simplilearn | PURDUE UNIVERSITY

Data Science With Python

DATA AND
ARTIFICIAL INTELLIGENCE

Web Scraping with BeautifulSoup

Learning Objectives

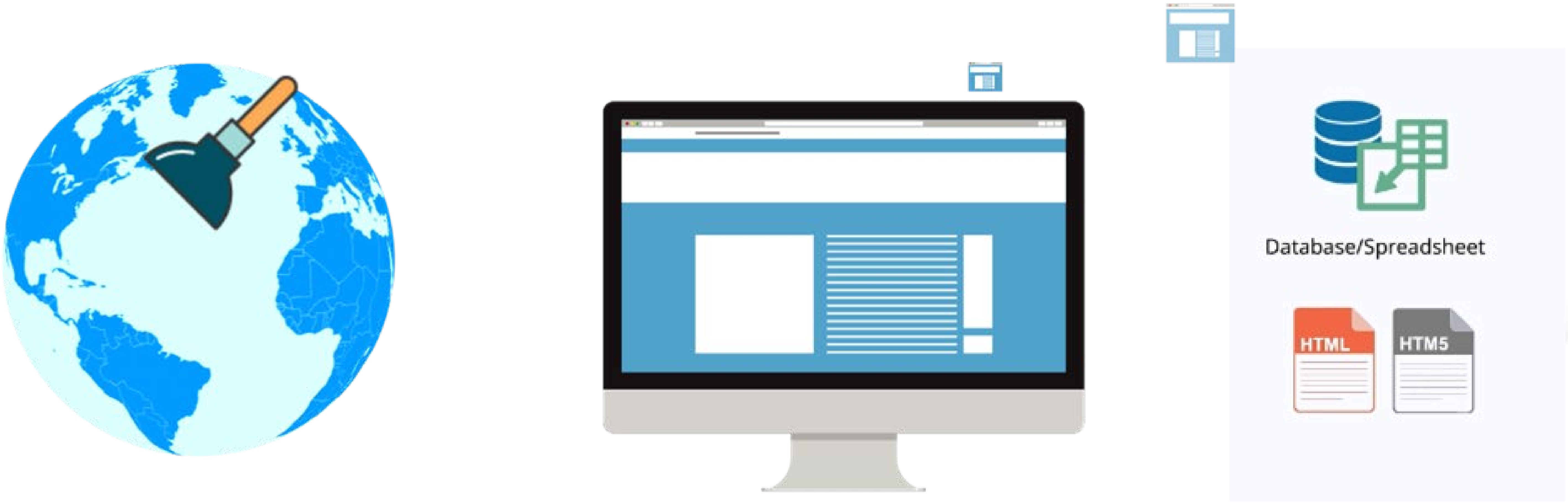
By the end of this lesson, you will be able to:

- Define web scraping and explain its importance
- List the steps involved in the web scraping process
- Describe basic terminologies, such as parser, object, and tree associated with the BeautifulSoup
- Explain various operations, such as searching, modifying, and navigating the tree to yield the required result

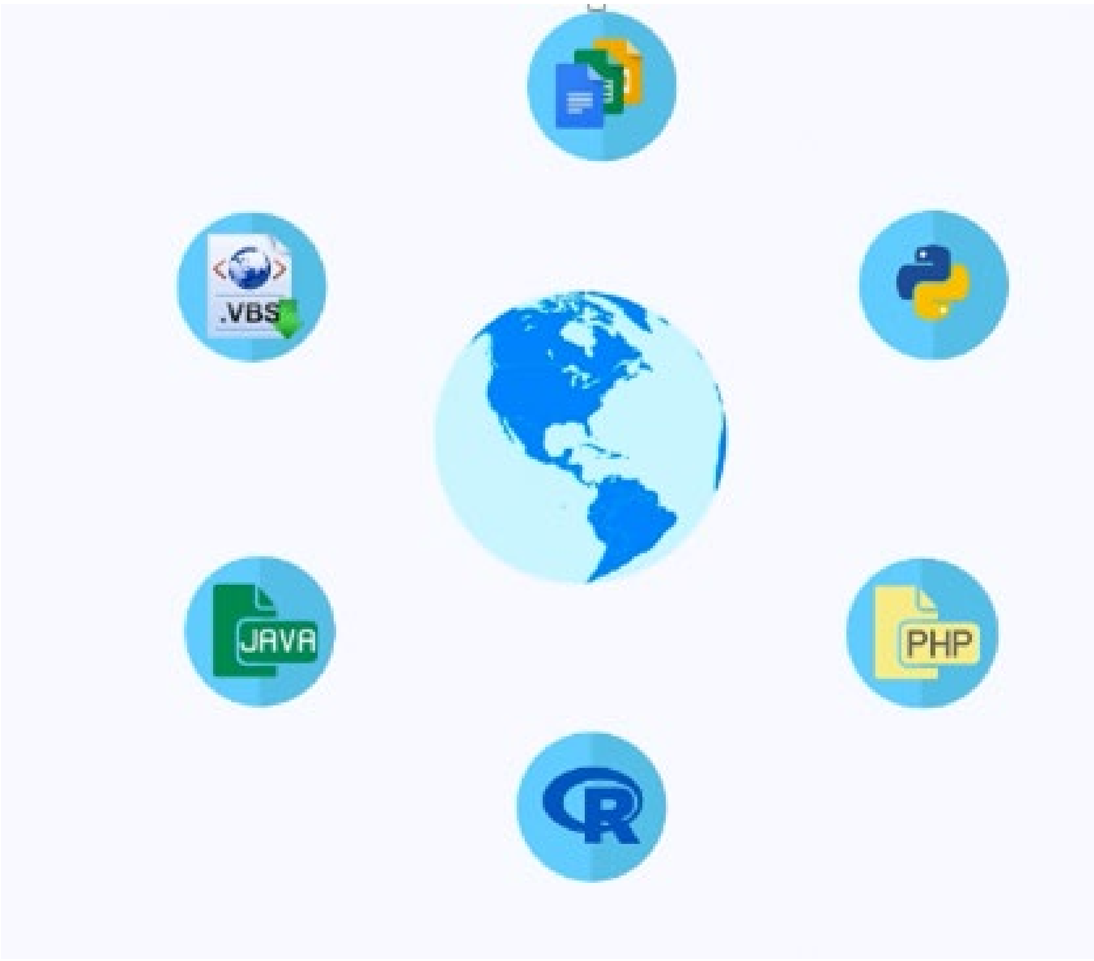
Web Scraping

What Is Web Scraping?

Web scraping is a computer software technique for extracting information from websites in an automated fashion.

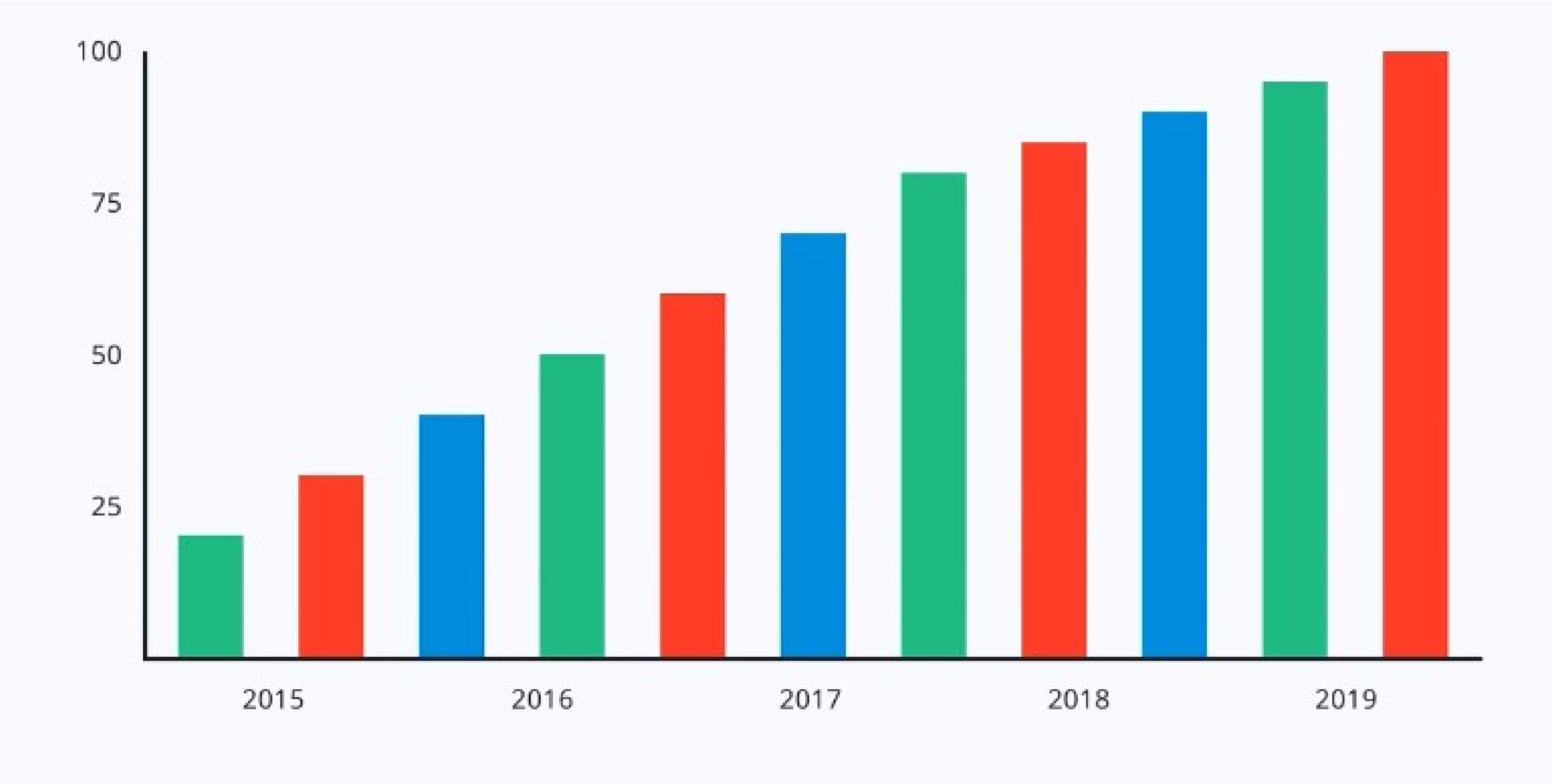


What Is Web Scraping?



Why Web Scraping

Every day, you find yourself in a situation where you need to extract data from the web.



Why Web Scraping

Web Scraping Process

Web Scraping Process: Basic Preparation

There are two basic things to consider before setting up the web scraping process:

- Understanding the target data on the Internet
- Finalizing the list of websites

Web Scraping Process

Once you have understood the target data and finalized the list of websites, you need to design the web scraping process.

The steps involved in a typical web scraping process are as follows :

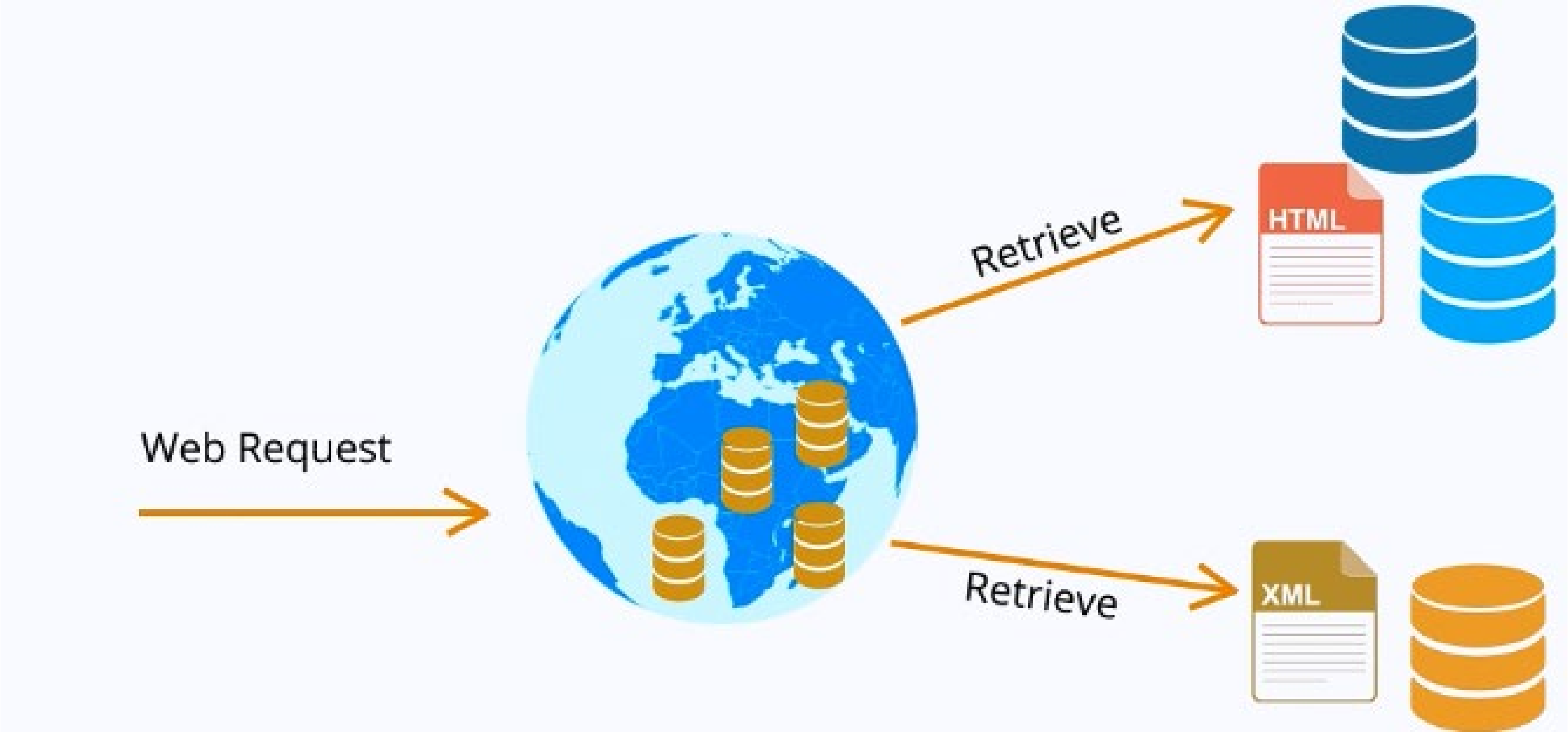
Web Request

Step 1: A web request is sent to the targeted website to collect the required data.

Web Scraping Process

Once you have understood the target data and finalized the list of websites, you need to design the web scraping process.

The steps involved in a typical web scraping process are as follows:

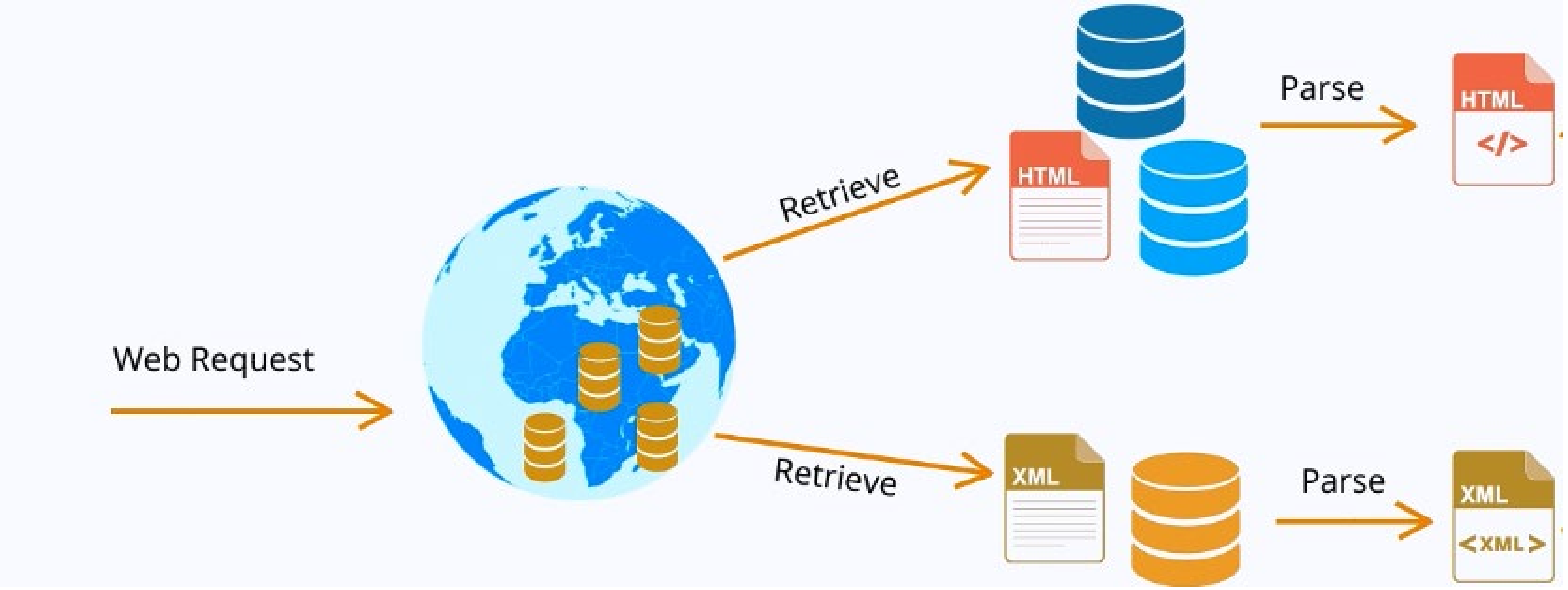


Step 2: The information is retrieved from the targeted website in HTML or XML format from web.

Web Scraping Process

Once you have understood the target data and finalized the list of websites, you need to design the web scraping process.

