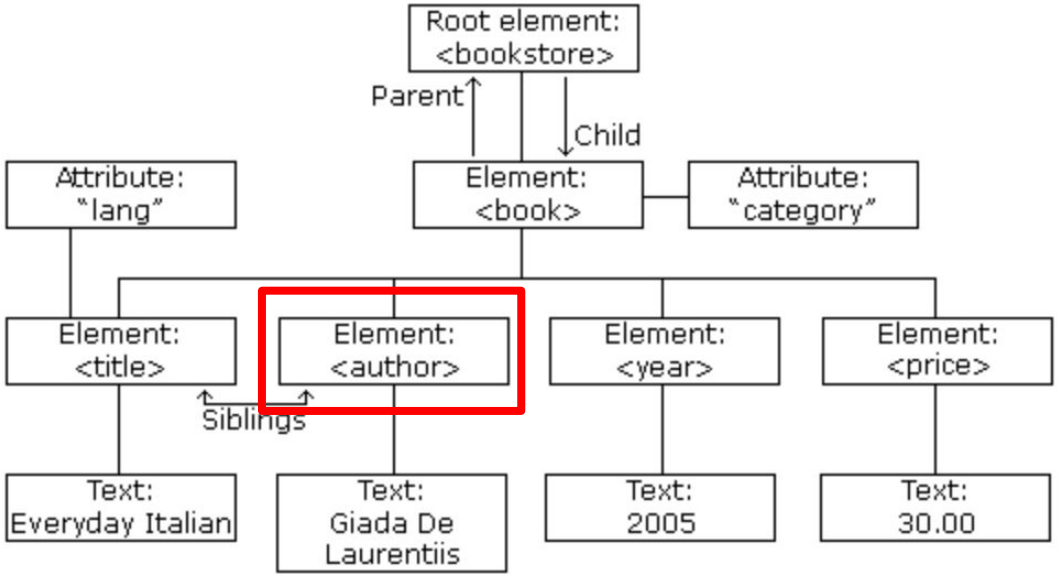



```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

book[2]

Any auther under root

XML Tree Structure



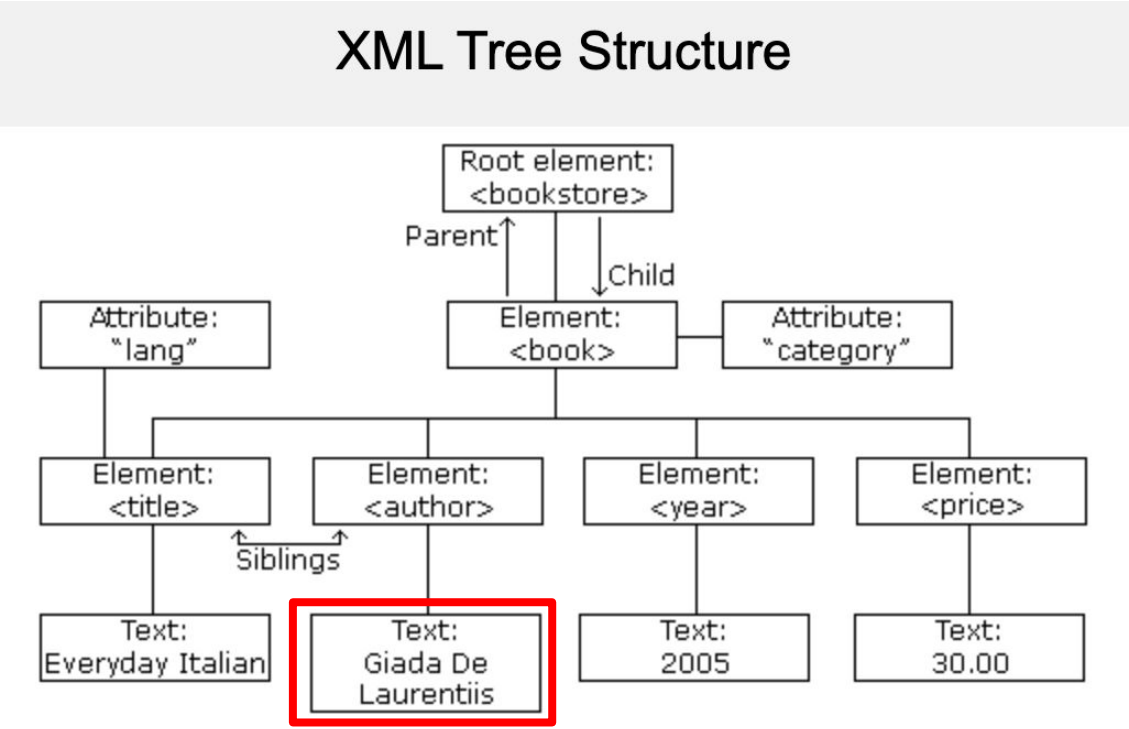
//author

root

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Text under author

//bookstore/book/author/text()




