

# Lecture 07

BeautifulSoup4 for Web Crawling

## 1. Web Crawling 원리

## 2. BeautifulSoup4 패키지 확인 및 설치

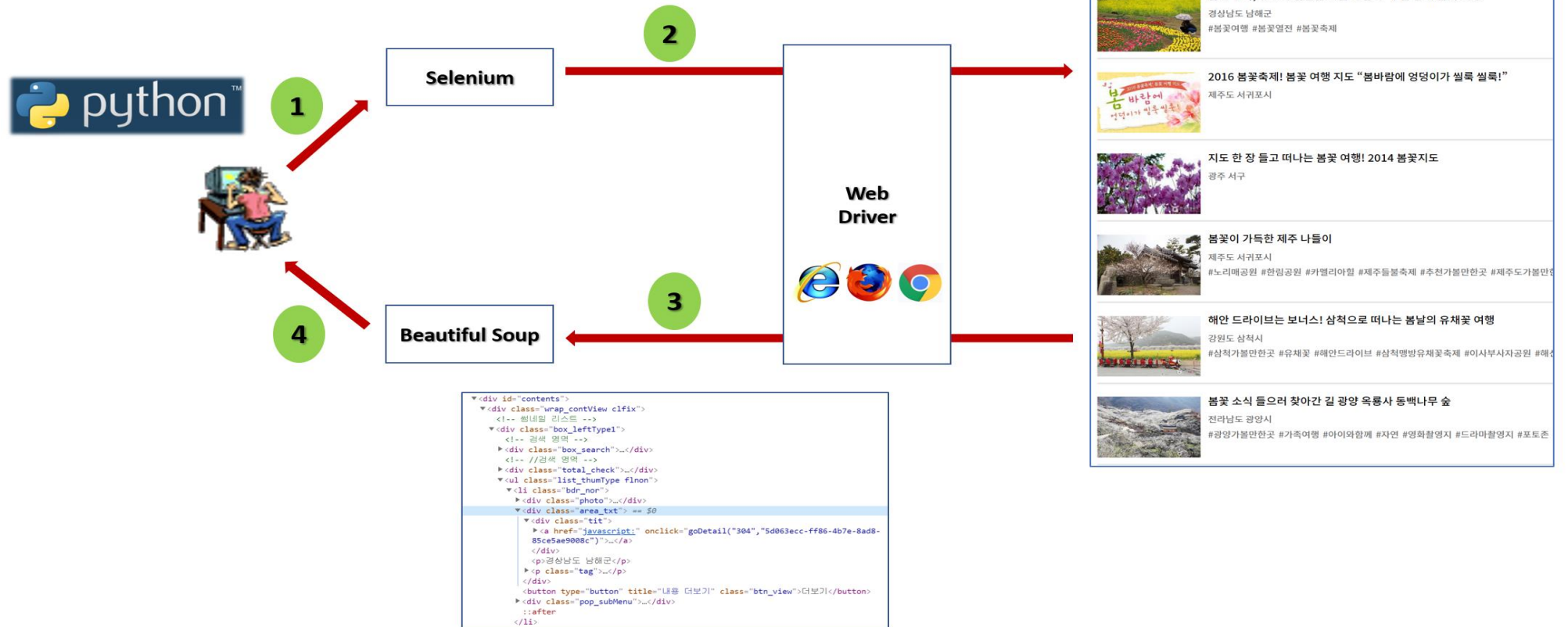
## 3. bs4 모듈의 BeautifulSoup에서 제공되는 메소드

- 3.1 find()
- 3.2 find\_all()
- 3.3 string
- 3.4 get\_text()
- 3.5 select()

## 4. 사용자 정의 식별자

## 5. 핵심 내용 실습

## Exercise



1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## ■ BeautifulSoup4 패키지 개발자

레오나르드  
리처드슨  
(Leonard  
Richardson)



비디오 게임 개발자

**Leonard Richardson** is an expert on RESTful API design, the developer of the popular Python library BeautifulSoup, and a science fiction novelist.

2020. 9. 5.