The steps involved in a typical web scraping process are as follows:

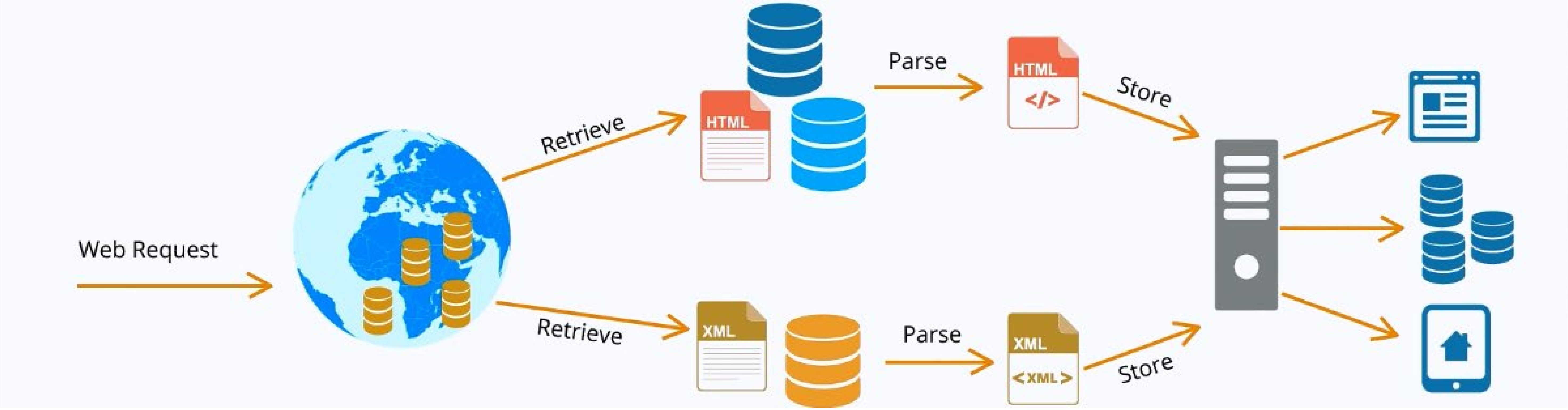


Step 3: The retrieved information is parsed to the several parsers based on the data format. Parsing is a technique to read data and extract information from the available document.

Web Scraping Process

Once you have understood the target data and finalized the list of websites, you need to design the web scraping process.

The steps involved in a typical web scraping process are as follows:



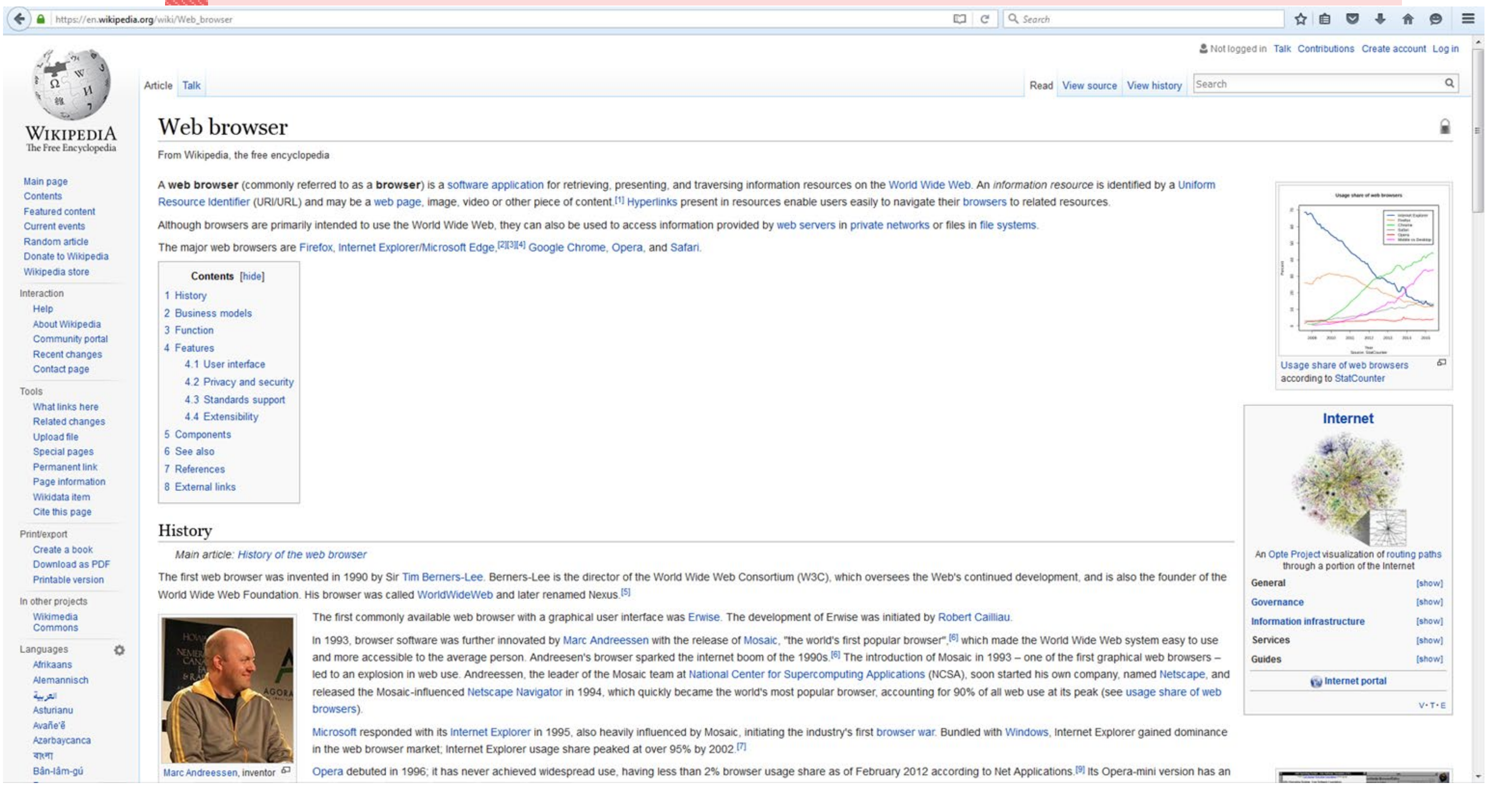
Step 4: The parsed data is stored in the desired format. You can follow the same process to scrap another targeted web.

Web Scraping Software

A web scraping software will interact with websites in the same way as your web browser.

A Web scraper is used to extract the information from web in routine and automated manner.

Web Browser



Web browser

From Wikipedia, the free encyclopedia

A **web browser** (commonly referred to as a **browser**) is a software application for retrieving, presenting, and traversing information resources on the World Wide Web. An *information resource* is identified by a Uniform Resource Identifier (URI/URL) and may be a web page, image, video or other piece of content.[1] Hyperlinks present in resources enable users easily to navigate their browsers to related resources.

Although browsers are primarily intended to use the World Wide Web, they can also be used to access information provided by web servers in private networks or files in file systems.

The major web browsers are Firefox, Internet Explorer/Microsoft Edge,[2][3][4] Google Chrome, Opera, and Safari.

Contents [hide]

1 History
2 Business models
3 Function
4 Features
4.1 User interface
4.2 Privacy and security
4.3 Standards support
4.4 Extensibility
5 Components
6 See also
7 References
8 External links

Usage share of web browsers according to StatCounter

An Opte Project visualization of routing paths through a portion of the Internet

History

Main article: History of the web browser

The first web browser was invented in 1990 by Sir Tim Berners-Lee. Berners-Lee is the director of the World Wide Web Consortium (W3C), which oversees the Web's continued development, and is also the founder of the World Wide Web Foundation. His browser was called WorldWideWeb and later renamed Nexus.[5]

Marc Andreessen, inventor

The first commonly available web browser with a graphical user interface was Erwise. The development of Erwise was initiated by Robert Cailliau.

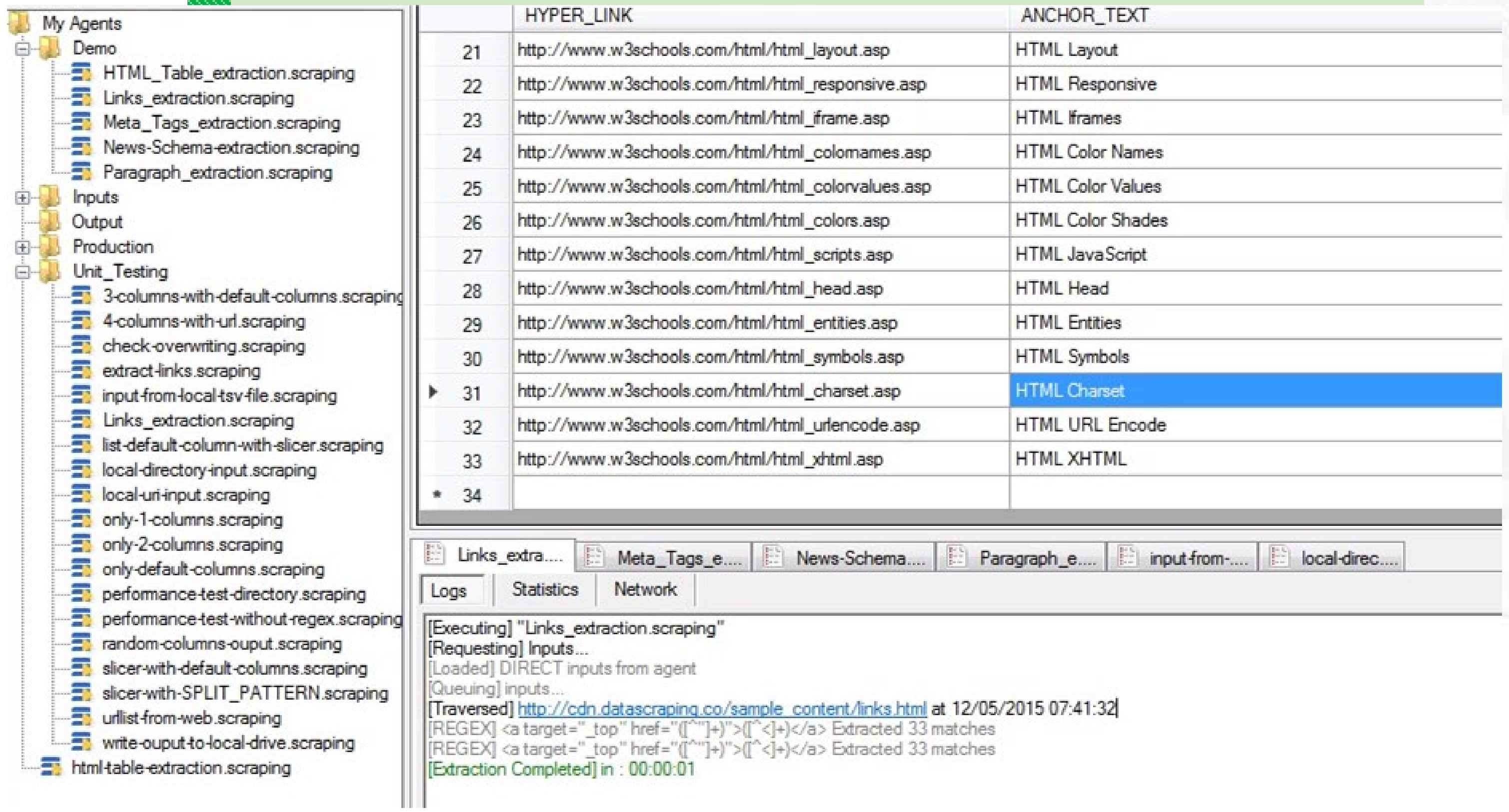
In 1993, browser software was further innovated by Marc Andreessen with the release of Mosaic, "the world's first popular browser",[6] which made the World Wide Web system easy to use and more accessible to the average person. Andreesen's browser sparked the internet boom of the 1990s.[6] The introduction of Mosaic in 1993 – one of the first graphical web browsers – led to an explosion in web use. Andreessen, the leader of the Mosaic team at National Center for Supercomputing Applications (NCSA), soon started his own company, named Netscape, and released the Mosaic-influenced Netscape Navigator in 1994, which quickly became the world's most popular browser, accounting for 90% of all web use at its peak (see usage share of web browsers).

Microsoft responded with its Internet Explorer in 1995, also heavily influenced by Mosaic, initiating the industry's first browser war. Bundled with Windows, Internet Explorer gained dominance in the web browser market; Internet Explorer usage share peaked at over 95% by 2002.[7]

Opera debuted in 1996; it has never achieved widespread use, having less than 2% browser usage share as of February 2012 according to Net Applications.[9] Its Opera-mini version has an

Displays the data

Web Scraping Software


My Agents
Demo
HTML_Table_extraction.scraping
Links_extraction.scraping
Meta_Tags_extraction.scraping
News-Schema-extraction.scraping
Paragraph_extraction.scraping
Inputs
Output
Production
Unit_Testing
3-columns-with-default-columns.scraping
4-columns-with-url.scraping
check-overwriting.scraping
extract-links.scraping
input-from-local-tsv-file.scraping
Links_extraction.scraping
list-default-column-with-slicer.scraping
local-directory-input.scraping
local-uri-input.scraping
only-1-columns.scraping
only-2-columns.scraping
only-default-columns.scraping
performance-test-directory.scraping
performance-test-without-regex.scraping
random-columns-ouput.scraping
slicer-with-default-columns.scraping
slicer-with-SPLIT_PATTERN.scraping
urllist-from-web.scraping
write-ouput-to-local-drive.scraping
html-table-extraction.scraping
Links_extra.... Meta_Tags_e.... News-Schema.... Paragraph_e.... input-from-.... local-direc....
Logs Statistics Network

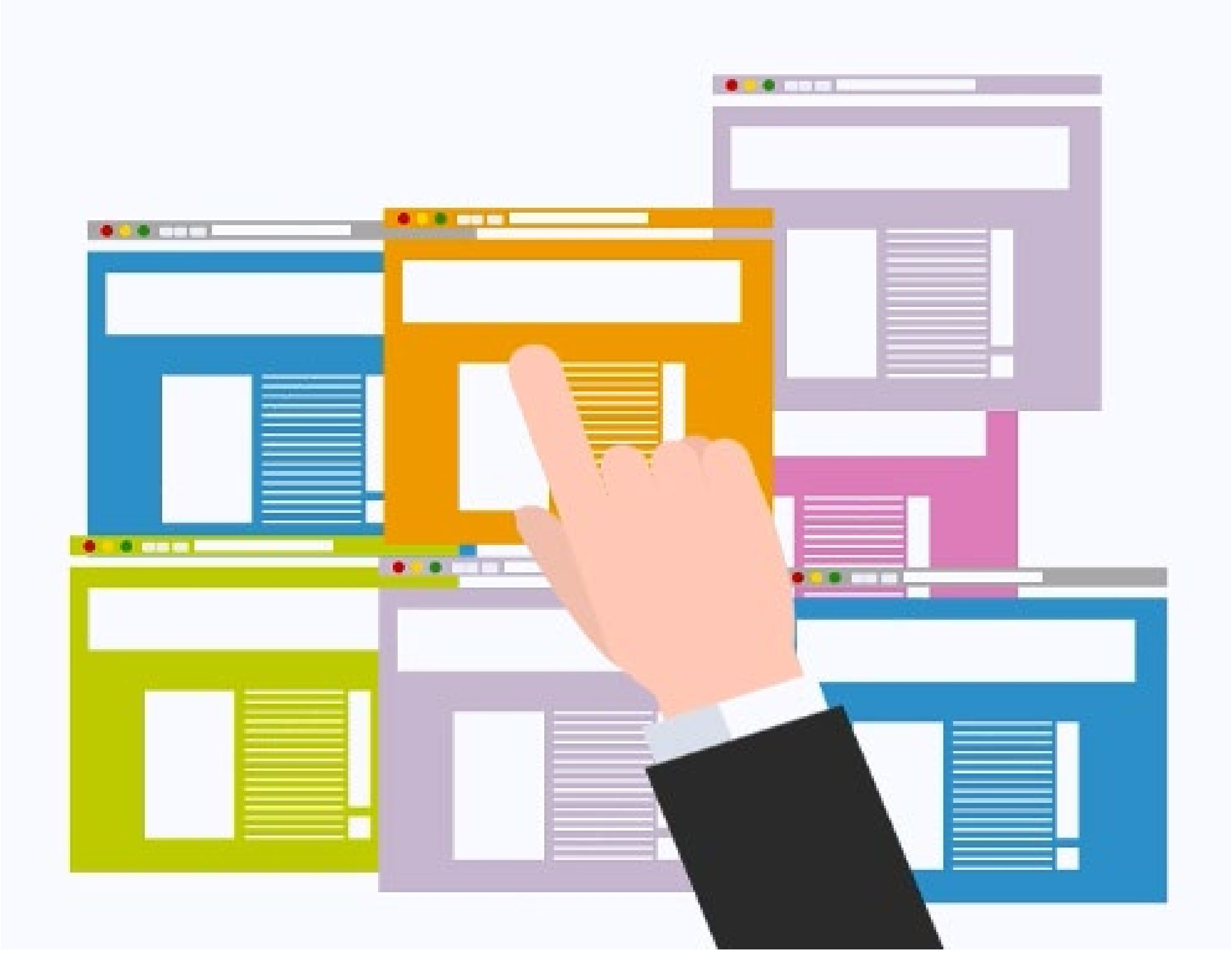
	HYPER_LINK	ANCHOR_TEXT
21	http://www.w3schools.com/html/html_layout.asp	HTML Layout
22	http://www.w3schools.com/html/html_responsive.asp	HTML Responsive
23	http://www.w3schools.com/html/html_iframe.asp	HTML Iframes
24	http://www.w3schools.com/html/html_colornames.asp	HTML Color Names
25	http://www.w3schools.com/html/html_colorvalues.asp	HTML Color Values
26	http://www.w3schools.com/html/html_colors.asp	HTML Color Shades
27	http://www.w3schools.com/html/html_scripts.asp	HTML JavaScript
28	http://www.w3schools.com/html/html_head.asp	HTML Head
29	http://www.w3schools.com/html/html_entities.asp	HTML Entities
30	http://www.w3schools.com/html/html_symbols.asp	HTML Symbols
▸ 31	http://www.w3schools.com/html/html_charset.asp	HTML Charset
32	http://www.w3schools.com/html/html_urlencode.asp	HTML URL Encode
33	http://www.w3schools.com/html/html_xhtml.asp	HTML XHTML
* 34		

```
[Executing] "Links_extraction.scraping"
[Requesting] Inputs...
[Loaded] DIRECT inputs from agent
[Queuing] inputs...
[Traversed] http://cdn.datascraping.co/sample_content/links.html at 12/05/2015 07:41:32
[REGEX] <a target="_top" href="([^"]+)">([^<]+)</a> Extracted 33 matches
[REGEX] <a target="_top" href="([^"]+)">([^<]+)</a> Extracted 33 matches
[Extraction Completed] in : 00:00:01
```

Saves data from the web page to the local file or database

Web Scraping Considerations

Reading and understanding the legal information along with terms and conditions mentioned in the website is important.