```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

XPath Helper

Install XPath Helper

- XPath Helper: link

Video Link

LXML

Install python packages

- Install lxml, requests by pip
  - pip install lxml requests

Document, tutorial

- Requests Document: https://requests.readthedocs.io/en/master/
- LXML Document: https://lxml.de/lxmlhtml.html
- LXML + requests tutorial: link

Pipeline

1. Use “requests” to get HTML document
2. Use Etree to parse the HTML document
3. Use XPath to select elements in HTML document

Detailed tutorial

Import requests, lxml

```
[In [1]: import requests

[In [2]: from lxml import etree
```

Request to web server

```
In [3]: response = requests.get("https://www.csie.ntu.edu.tw/news/news.php?class
   ...: =101")

[In [4]: print(response.status_code) # 200 == "Success"
200
```

Get HTML content

```
In [5]: html_text = response.content.decode() # Fetch HTML content and decode in
   ...: to UTF-8

In [6]: html_text[:50]
Out[6]: '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Trans'
```

Parse HTML by lxml.etree

```
In [10]: root = etree.HTML(html_text)
```

Select elements by XPath

```
titles = root.xpath("//div[1]/div/div[2]/div/div/div[2]/                    ")
print( [title.text for title in titles] )
```

```
['110學年度資訊學群畢業典禮照片及影片', '【國立臺灣大學】 公告　主旨：公告本校「博、碩士學位論文違反學術倫理案件處理要點」修正條文對照表及全文如附件。',
```

```
dates = root.xpath("//div[1]/div/div[2]/div/div/div[2]/                    ")
print( [date.text for date in dates] )
```

```
['2022-05-26', '2022-10-27', '2022-10-27', '2022-10-25', '2022-10-17', '2022-09-30', '2022-09-26', '2022-09-21', '2022-09-12
```

Python HW

Requirement

- Crawl the announcement page of CSIE website within specified range of dates. Please use the request headers in TA sample codes:

  https://github.com/Shelley1214/ItC-python-hw-sample-code

- The results should contain but not limited to the following fields:
  - Post date
    - e.g. `2022-05-26`
  - Title
    - e.g. `110學年度資訊學群畢業典禮照片及影片`
  - Content
    - recursively find all the *text* in `<div class="editor content">`

Requirement

- Please save the results to a CSV file which **can be opened by Excel** using utf-8. Please note that:
  - User should be able to specify the path to write the CSV file with `--output` argument.
  - Formats
    - Each record in one line.
    - Fields of a record are seperated by a comma “,” with no space or new line between.
    - Strings in the CSV file are enclosed by a pair of double quotation mark (e.g. "I’m string "). And any double quote within a string should be replace by 2 double quotation mark. For instance, the string: “Prof. Yuguang “Michael” Fang, University of Florida” should be replaced by “Prof. Yuguang ““Michael”” Fang, University of Florida”

What TAs will run

```
python3 main.py --start-date [start date] --end-date [end date] --output [out filename]
```

- `--start-date` and `--end-date` will be in the format of `[Year]-[month]-[day]`. For instance, `2022-05-26`
- `--output` is the csv filename to save. For instance, `output.csv`.

Evaluation

- (10pts) Run without error
- (10pts) Correctly parse arguments
- (20pts) Output files to correct place and can be opened by Excel and pandas.read_csv
- (10pts) Sort by post date (current to before)
- (50pts) Contents are correct
- (-20pts) Sleep 0.1 seconds before every request. This rule is **required**. You will lose points if you violate the rule.
- You must NOT use commands such as sudo or other commands that interfere with the environment; any malicious attempt against the environment will lead to zero point in this assignment.