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

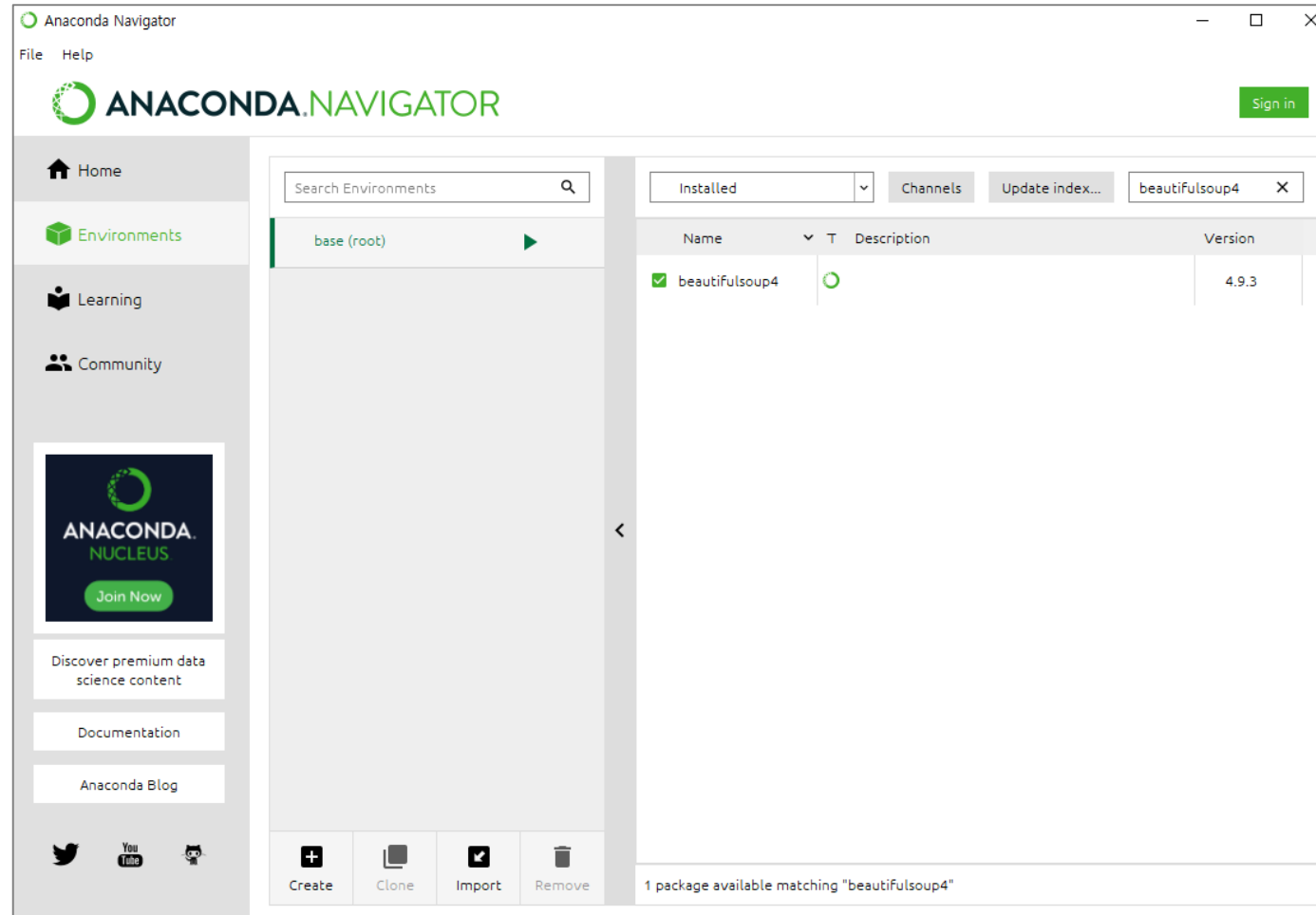
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드  
3.1 find( )  
3.2 find\_all( )  
3.3 string  
3.4 get\_text( )  
3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## ■ Anaconda Navigator 활용하여 BeautifulSoup4 패키지 확인 및 설치