Web Scraping Considerations

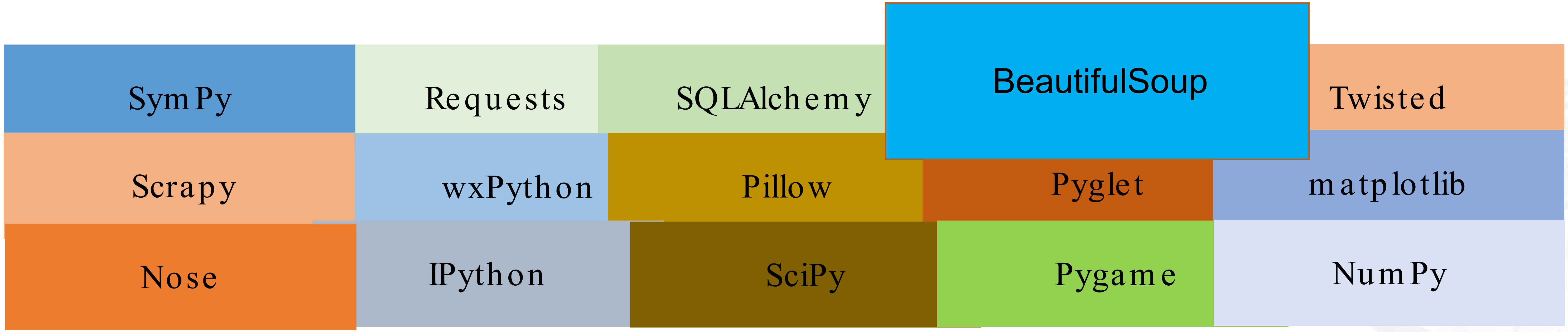
- Legal Constraints
- Notice
- Copyright
- Trademark Material
- Patented Information

Web Scraping Tool: BeautifulSoup

SymPy	Requests	SQLAlchemy	BeautifulSoup	Twisted
Scrapy	wxPython	Pillow	Pyglet	matplotlib
Nose	IPython	SciPy	Pygame	NumPy

Web Scraping Tool: BeautifulSoup

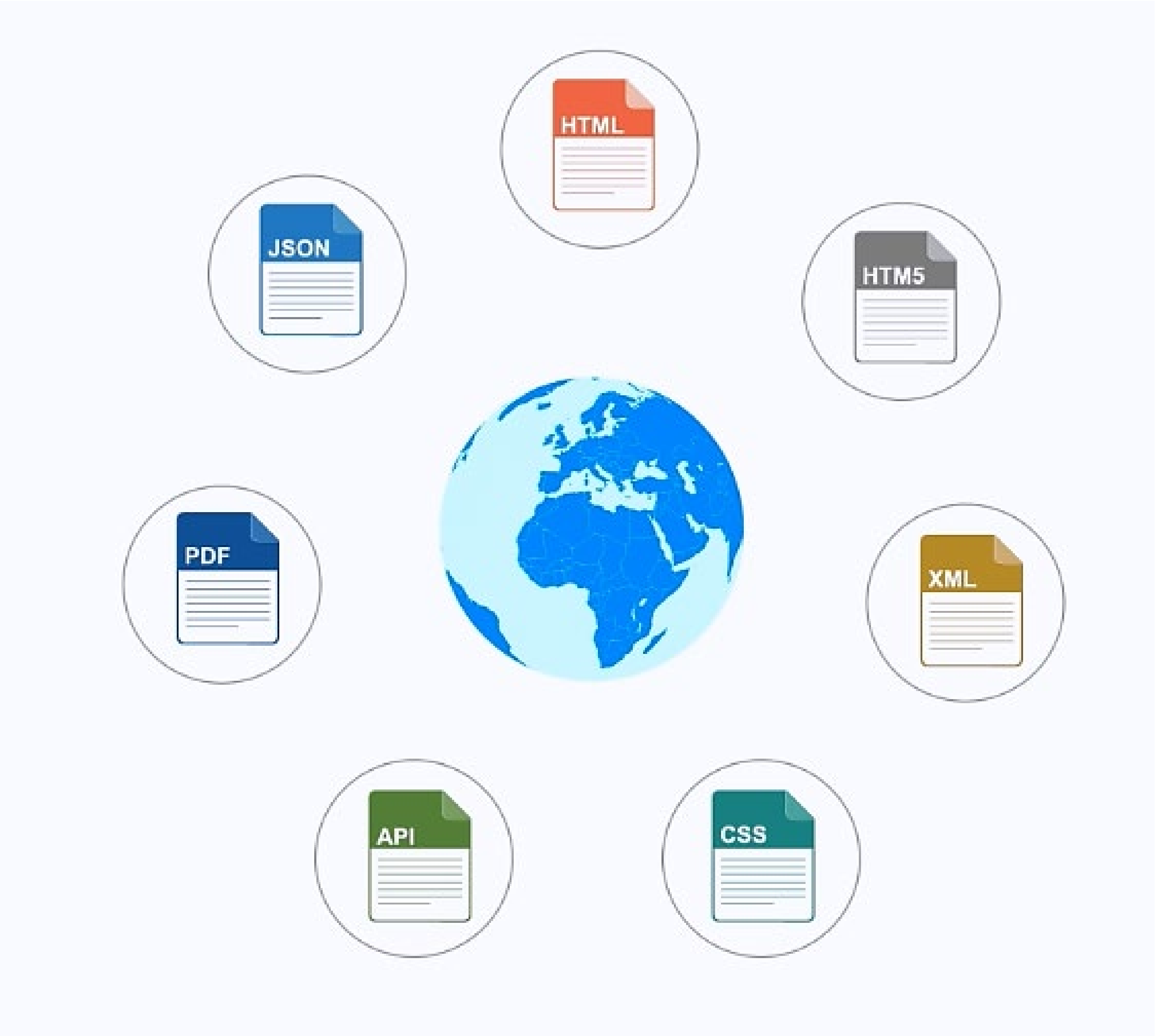
BeautifulSoup, is an easy, intuitive, and a robust Python library designed for web scraping.



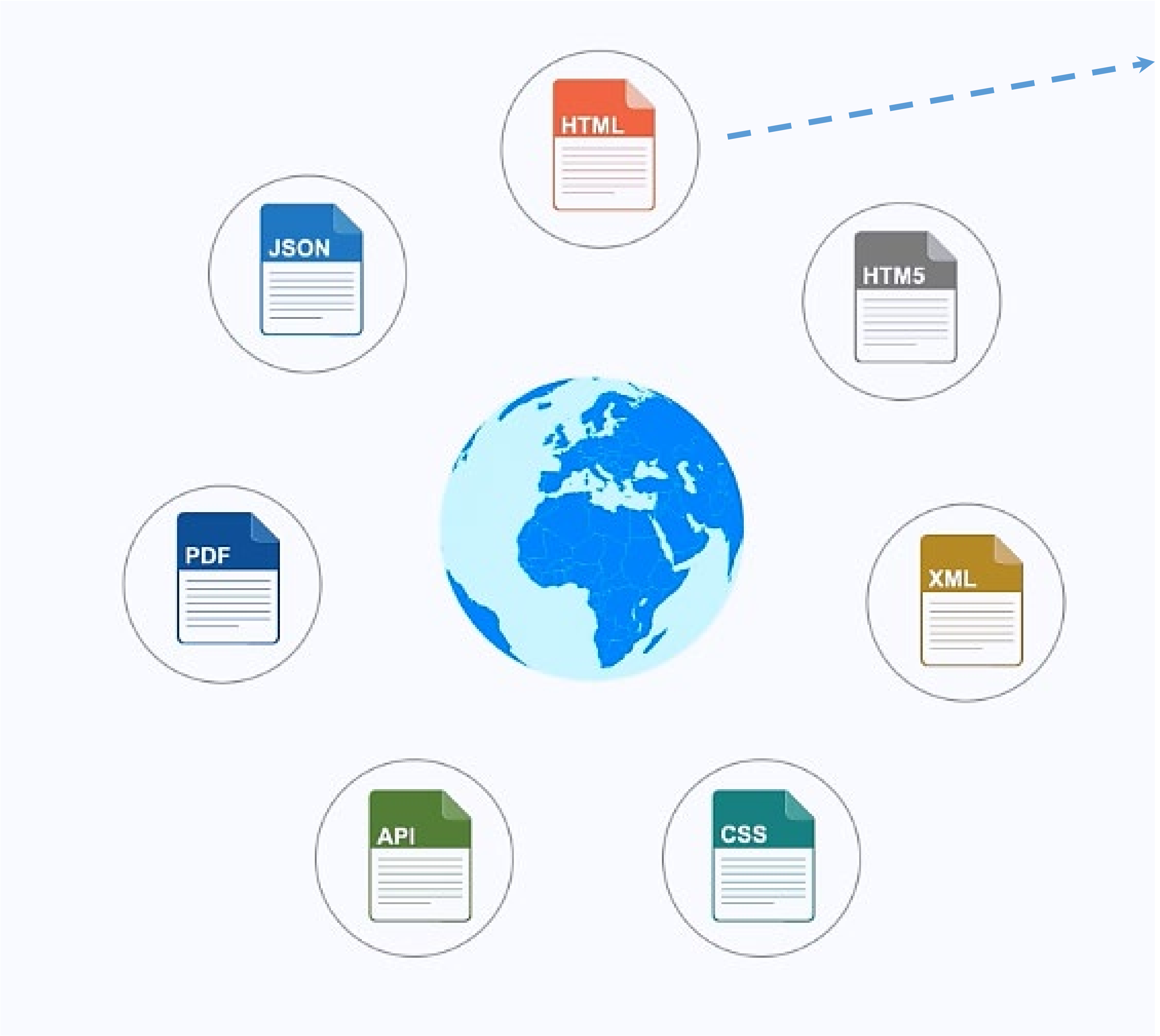
Features of BeautifulSoup

- Efficient tool for dissecting documents and extracting information from the web pages
- Has powerful sets of built-in methods for navigating, searching, and modifying a parse tree
- Contains a parser that supports both html and xml documents
- Converts all incoming documents to unicode automatically
- Converts all outgoing documents to UTF-8 automatically

Common Data/Page Formats on the Web

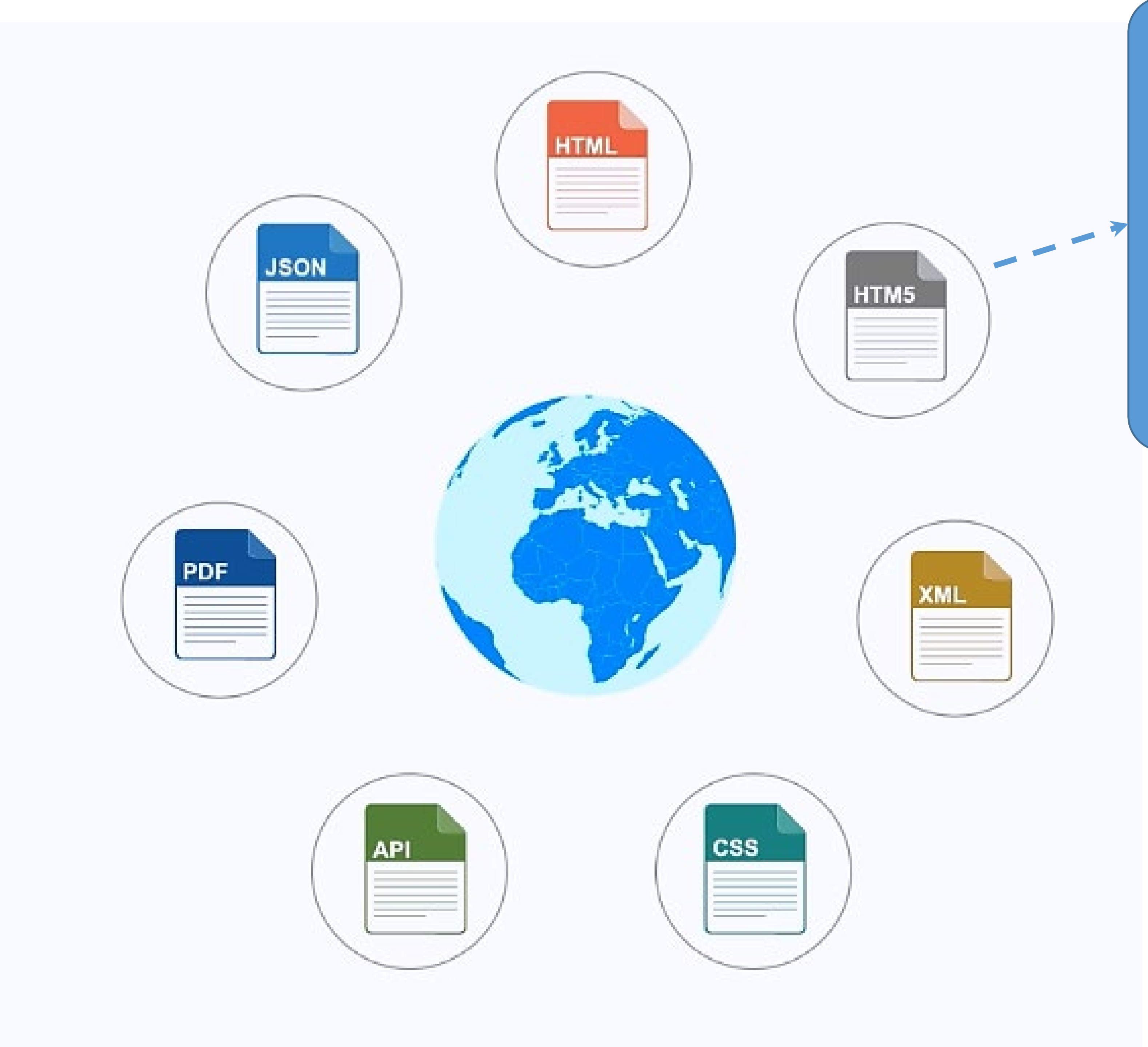


Common Data/Page Formats on the Web



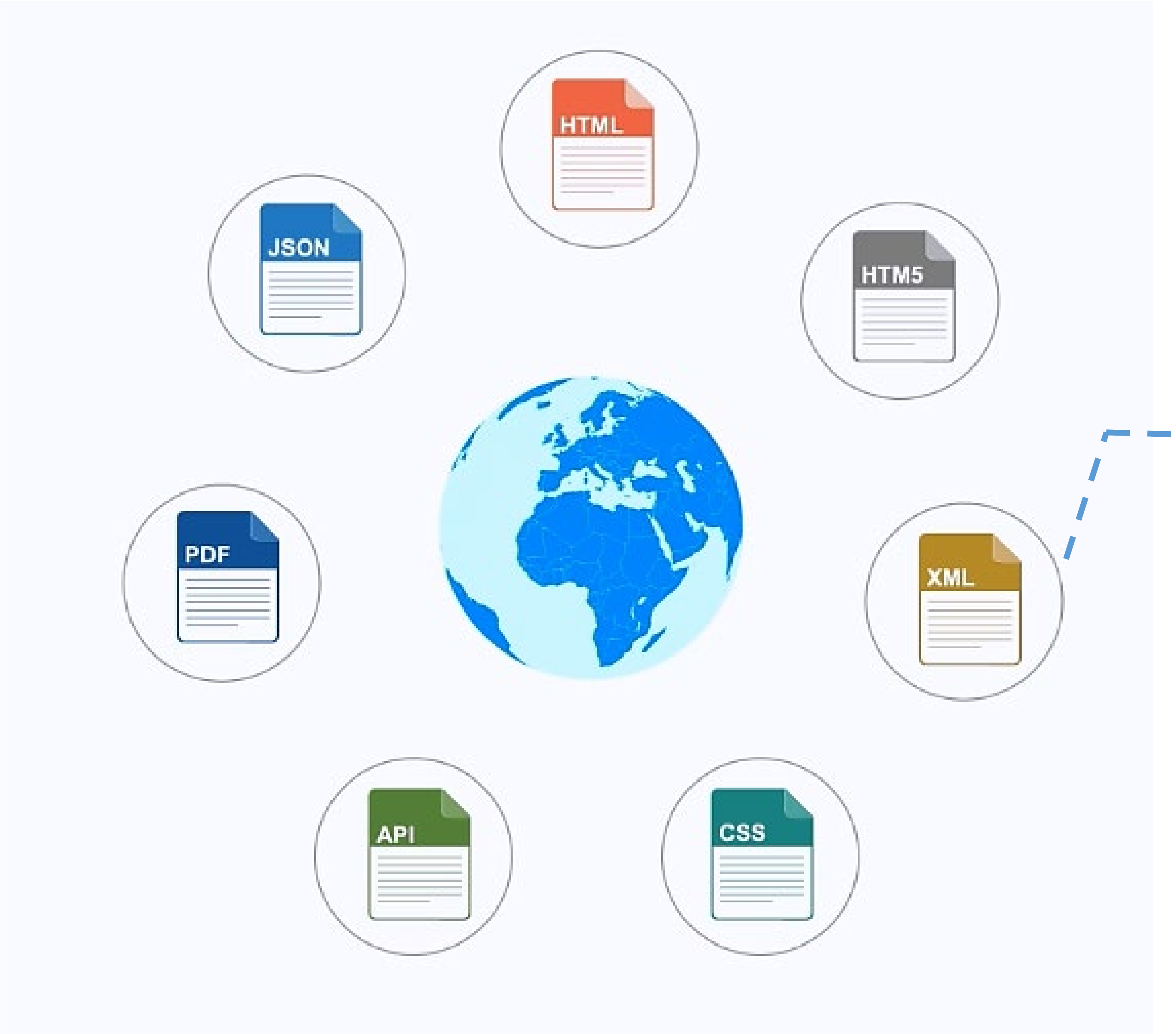
An HTML page is one of the oldest, easiest, and the most popular methods to upload information on the web.

Common Data/Page Formats on the Web



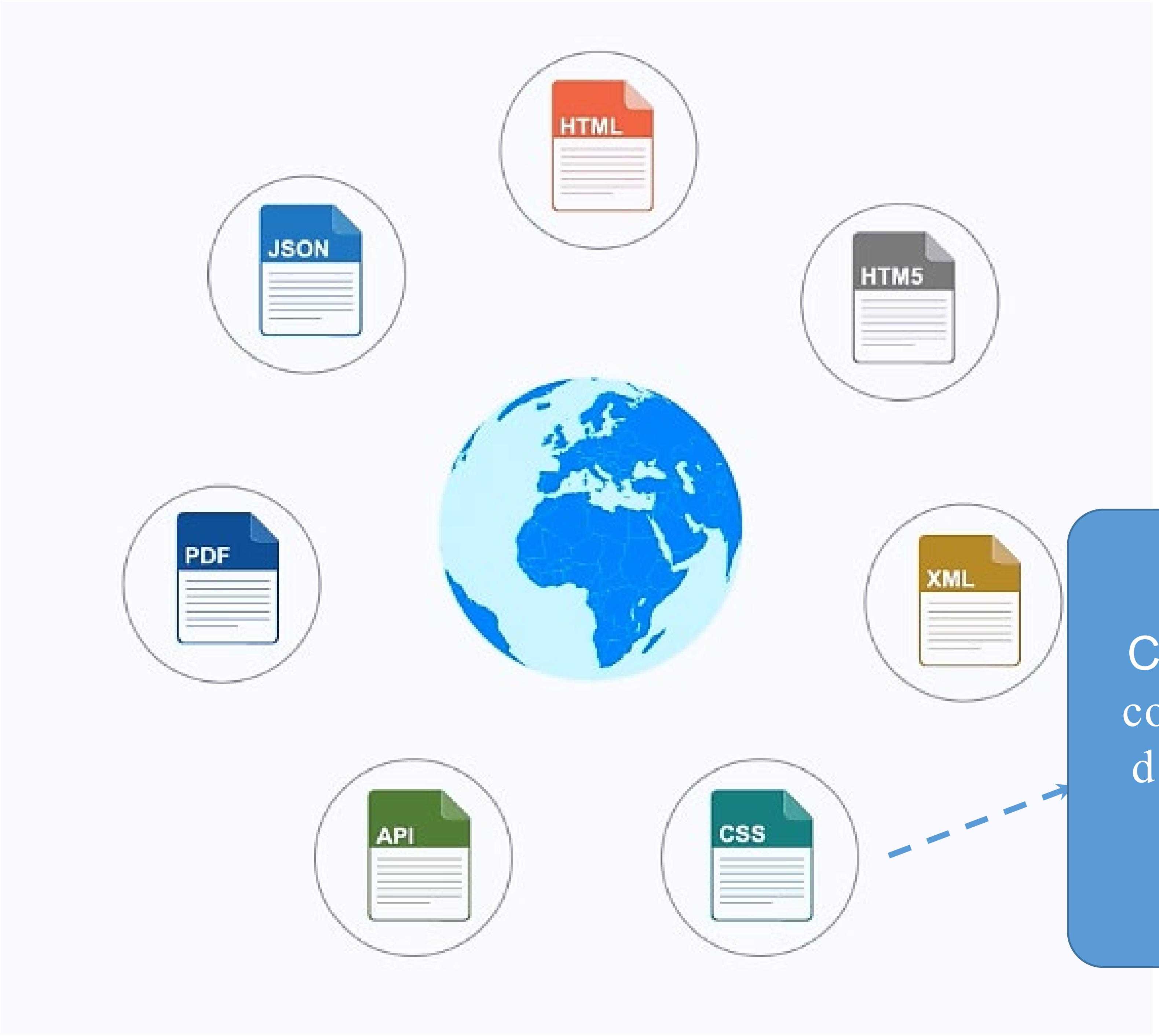
An HTML 5 is a new HTML standard which gained popularity with the mobile devices.

Common Data/Page Formats on the Web



XML is another popular way to upload your information on the web.

Common Data/Page Formats on the Web



CSS is mainly used for the consistent presentation of data using cascaded style sheets.

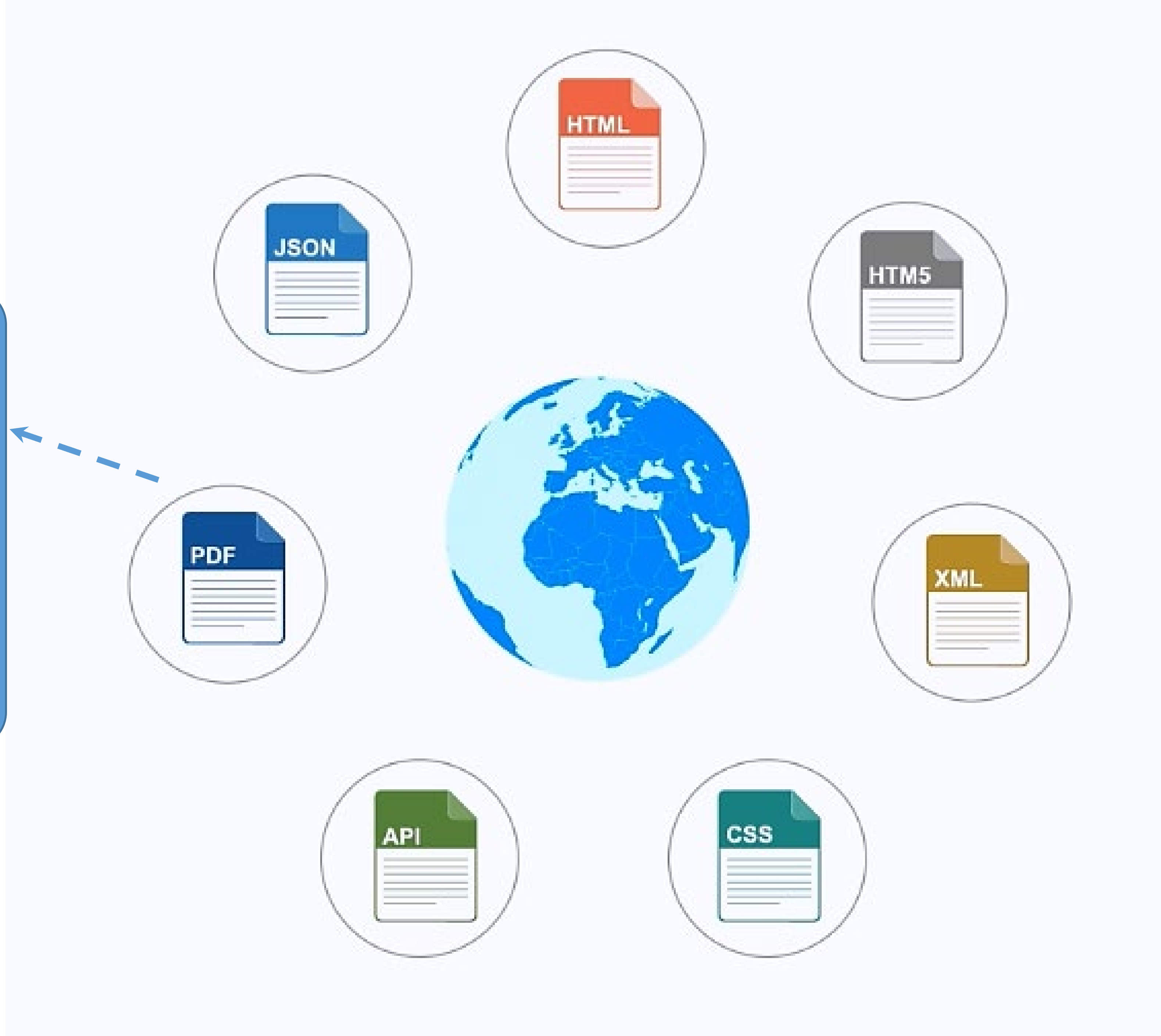
Common Data/Page Formats on the Web

HTML
JSON
HTM5
PDF
XML
API
CSS

Application Program Interface or APIs have now become a common practice to extract information from the web.

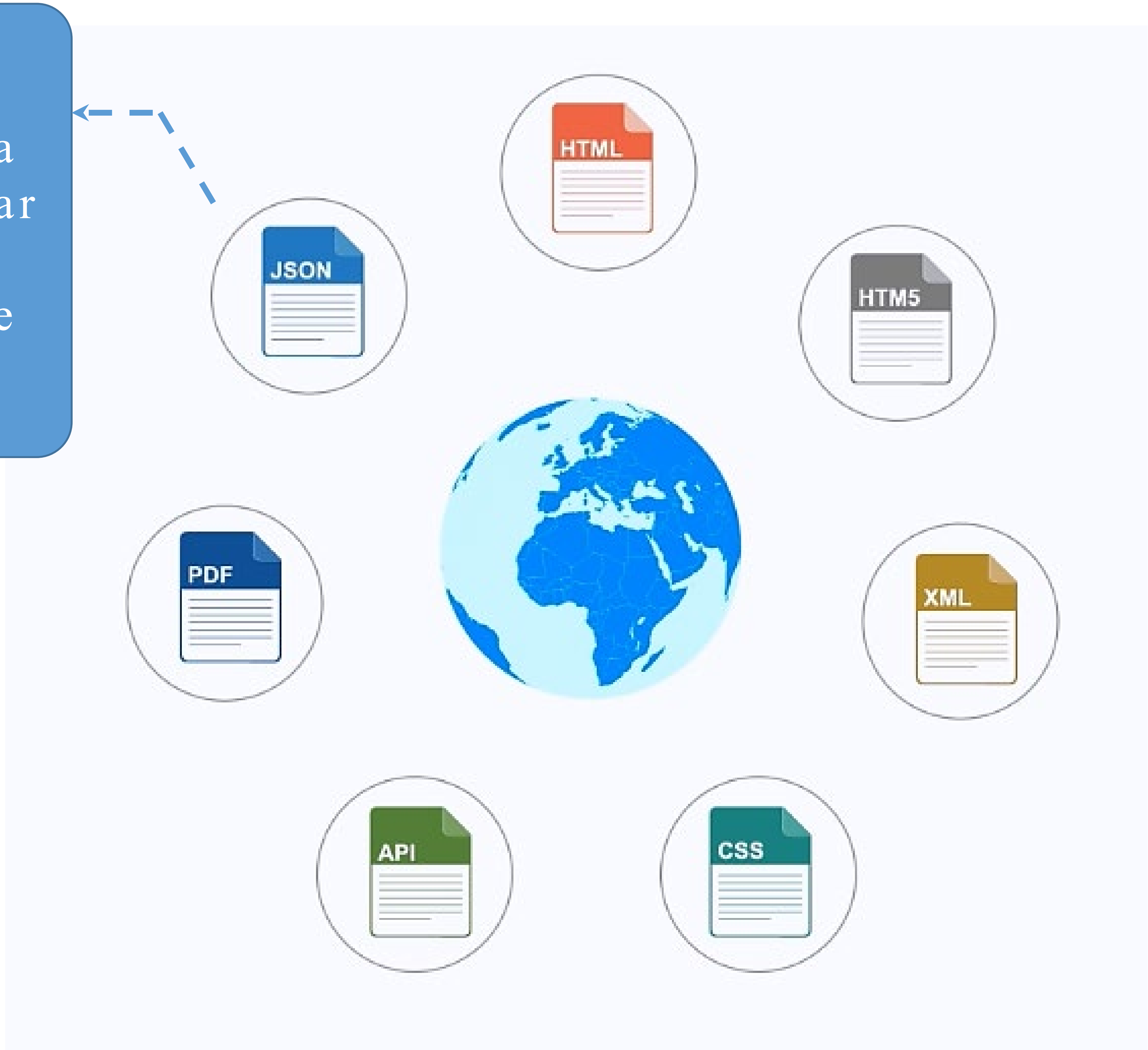
Common Data/Page Formats on the Web

PDF is also widely used to upload information and reports.



Common Data/Page Formats on the Web

JavaScript Object Notation, or JSON, is a lightweight and popular format used for information exchange on the web.



Parser

Parser

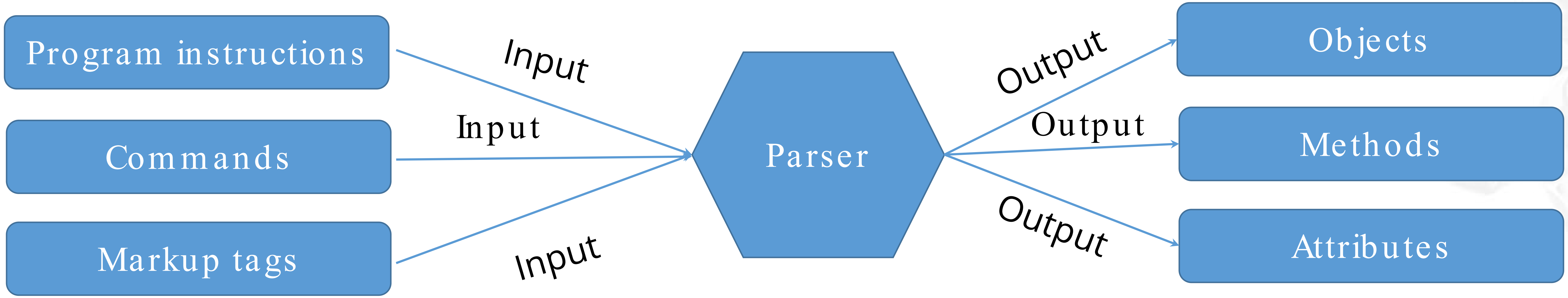
What is a parser?

How does it help Data Scientists in the web scraping process?

Parser

A Parser is a basic tool to interpret or render information from a web document.

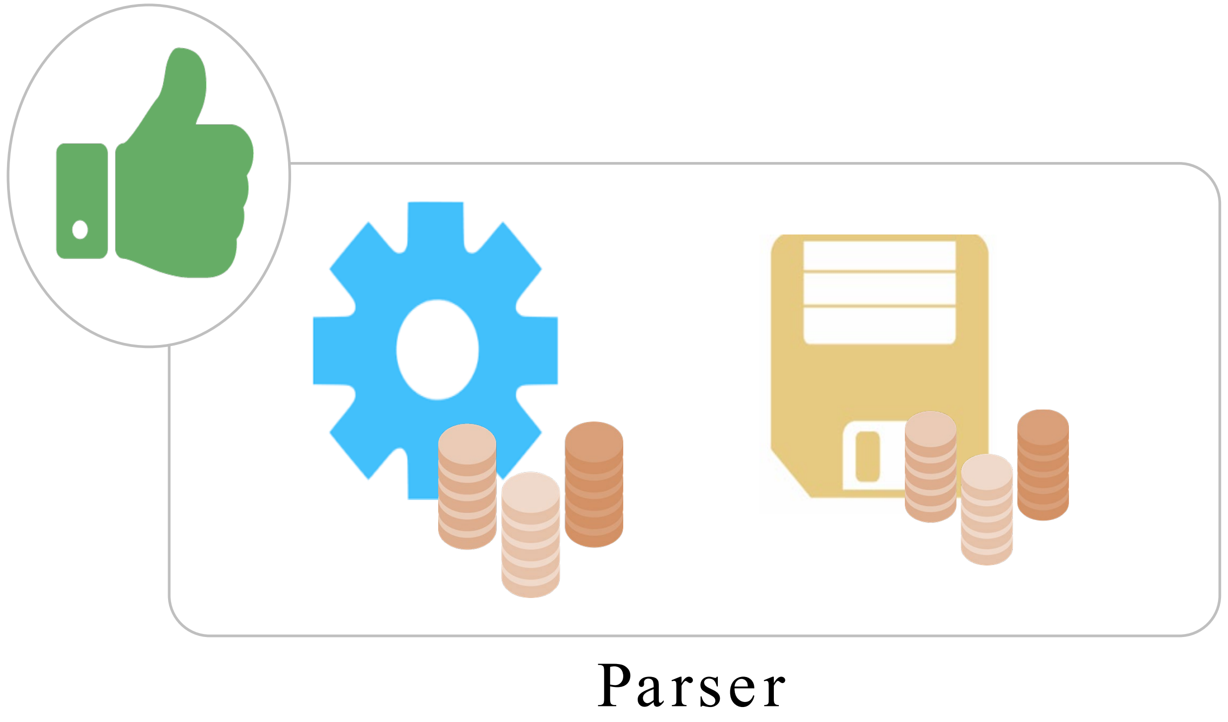
A Parser is also used to validate the input information before processing it.

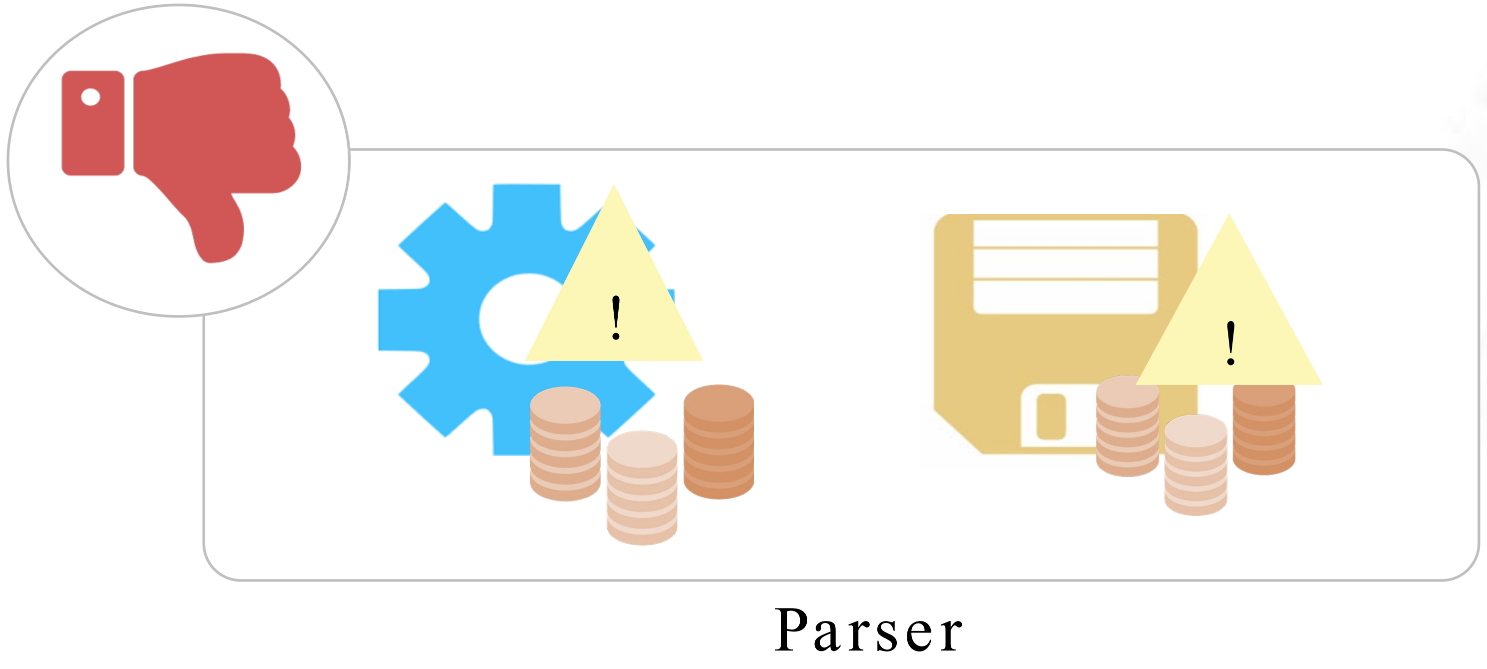


Importance of Parsing

Parsing data is one of the most important steps in the web scraping process.

Failing to parse the data would eventually lead to a failure of the entire process.