1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )  
3.2 find\_all( )  
3.3 string  
3.4 get\_text( )  
3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## ■ conda 활용하여 BeautifulSoup4 패키지 확인

```
(base) C:\Users\tina> conda search BeautifulSoup4
```

```
(base) C:\Users\tina> pip search BeautifulSoup4
```

## ■ conda 또는 pip 활용하여 BeautifulSoup4 패키지 설치

```
(base) C:\Users\tina> conda install BeautifulSoup4
```

```
(base) C:\Users\tina> pip install BeautifulSoup4
```

# BeautifulSoup4 패키지 확인 및 설치

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## ■ 참고 사이트

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

<https://www.crummy.com/software/BeautifulSoup/bs4/doc.ko/>

## 뷰티플수프 문서

한글판 johnsonj 2012.11.08 [원문 위치](#)

[뷰티플수프](#)는 HTML과 XML 파일로부터 데이터를 뽑아내기 위한 파이썬 라이브러리이다. 여러분이 선호하는 해석기와 함께 사용하여 일반적인 방식으로 해석 트리를 향해, 검색, 변경할 수 있다. 주로 프로그래머의 수고를 덜어준다.

이 지도서에서는 뷰티플수프 4의 중요한 특징들을 예제와 함께 모두 보여준다. 이 라이브러리가 어느 곳에 유용한지, 어떻게 작동하는지, 또 어떻게 사용하는지, 어떻게 원하는대로 바꿀 수 있는지, 예상을 빗나갔을 때 어떻게 해야 하는지를 보여준다.

이 문서의 예제들은 파이썬 2.7과 Python 3.2에서 똑 같이 작동한다.

혹시 [뷰티플수프 3](#)에 관한 문서를 찾고 계신다면 뷰티플수프 3는 더 이상 개발되지 않는다는 사실을 꼭 아셔야겠다. 새로 프로젝트를 시작한다면 뷰티플수프 4를 적극 추천한다. 뷰티플수프 3와 뷰티플수프 4의 차이점은 [BS4 코드 이식하기](#)를 참조하자.



## 1. Web Crawling 원리

## 2. BeautifulSoup4 패키지 확인 및 설치

## 3. bs4 모듈의 BeautifulSoup 에서 제공되는 메소드

### 3.1 find( )

### 3.2 find\_all( )

### 3.3 string

### 3.4 get\_text( )

### 3.5 select( )

## 4. 사용자 정의 식별자

## 5. 핵심 내용 실습

## Exercise

### find()

서명: find(*name, attrs, recursive, text, \*\*kwargs*)

find\_all() 메소드는 전체 문서를 훑어서 결과를 찾지만, 어떤 경우는 결과 하나만 원할 수도 있다. 문서에 오직 <body> 태그가 하나 뿐임을 안다면, 전체 문서를 훑어 가면서 더 찾는 것은 시간 낭비이다. find\_all 메소드를 호출할 때마다, limit=1을 건네기 보다는 find() 메소드를 사용하는 편이 좋다. 다음 코드 두 줄은 거의 동등하다.

```
soup.find_all('title', limit=1)
# [<title>The Dormouse's story</title>]
```

```
soup.find('title')
# <title>The Dormouse's story</title>
```

유일한 차이점은 find\_all() 메소드가 단 한개의 결과만 담고 있는 리스트를 돌려주고, find()는 그냥 그 결과를 돌려준다는 점이다.

find\_all()이 아무것도 찾을 수 없다면, 빈 리스트를 돌려준다. find()가 아무것도 찾을 수 없다면, None을 돌려준다.