Various Parsers

Various parsers supported by BeautifulSoup are:

html.parser

HTML parser is Python-based, fast, and lenient.

lxml html

Lxml html is not built using Python and it depends on C. However, it is fast and lenient in nature.

lxml xml

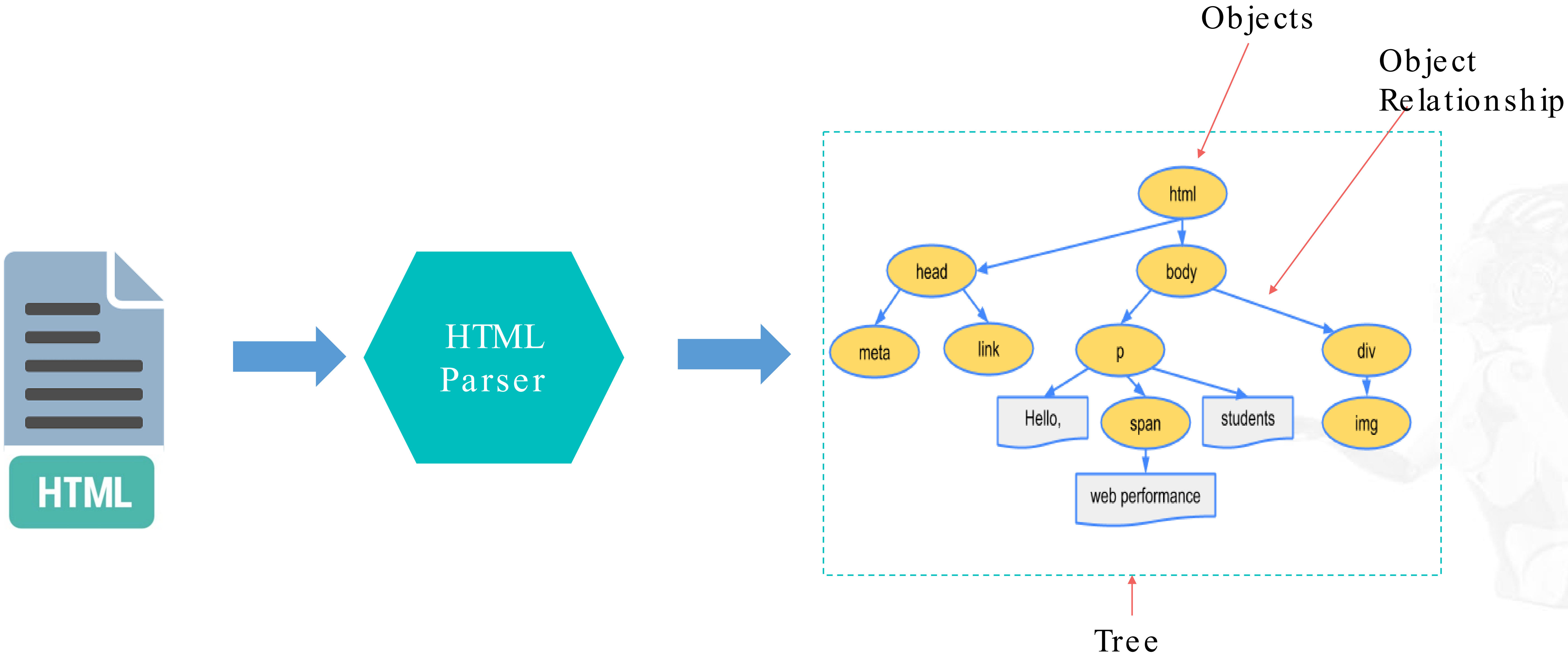
Lxml xml is the only xml parser available and it also depends on C.

html5lib

HTML5lib is another Python-based parser; however, it is slow and can create valid HTML5.

Importance of Objects

A web document gets transformed into a complex tree of objects.



A tree is defined as a collection of simple and complex objects.

Types of Objects

BeautifulSoup transforms a complex HTML document into a complex tree of Python objects. There are four types of objects. They are:

Tag

A tag object is an XML or HTML tag in the web document. Tags have a lot of attributes and methods.

NavigableString

A NavigableString is a string or set of characters that correspond to the text present within a tag.

BeautifulSoup

A BeautifulSoup represents the entire web document and supports navigating and searching the document tree.

Comment

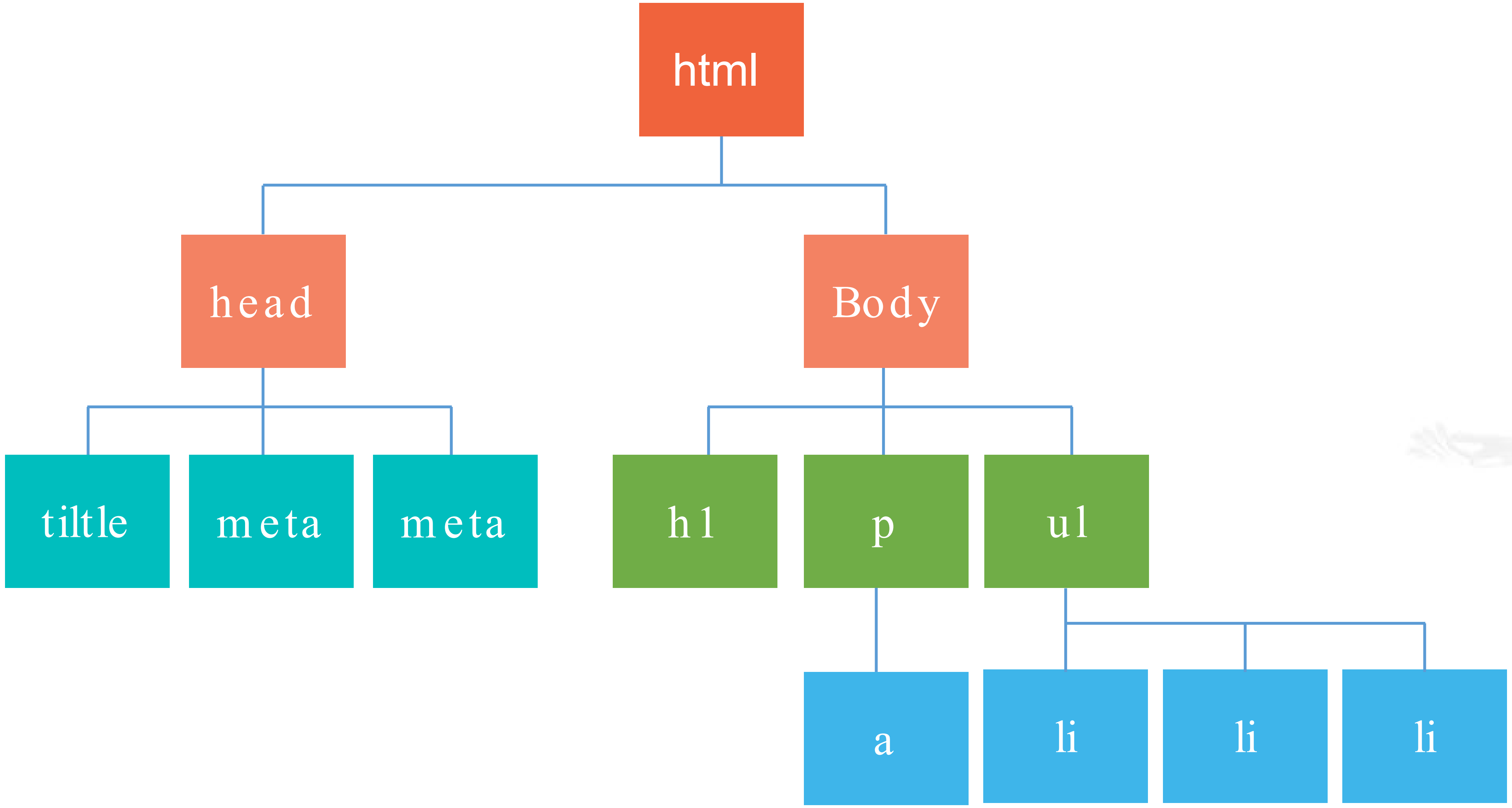
A Comment represents the comment or information section of the document. It is a special type of NavigableString.

Parsing Web Documents and Extracting Data Using Objects

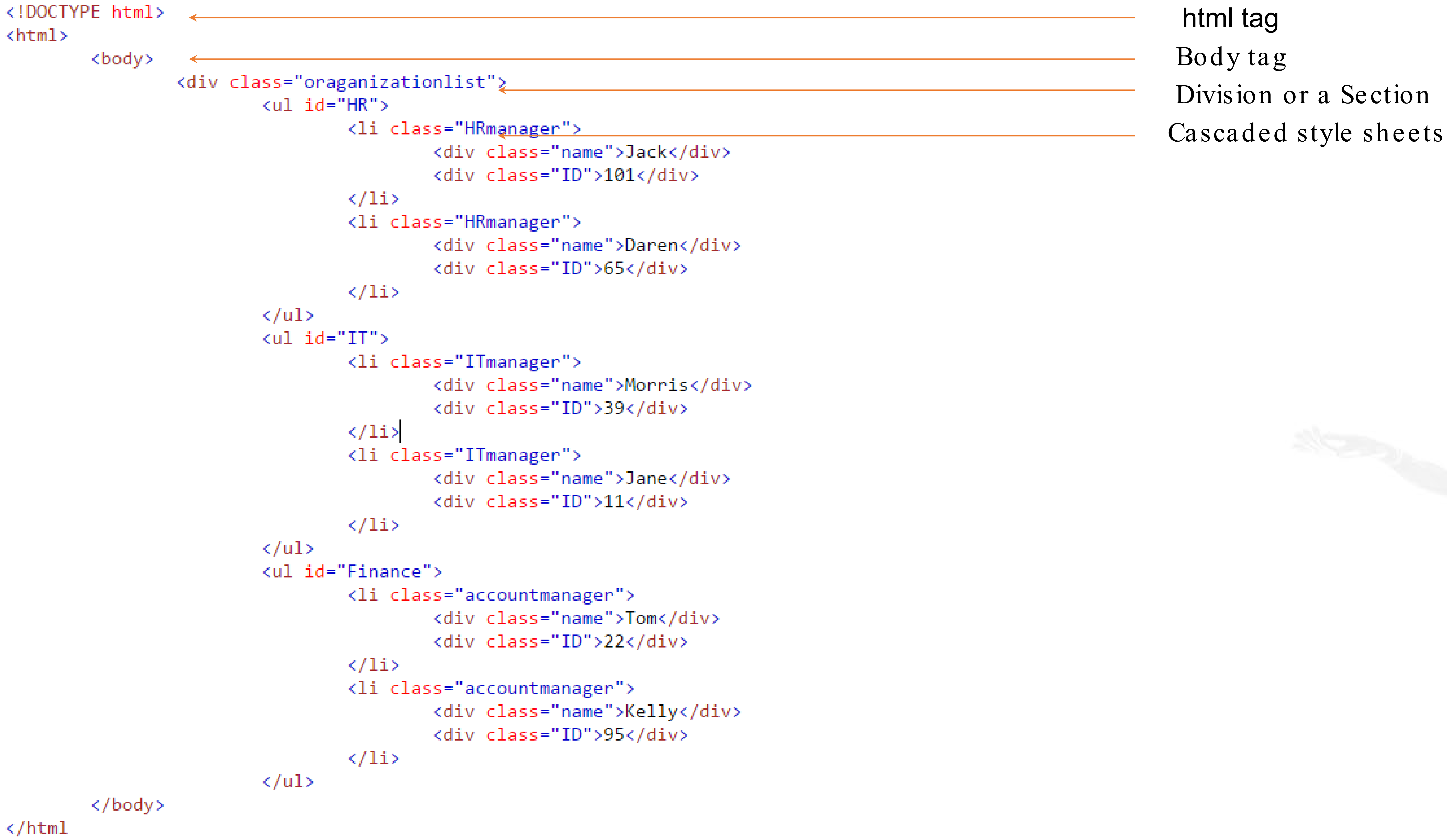
Demonstrate how to scrape a web document, parse it, and use objects to extract information.

ASSISTED PRACTICE

Understanding Tree



Understanding Tree



```
<!DOCTYPE html>
<html>
        <body>
                <div class="oraganizationlist">
                        <ul id="HR">
                                <li class="HRmanager">
                                        <div class="name">Jack</div>
                                        <div class="ID">101</div>
                                </li>
                                <li class="HRmanager">
                                        <div class="name">Daren</div>
                                        <div class="ID">65</div>
                                </li>
                        </ul>
                        <ul id="IT">
                                <li class="ITmanager">
                                        <div class="name">Morris</div>
                                        <div class="ID">39</div>
                                </li>
                                <li class="ITmanager">
                                        <div class="name">Jane</div>
                                        <div class="ID">11</div>
                                </li>
                        </ul>
                        <ul id="Finance">
                                <li class="accountmanager">
                                        <div class="name">Tom</div>
                                        <div class="ID">22</div>
                                </li>
                                <li class="accountmanager">
                                        <div class="name">Kelly</div>
                                        <div class="ID">95</div>
                                </li>
                        </ul>
        </body>
</html
```

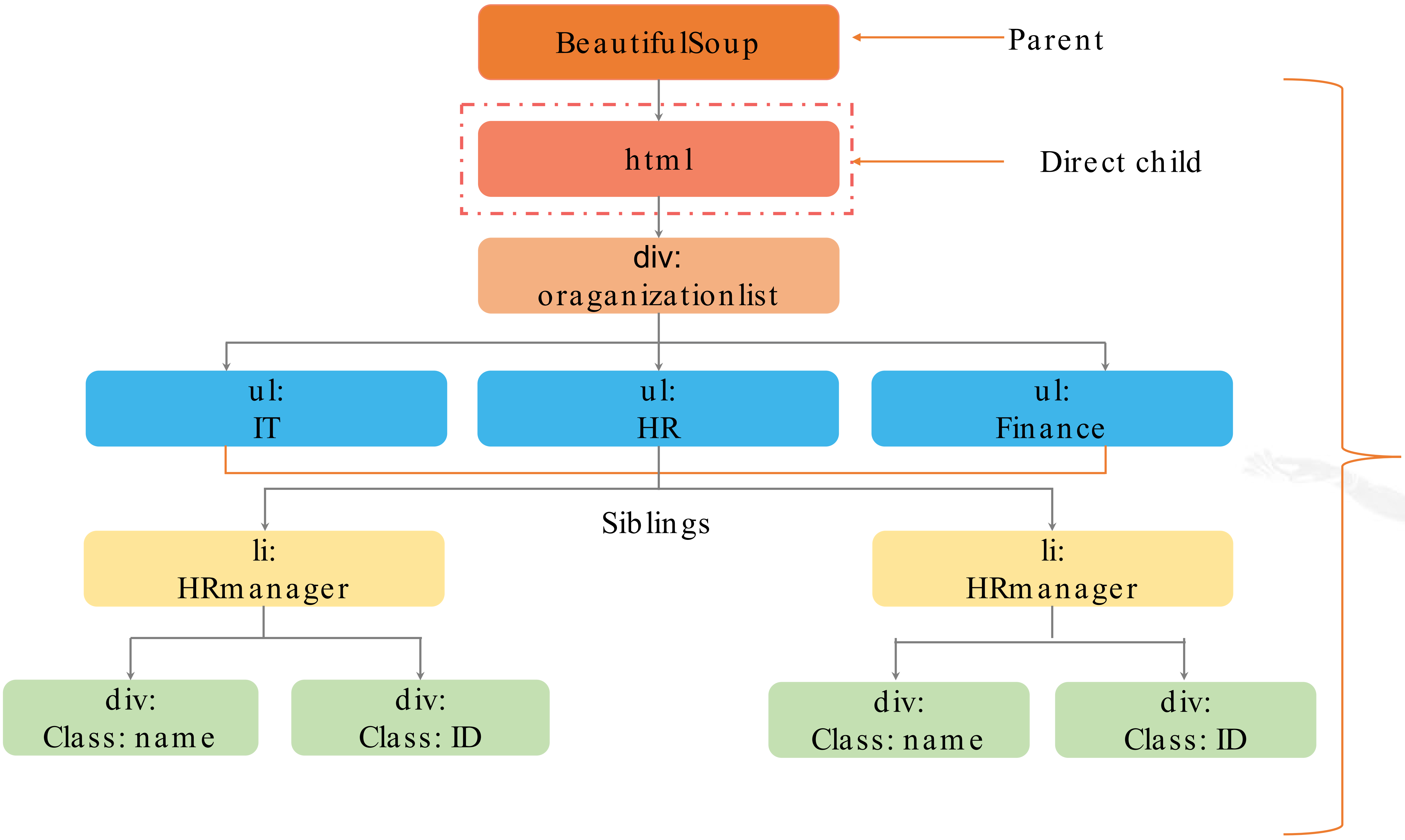
html tag

Body tag

Division or a Section

Cascaded style sheets

Understanding Tree

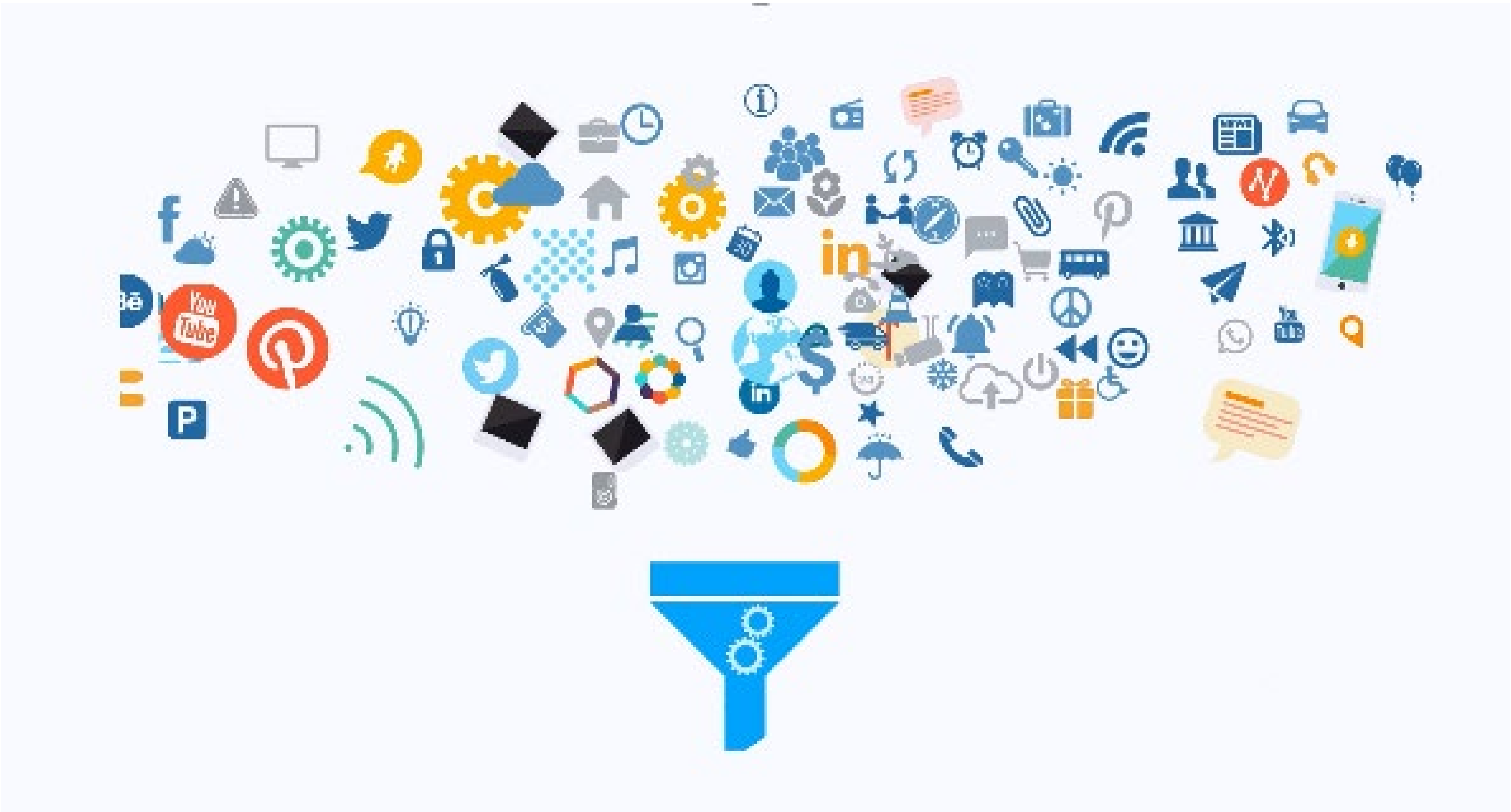


Various Operations

Searching Tree: Filters

With the help of the search filters technique, you can extract specific information from the parsed document.

The filters can be treated as search criteria for extracting the information based on the elements present in the document.



Searching Tree: Filters

There are various kinds of filters used for searching information from a tree.

String

A string is the simplest filter. BeautifulSoup will perform a match against the search string.

Regular Expressions

A regular expression filters the match against the search criteria.

List

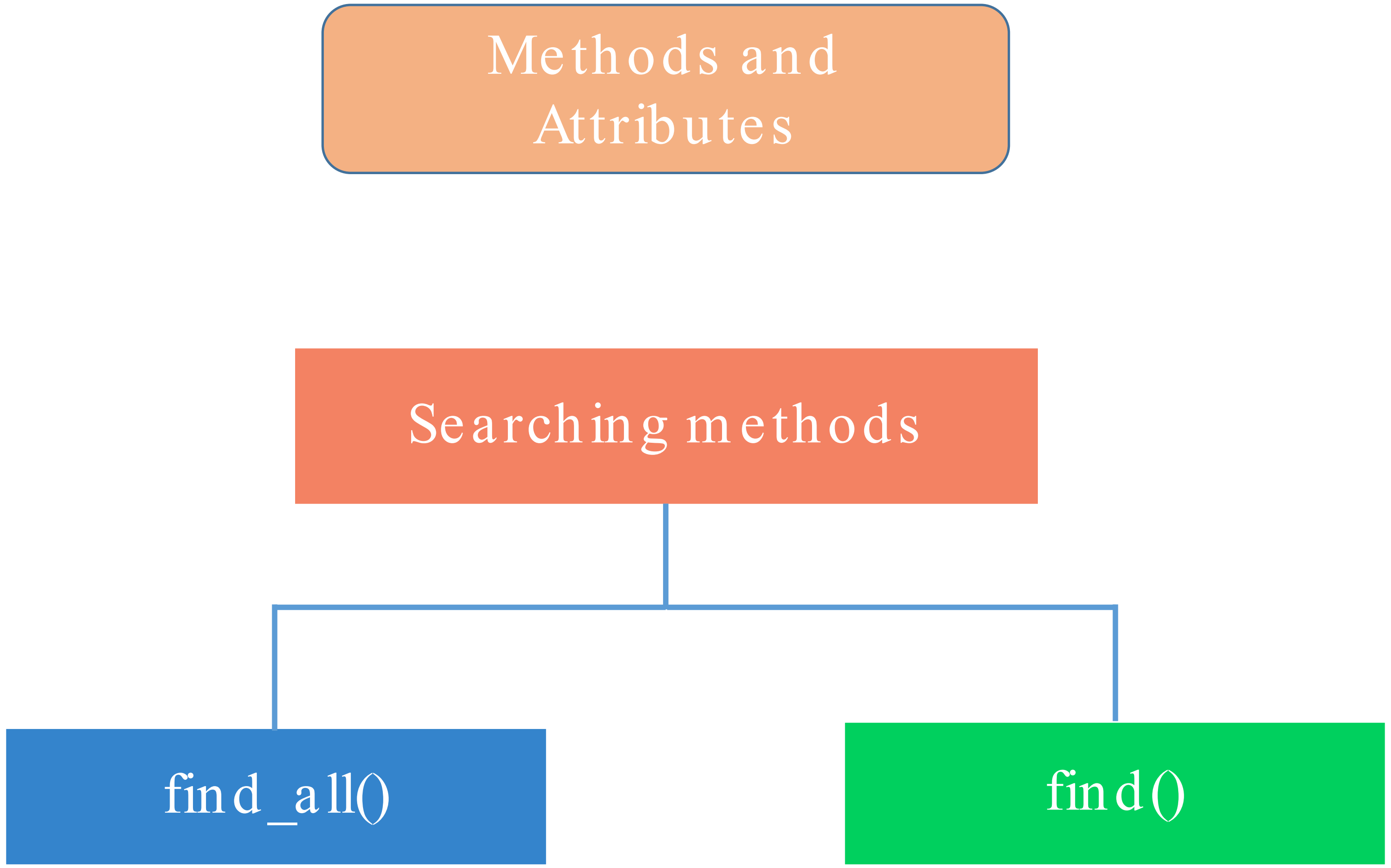
A list filters the string that matches against the search item in the list.

Function

A function filters the elements that match against its only argument.

Searching the Tree: find_all()

BeautifulSoup defines a lot of methods for searching the parsed tree.



Methods and Attributes

Searching methods

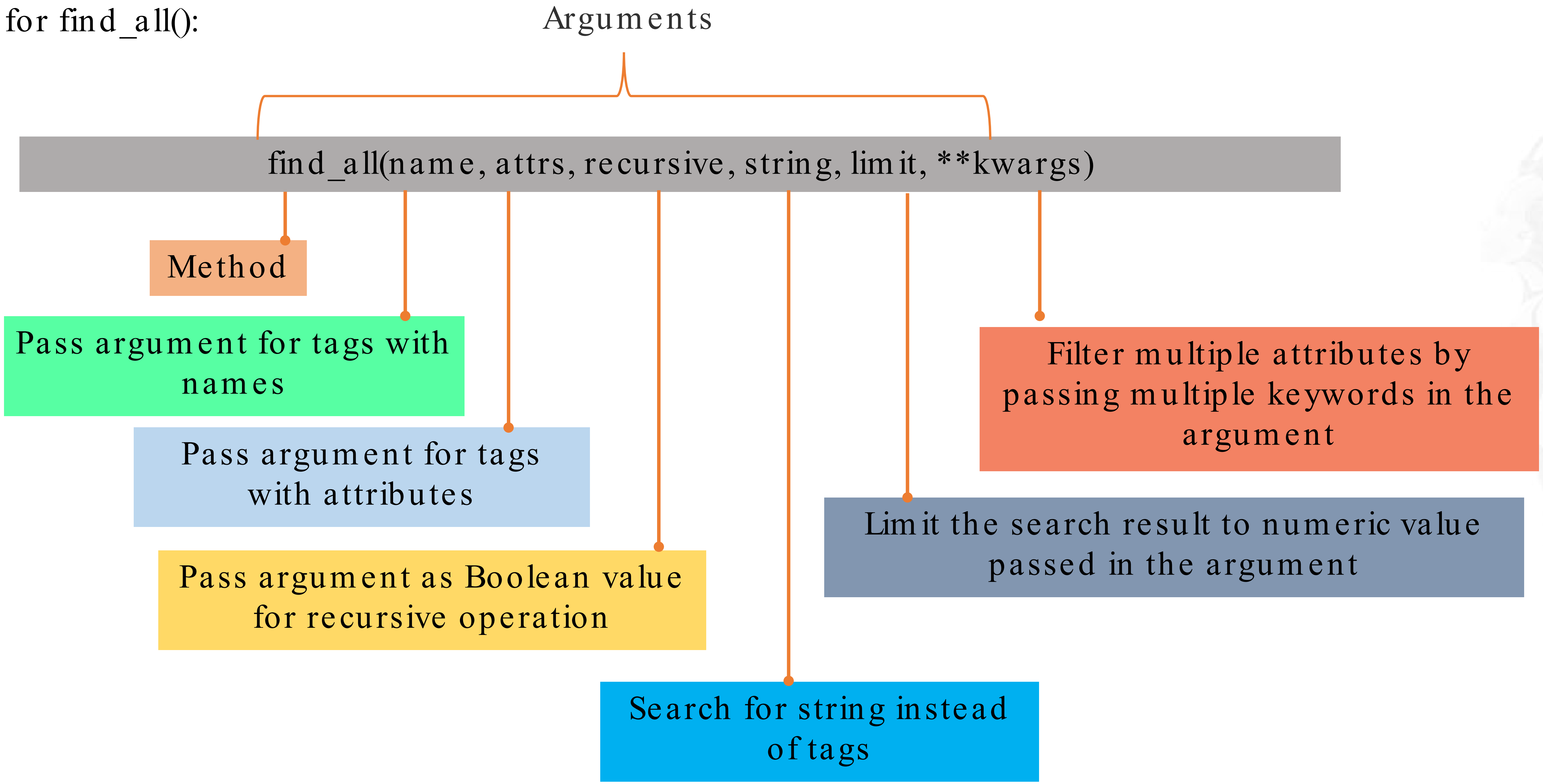
find_all()

find()

Searching the tree with find_all()

The find_all() searches and retrieves all tags’ descendants that matche your filters.

The syntax for find_all():



Arguments

find_all(name, attrs, recursive, string, limit, **kwargs)

Method

Pass argument for tags with names

Pass argument for tags with attributes

Pass argument as Boolean value for recursive operation

Search for string instead of tags

Limit the search result to numeric value passed in the argument

Filter multiple attributes by passing multiple keywords in the argument

Searching the tree with find ()

The find_all() finds the entire document looking for results.

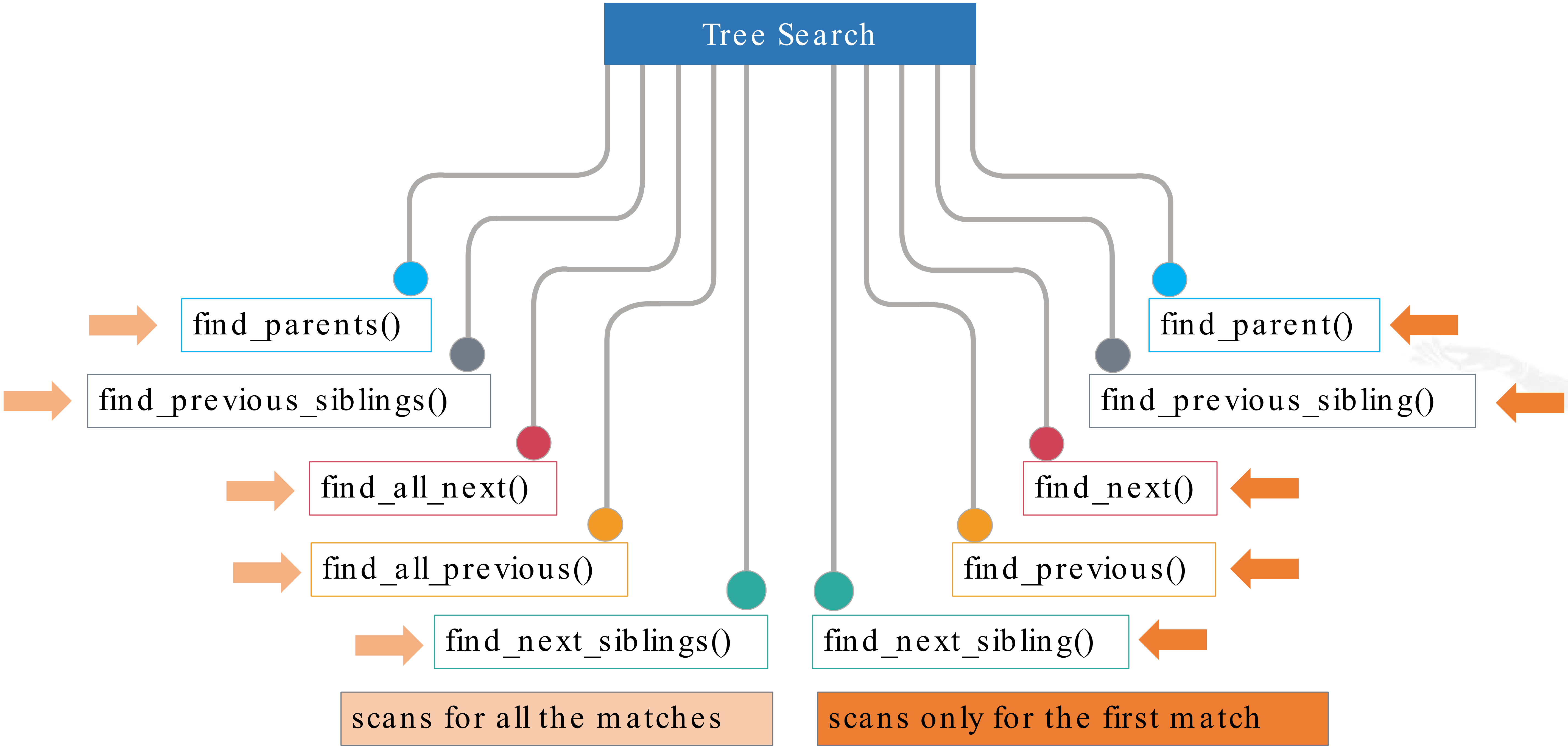
To find one result, use find().

The find() method has a syntax similar to that of the find_all() method; however, there are some key differences.

Method Name	Search Scope	Match Found	Match Not Found
Find_all()	Scans entire document	Returns list with values	Returns empty list
Find()	Searches only for passed argument	Returns only the first match value	Returns Nothing

Searching the Tree with Other Methods

Searching the parse tree can also be performed by various other methods such as:



Tree Search

- find_parents()
- find_previous_siblings()
- find_all_next()
- find_all_previous()
- find_next_siblings()

- find_parent()
- find_previous_sibling()
- find_next()
- find_previous()
- find_next_sibling()

scans for all the matches

scans only for the first match

Searching in a Tree with Filters

Demonstrate the ways to search in a tree using filters.

Navigating Options

With the help of BeautifulSoup, it is easy to navigate the parse tree based on the need.

There are four options to navigate the tree. They are:

- Navigating Down
- Navigating Up
- Navigating Sideways
- Navigating Back and Forth

Navigating Options

There are four options to navigate the tree. They are:

Navigating Down

Navigating Up

Navigating Sideways

Navigating Back and Forth

This technique shows you how to extract information from children tags. Following are the attributes used to navigate down:

- .contents and .children
- .descendants
- .string
- .strings and stripped_strings

Navigating Options

There are four options to navigate the tree:

- Navigating Down
- Navigating Up
- Navigating Sideways
- Navigating Back and Forth

Every tag has a parent and two attributes, .parents and .parent, to help navigate up the family tree.

Navigating Options

There are four options to navigate the tree:

- Navigating Down
- Navigating Up
- Navigating Sideways
- Navigating Back and Forth

This technique shows you how to extract information from the same level in the tree. The attributes used to navigate sideways are: .next_sibling and .previous_sibling.

Navigating Options

There are four options to navigate the tree:

- Navigating Down
- Navigating Up
- Navigating Sideways
- Navigating Back and Forth

This technique shows you how to parse the tree back and forth.
The attributes used to navigate back and forth are:
.next_element and .previous_element
.next_elements and .previous_elements

Navigating a Tree

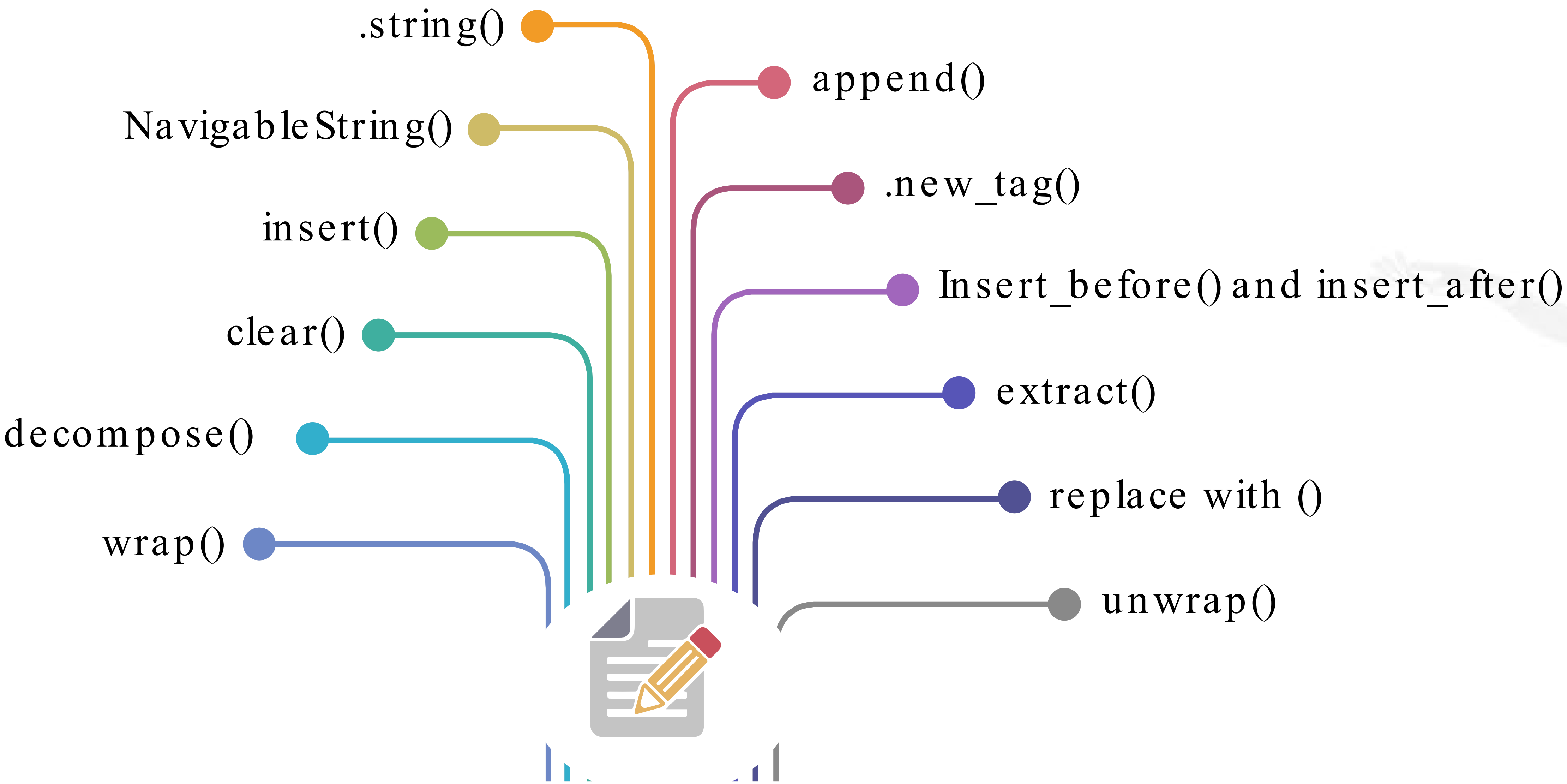
Demonstrate how to navigate the web tree using various techniques.