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## ■ 실습 노트 참고

- 07\_(9~31 pages) 강의용 BeautifulSoup.ipynb



1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## find()

```
from bs4 import BeautifulSoup
markup = '''
<html>
  <head>
    <title>
      The Dormouse's story
    </title>
  </head> '''
soup = BeautifulSoup(markup)
```

soup

```
<html>
<head>
<title>
  The Dormouse's story
</title>
</head> </html>
```

```
soup.find('title')
```

```
<title>
  The Dormouse's story
</title>
```

```
soup.find_all('title')
```

```
[<title>
  The Dormouse's story
</title>]
```

```
soup.find_all('title', limit=1)
```

```
[<title>
  The Dormouse's story
</title>]
```

```
print(soup.find('td'))
```

None

```
print(soup.find_all('td'))
```

[]

## find\_all()

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find()
    - 3.2 find\_all()
    - 3.3 string
    - 3.4 get\_text()
    - 3.5 select()
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## find\_all()

서명: find\_all(*name*, *attrs*, *recursive*, *text*, *limit*, *\*\*kwargs*)

find\_all() 메소드는 태그의 후손들을 찾아서 지정한 여과기에 부합하면 모두 추출한다.

name 인자

인자를 name에 건네면 뷰티풀수프는 특정 이름을 가진 태그에만 관심을 가진다.  
이름이 부합되지 않는 태그와 마찬가지로, 텍스트 문자열은 무시된다.

다음은 가장 단순한 사용법이다:

```
soup.find_all("title")
# [<title>The Dormouse's story</title>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find()
    - 3.2 find\_all()
    - 3.3 string
    - 3.4 get\_text()
    - 3.5 select()
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## find\_all()

서명: find\_all(name, attrs, recursive, text, limit, \*\*kwargs)

### 키워드 인자

인지되지 않는 인자는 한 태그의 속성중 하나에 대한 여과기로 변환된다. id라는 인자에 대하여 값을 하나 건네면, 뷰티풀수프는 각 태그의 'id'속성에 대하여 걸러낸다:

```
soup.find_all(id='link2')  
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>]
```

href에 대하여 값을 건네면, 뷰티풀수프는 각 태그의 'href'속성에 대하여 걸러낸다:

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find()
    - 3.2 find\_all()**
    - 3.3 string
    - 3.4 get\_text()
    - 3.5 select()
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## find\_all()

```
markup = '''
<html>
  <head>
    <title>
      The Dormouse's story
    </title>
  </head>
  <body>
    <p>
      <a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
      <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
      <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
    </p>
  </body>
'''
soup = BeautifulSoup(markup)
```

```
soup.find_all(id='link2')
```

```
[<a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

`.string`

**NavigableString** : 태그 안에 있는 텍스트

태그에 오직 자손이 하나라면, 그리고 그 자손이 NavigableString이라면, 그 자손은 `.string`으로 얻을 수 있다.

```
title_tag.string  
# u'The Dormouse's story'
```

태그에 하나 이상의 태그가 있다면, `.string`이 무엇을 가리킬지 확실하지 않다. 그래서 그럴 경우 `.string`은 `None`으로 정의된다.

```
print(soup.html.string)  
# None
```

1. Web Crawling  
원리
2. BeautifulSoup4  
패키지 확인 및  
설치
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
  - 3.1 find( )
  - 3.2 find\_all( )
  - 3.3 string
  - 3.4 get\_text( )
  - 3.5 select( )
4. 사용자 정의  
식별자
5. 핵심 내용 실습

## Exercise

**.string**

```
markup = '<a href="http://example.com/">#nI linked to <i>example.com</i>#n</a>'  
soup = BeautifulSoup(markup)
```

```
print(soup.string)
```

None

```
markup = '<a href="http://example.com/">#nI linked to example.com #n</a>'  
soup = BeautifulSoup(markup)
```

```
print(soup.string)
```

I linked to example.com

# get\_text()

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## get\_text()

문서나 태그에서 텍스트 부분만 추출하고 싶다면, `get_text()` 메소드를 사용할 수 있다. 이 메소드는 문서나 태그 아래의 텍스트를, 유니코드 문자열 하나로 모두 돌려준다:

```
markup = '<a href="http://example.com/">\n linked to <i>example.com</i>\n</a>'
soup = BeautifulSoup(markup)

soup.get_text()
u'\n linked to example.com\n'
soup.i.get_text()
u'example.com'
```