ASSISTED PRACTICE

Modifying the Tree

With BeautifulSoup, you can also modify the tree and write your changes as a new HTML or XML document.

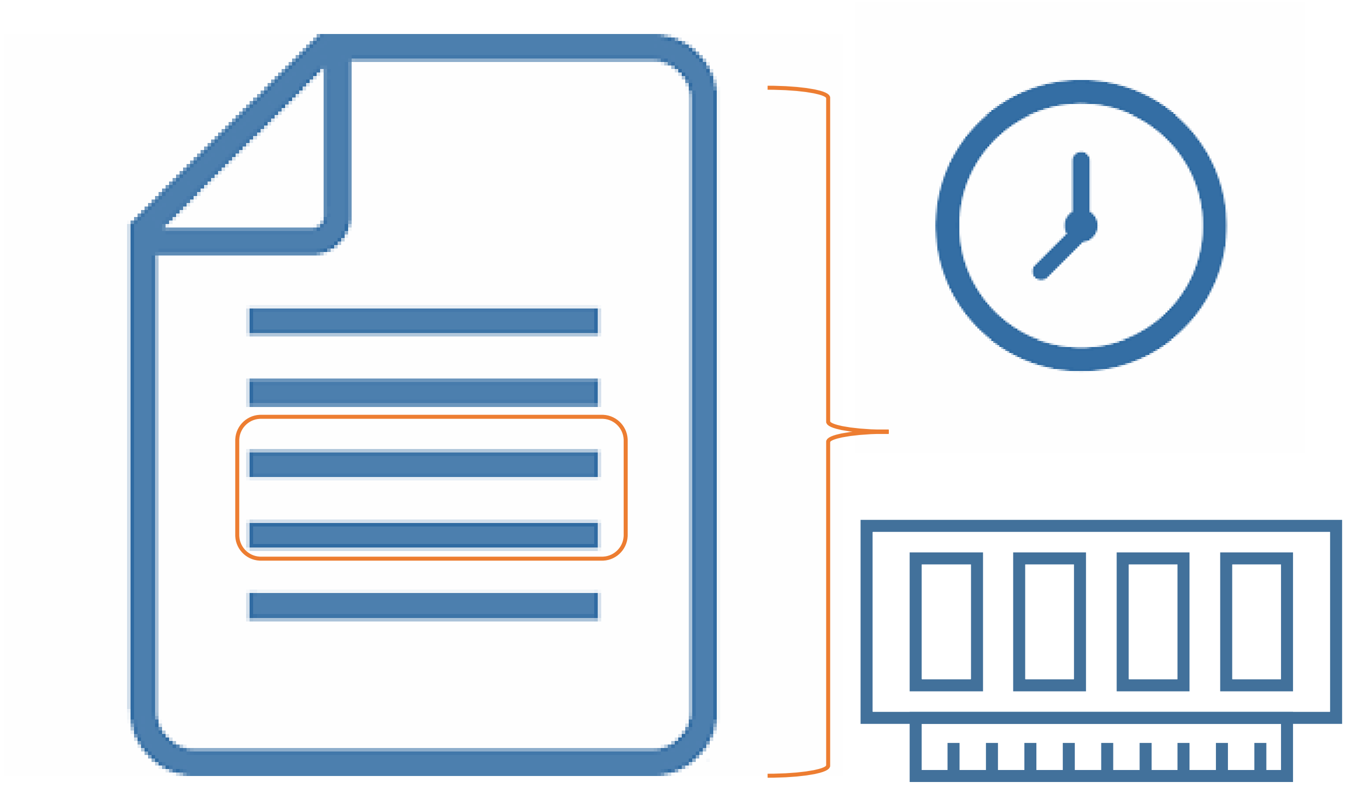
There are several methods to modify the tree:



Modifying the Tree

Demonstrate how to modify a web tree to get the desired result with the help of an example.

Parsing Only Part of the Document



But, how can you overcome this problem?

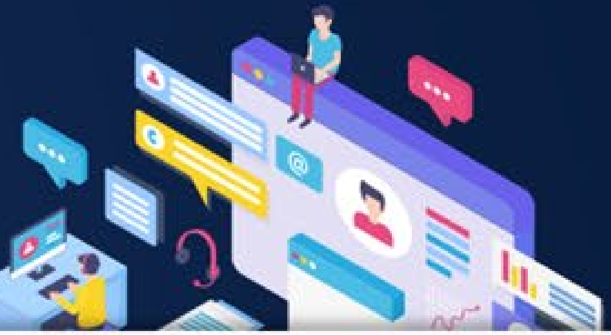
Use SoupStrainer class

↓

Allows you to choose the part of the document to be parsed

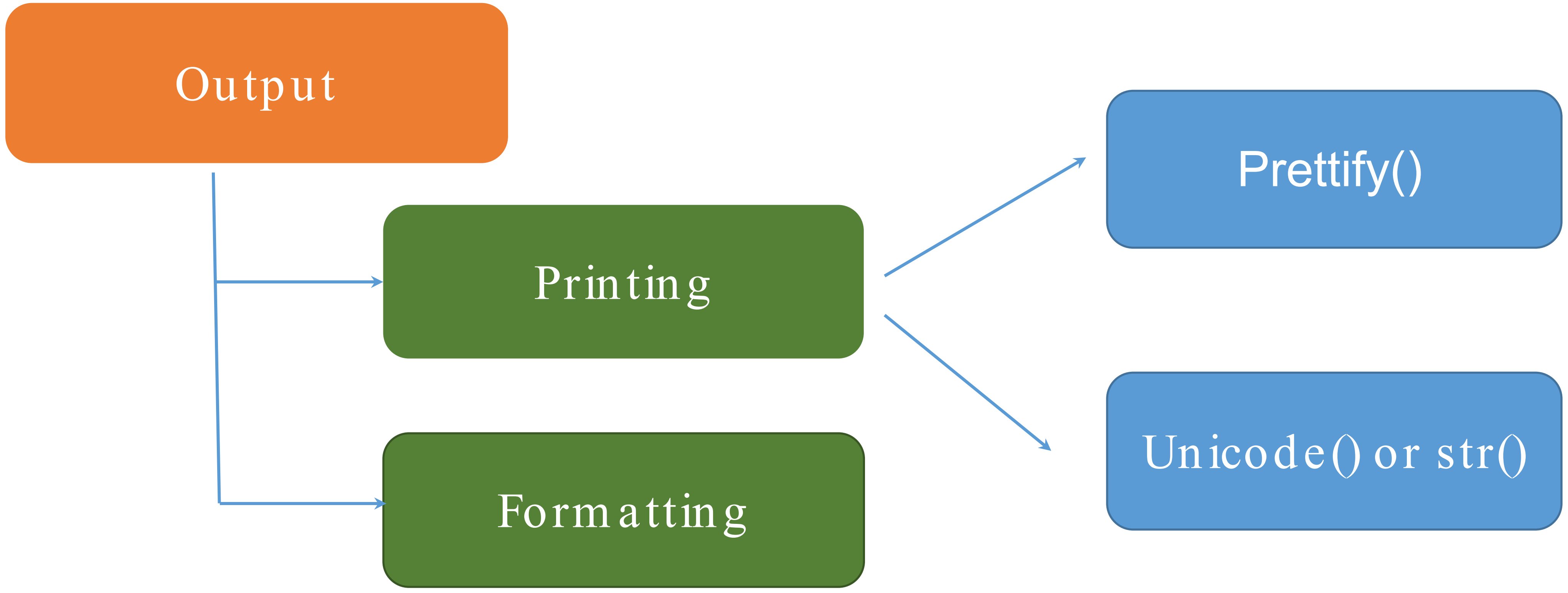
This feature of parsing a part of the document will not work with the html5lib parser.

Parsing Part of the Document



Demonstrate how to parse only a part of document with the help of an example.

Output: Printing and Formatting



Output

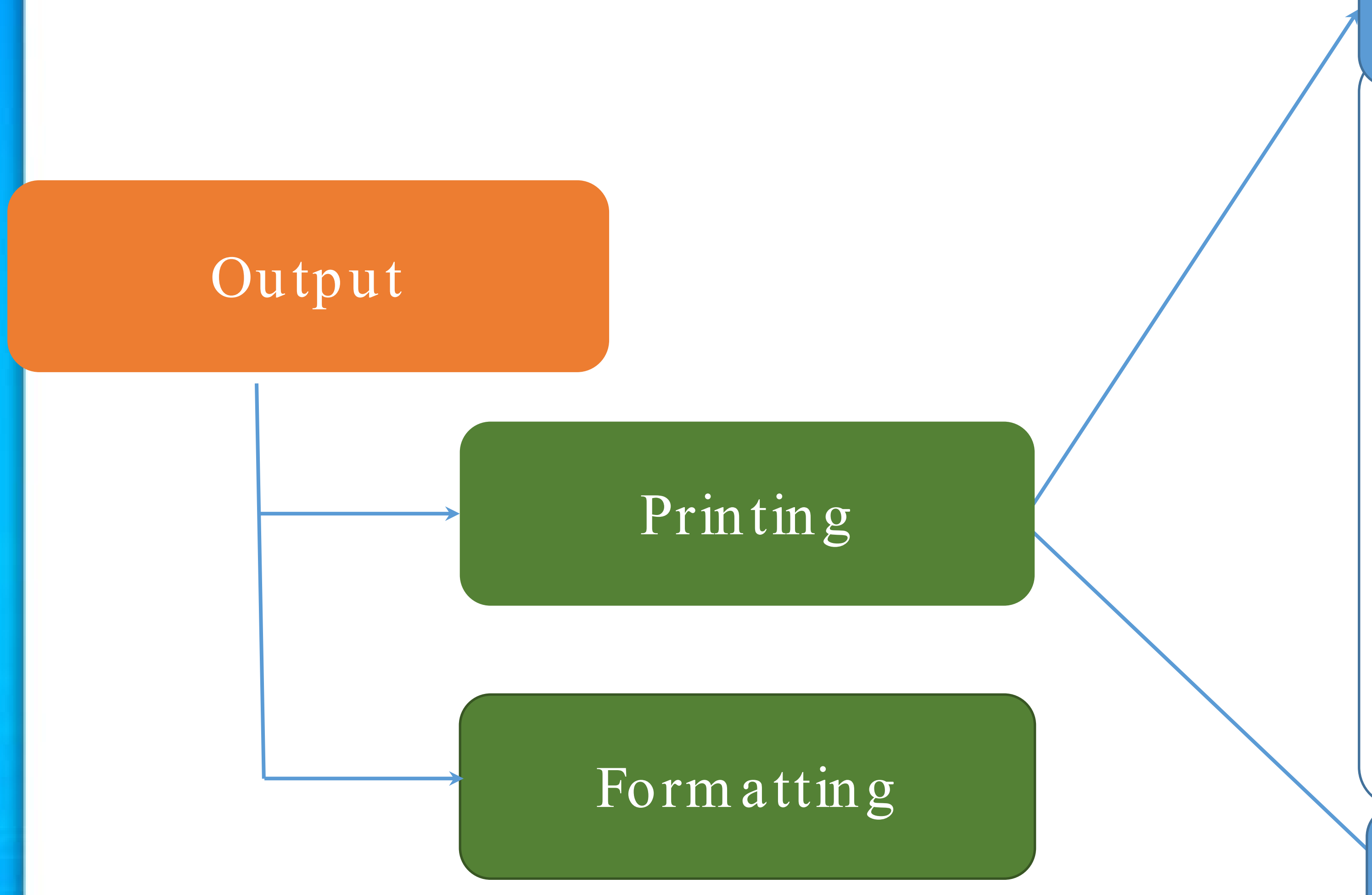
Printing

Formatting

Prettify()

Unicode() or str()

Output: Printing and Formatting



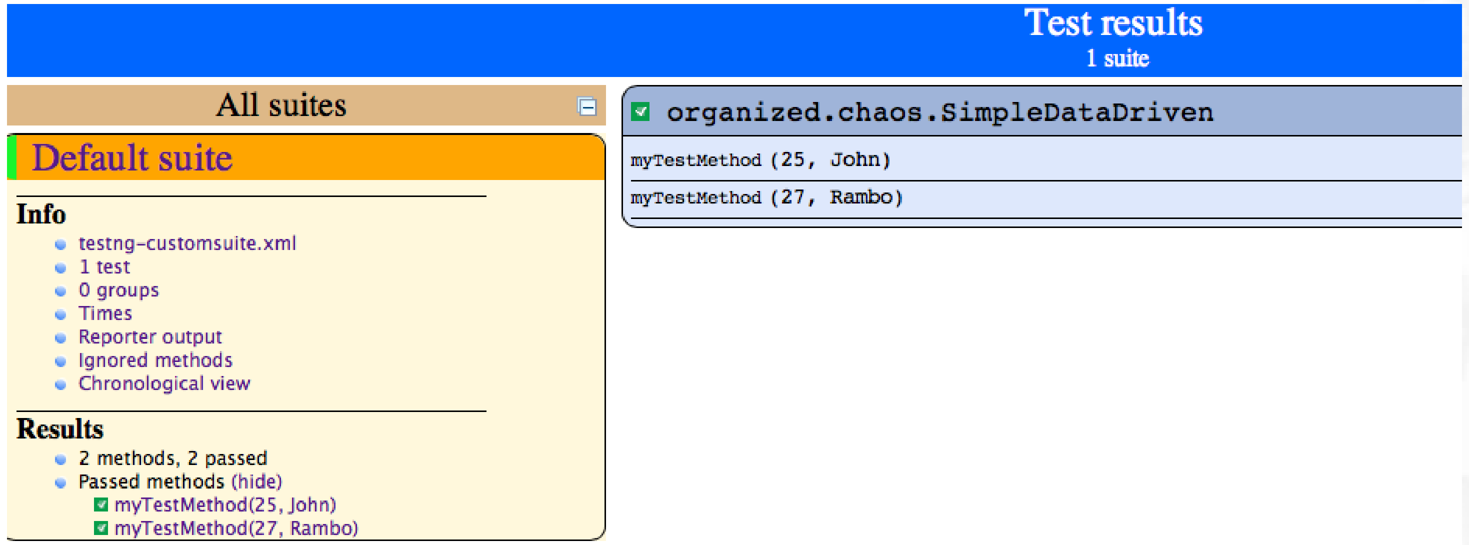
Output

Printing

Formatting

Prettify()

The prettify or pretty printing method turns a parse tree into a decorative formatted Unicode string.



Test results
1 suite

All suites

Default suite

Info
- testng-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

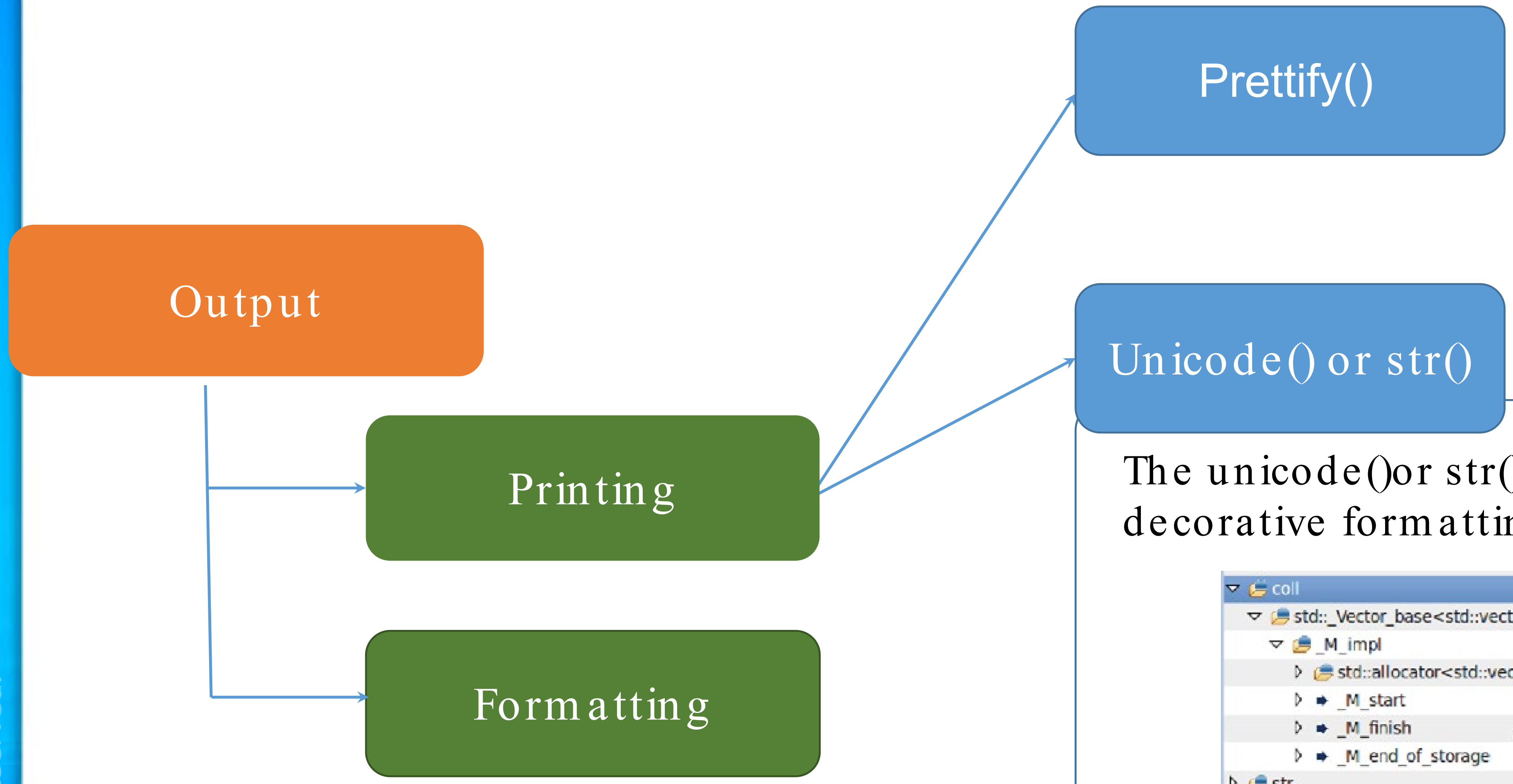
Results
- 2 methods, 2 passed
- Passed methods (hide)
  - ☑ myTestMethod(25, John)
  - ☑ myTestMethod(27, Rambo)

☑ organized.chaos.SimpleDataDriven

```
myTestMethod (25, John)
myTestMethod (27, Rambo)
```

Unicode() or str()

Output: Printing and Formatting



Output

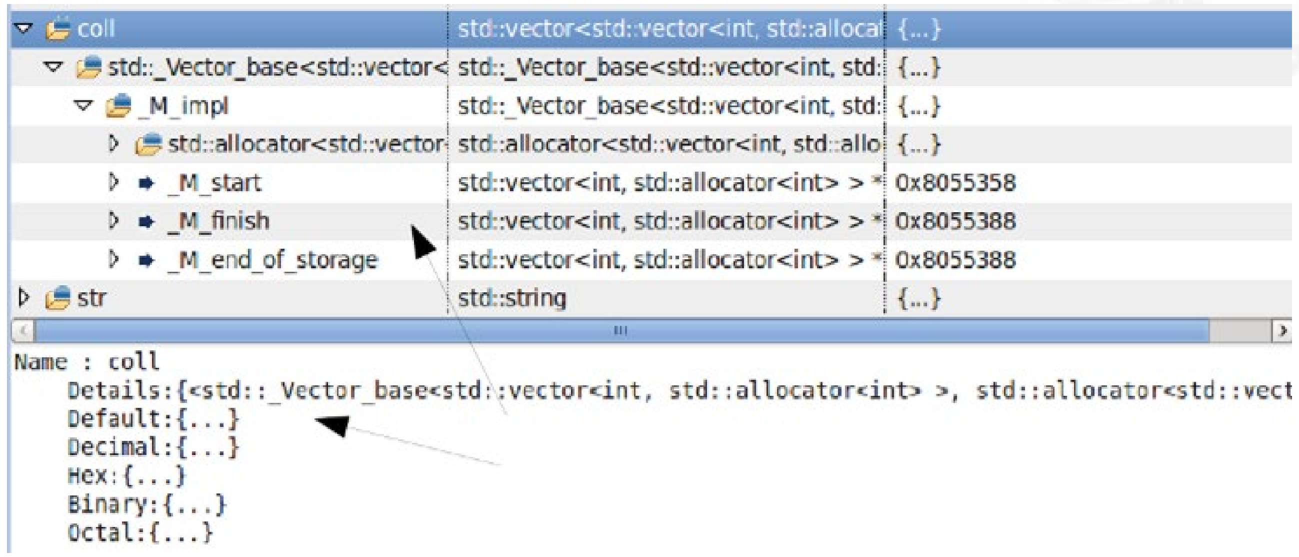
Printing

Formatting

Prettify()

Unicode() or str()

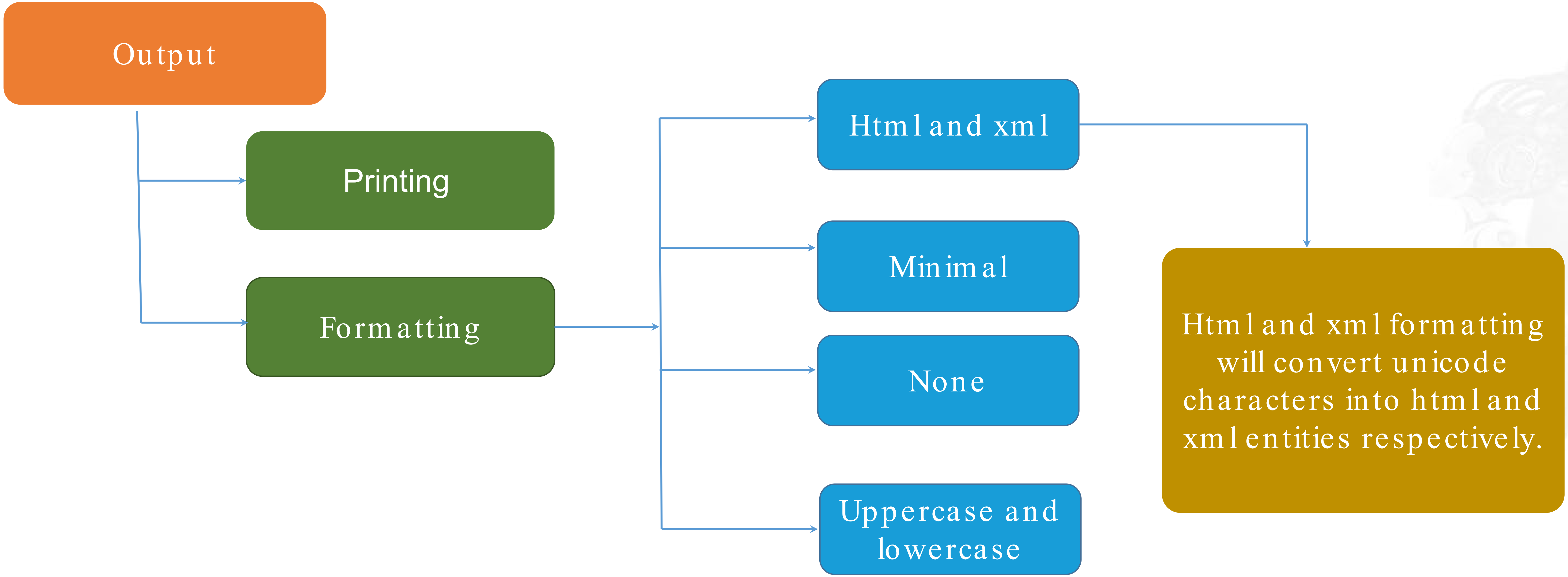
The unicode()or str() method turns a parse tree into a non-decorative formatting string.



```
▽ coll                                 std::vector<std::vector<int, std::allocat  {...}
  ▽ std::_Vector_base<std::vector<     std::_Vector_base<std::vector<int, std:    {...}
    ▽ _M_impl                          std::_Vector_base<std::vector<int, std:    {...}
      ▷ std::allocator<std::vector     std::allocator<std::vector<int, std::allo  {...}
      ▷ _M_start                       std::vector<int, std::allocator<int> > *   0x8055358
      ▷ _M_finish                      std::vector<int, std::allocator<int> > *   0x8055388
      ▷ _M_end_of_storage              std::vector<int, std::allocator<int> > *   0x8055388
▷ str                                  std::string                                {...}
Name : coll
    Details:{<std::_Vector_base<std::vector<int, std::allocator<int> >, std::allocator<std::vect
    Default:{...}
    Decimal:{...}
    Hex:{...}
    Binary:{...}
    Octal:{...}
```

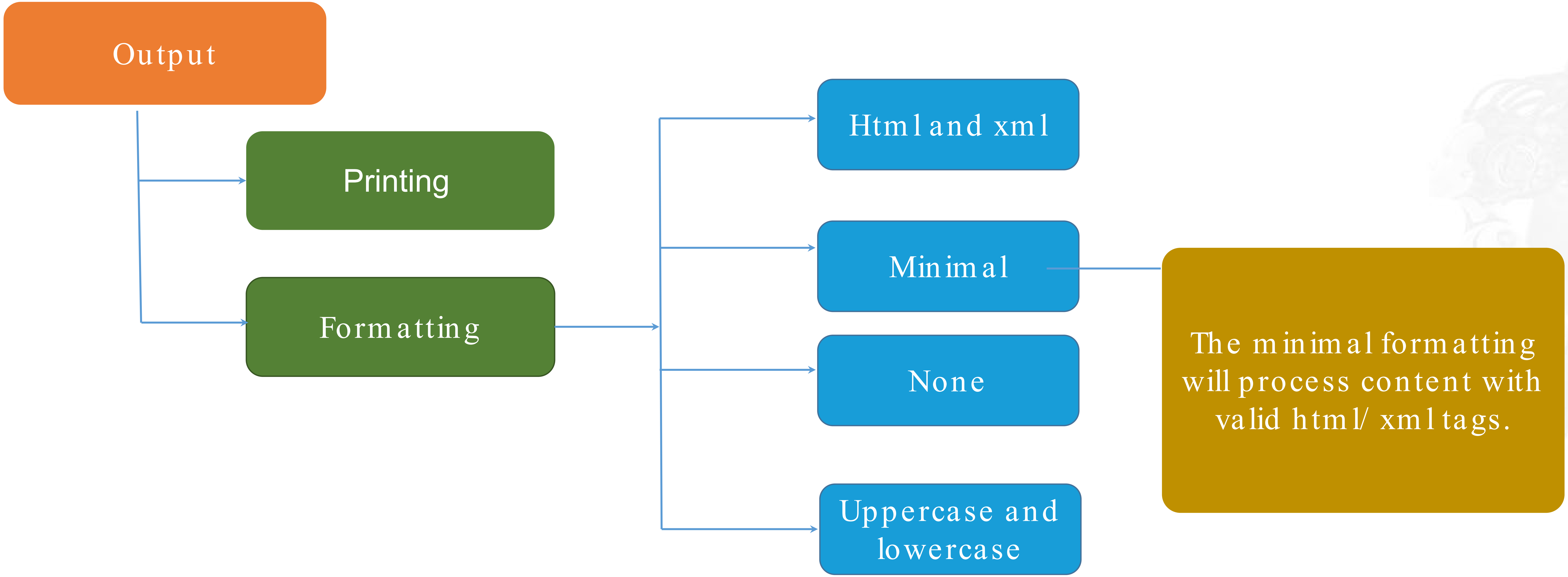
Output: Printing and Formatting

The formatters are used to generate different types of output with the desired formatting.


Output
Printing
Formatting
Html and xml
Minimal
None
Uppercase and lowercase
Html and xml formatting will convert unicode characters into html and xml entities respectively.

Output: Printing and Formatting

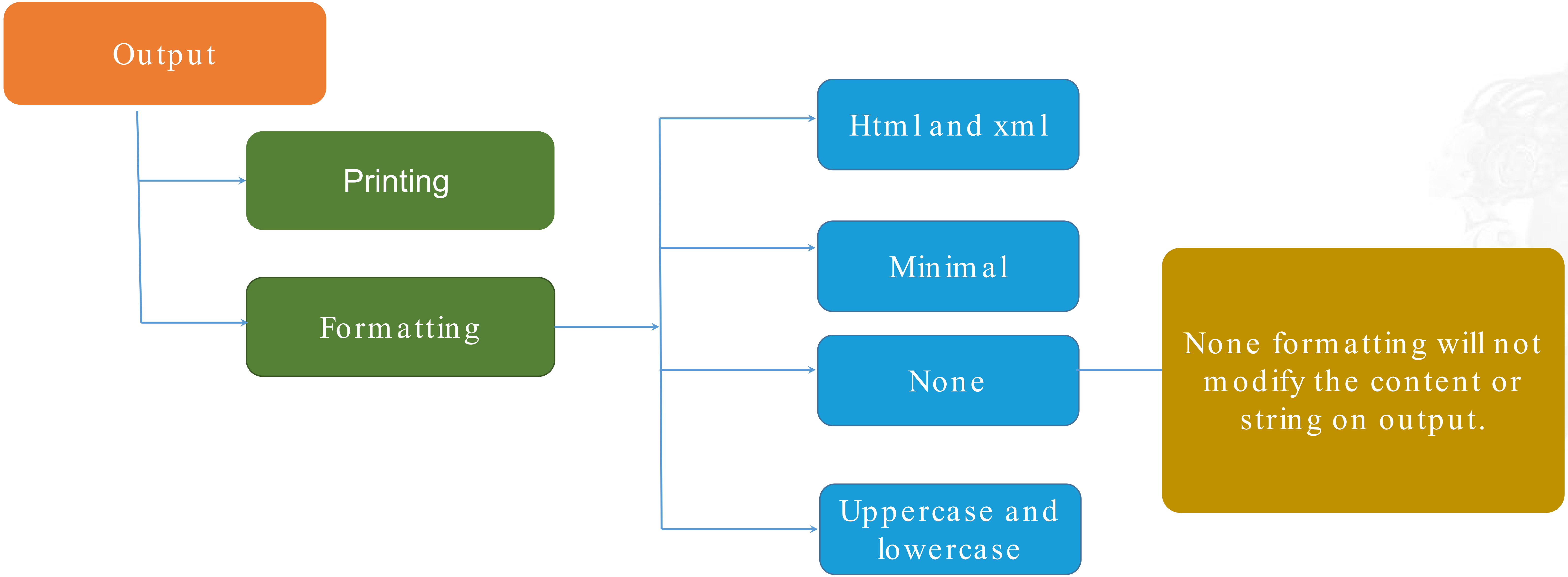
The formatters are used to generate different types of output with the desired formatting.



The minimal formatting will process content with valid html/ xml tags.

Output: Printing and Formatting

The formatters are used to generate different types of output with the desired formatting.



Output

Printing

Formatting

Html and xml

Minimal

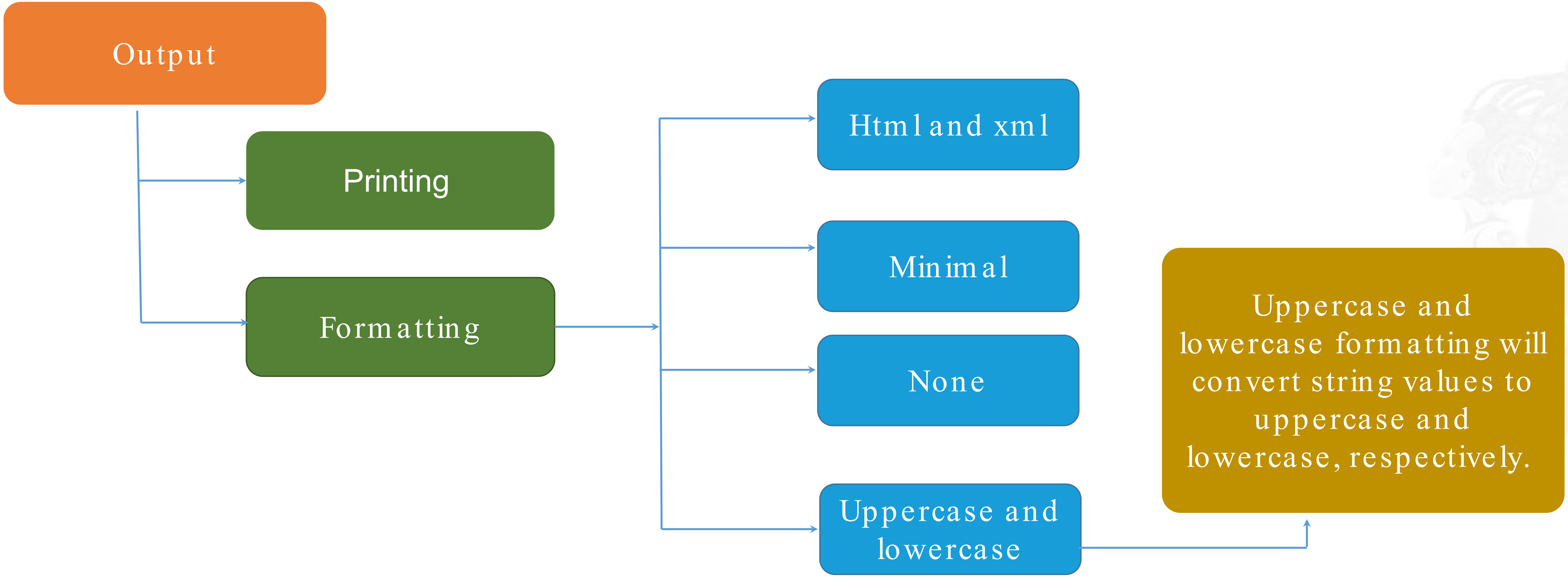
None

Uppercase and lowercase

None formatting will not modify the content or string on output.

Output: Printing and Formatting

The formatters are used to generate different types of output with the desired formatting.


Output
Printing
Formatting
Html and xml
Minimal
None
Uppercase and lowercase
Uppercase and lowercase formatting will convert string values to uppercase and lowercase, respectively.

Formatting and Printing

Demonstrate how to format, print, and encode the web document.

ASSISTED PRACTICE

Encoding

Document Encoding

- HTML or XML documents are written in specific encodings, such as ASCII or UTF-8.
- When you load the document into BeautifulSoup, it gets converted into Unicode.
- The original encoding can be extracted from attribute .original encoding of the BeautifulSoup object.

Output Encoding

- When you write a document from BeautifulSoup, you get a UTF -8 document irrespective of the original encoding.
- If some other encoding is required, you can pass it to prettify.

Web Scraping

Scrape the Simplilearn website page and perform the following tasks:

- View and print the Simplilearn web page content in a proper format
- View the head and title
- Print all the href links present in the Simplilearn web page

Simplilearn website URL: http://www.simplilearn.com/

Web Scraping

Scrape the Simplilearn website resource page and perform the following tasks:

- View and print the Simplilearn web page content in a proper format
- View the head and title
- Print all the href links present in the Simplilearn web page
- Search and print the resource headers of the Simplilearn web page
- Search resource topics
- View the article names and navigate through them

Simplilearn website URL: http://www.simplilearn.com/resources

Knowledge Check

Knowledge Check

1

Which of the following is the only xml parser?

a. html.parser

b. lxml

c. lxml.xml

d. html5lib

Knowledge Check

1

Which of the following is the only xml parser?

a. html.parser

b. lxml

c. lxml.xml

d. html5lib

The correct answer is c

lxml.xml is the only xml parser available for BeautifulSoup object.

Knowledge Check

2

In which of the following formats is the BeautifulSoup output encoded?

a. ASCII

b. Unicode

c. latin -1

d. UTF-8

Knowledge Check

2

In which of the following formats is the BeautifulSoup output encoded?

a. ASCII

b. Unicode

c. latin -1

d. UTF-8

The correct answer is **d**

The output of the BeautifulSoup is always UTF -8 encoded.

Knowledge Check

3

Which of the following libraries is used to extract a web page?

a. Beautiful Soup

b. Pandas

c. Requests

d. Numpy

Knowledge Check

3

Which of the following libraries is used to extract a web page?

a. Beautiful Soup

b. Pandas

c. Requests

d. Numpy

The correct answer is c

Requests is the right API to extract the web page.

Knowledge Check

4

Which of the following is NOT an object in BeautifulSoup?

a. Tag

b. NextSibling

c. NavigableString

d. Comment

Knowledge Check

4

Which of the following is NOT an object in BeautifulSoup?

a. Tag

b. NextSibling

c. NavigableString

d. Comment

The correct answer is **b**

NextSibling is a navigation method.

Key Takeaways

You are now able to:

- Define web scraping and explain its importance
- List the steps involved in the web scraping process
- Describe basic terminologies, such as parser, object, and tree associated with the BeautifulSoup
- Explain various operations, such as searching, modifying, and navigating the tree to yield the required result

DATA AND ARTIFICIAL INTELLIGENCE

Thank You