텍스트를 합칠 때 사용될 문자열을 지정해 줄 수 있다:

```
# soup.get_text("|")
u'\n linked to |example.com|\n'
```

뷰티풀수프에게 각 텍스트의 앞과 뒤에 있는 공백을 걷어내라고 알려줄 수 있다:

```
# soup.get_text("|", strip=True)
u'l linked to|example.com'
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

get\_text()

```
from bs4 import BeautifulSoup
markup = '<a href="http://example.com/">¶¶ linked to <i>example.com</i>¶¶</a>'
soup = BeautifulSoup(markup)
```

```
soup.get_text()
```

```
'¶¶ linked to example.com¶¶'
```

```
soup.i.get_text()
```

```
'example.com'
```

```
markup
```

```
'<a href="http://example.com/">¶¶ linked to <i>example.com</i>¶¶</a>'
```



1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

## get\_text()

markup

```
'<a href="http://example.com/">¶¶¶¶ linked to <i>example.com</i>¶¶¶¶</a>'
```

```
soup.get_text("~~~")
```

```
'¶¶¶¶ linked to ~~~example.com~~~¶¶¶¶'
```

```
soup.get_text(strip=True)
```

```
'¶ linked toexample.com'
```

```
soup.get_text("~~~", strip=True)
```

```
'¶ linked to~~~example.com'
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

get\_text()

markup

```
'<a href="http://example.com/">#n| linked to <i>example.com</i>#n</a>'
```

```
soup.get_text("|")
```

```
'#n| linked to |example.com|#n'
```

```
soup.get_text(strip=True)
```

```
'| linked toexample.com'
```

```
soup.get_text("|", strip=True)
```

```
'| linked to|example.com'
```

1. Web Crawling  
원리
2. BeautifulSoup4  
패키지 확인 및  
설치
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
  - 3.1 find( )
  - 3.2 find\_all( )
  - 3.3 string
  - 3.4 get\_text( )
  - 3.5 select( )
4. 사용자 정의  
식별자
5. 핵심 내용 실습

Exercise

다음과 같이 태그를 검색할 수 있다:

```
soup.select("title")  
# [<title>The Dormouse's story</title>]
```

다른 태그 아래의 태그를 찾을 수 있다:

```
soup.select("body a")  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
# <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

```
soup.select("html head title")  
# [<title>The Dormouse's story</title>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

```
from bs4 import BeautifulSoup
markup = '''
<html>
  <head>
    <title>
      The Dormouse's story
    </title>
  </head>
  <body>
    <p>
      <a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
      <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
      <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
    </p>
  </body>
'''
soup = BeautifulSoup(markup)
```

```
soup.select("body a")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
<a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
<a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

```
soup.select("html head title")
```

```
[<title>
  The Dormouse's story
</title>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

다른 태그 바로 아래에 있는 태그를 찾을 수 있다:

```
soup.select("head > title")  
# [<title>The Dormouse's story</title>]
```

```
soup.select("p > a")  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

```
soup.select("body > a")  
# []
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

```
soup.select("head > title")
```

```
[<title>
  The Dormouse's story
</title>]
```

```
soup.select("p > a")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
  <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
  <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

```
soup.select("body > a")
```

```
[]
```

# select( )

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

CSS 클래스로 태그를 찾는다:

```
soup.select(".sister")  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

```
soup.select("[class~=sister]")  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

1. Web Crawling  
원리
2. BeautifulSoup4  
패키지 확인 및  
설치
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
  - 3.1 find( )
  - 3.2 find\_all( )
  - 3.3 string
  - 3.4 get\_text( )
  - 3.5 select( )
4. 사용자 정의  
식별자
5. 핵심 내용 실습

## Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

```
soup.select(".sister")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
 <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
 <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

```
soup.select("[class~=sister]")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
 <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
 <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

```
soup.select("[class=sister]")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
 <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
 <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```



1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

※ 참고 : find or find\_all ('태그', 'class name') 가능

```
1 soup.find('a', 'sister')
```

class name

```
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>
```

```
1 soup.find_all('a', 'sister')
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
<a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
<a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

※ 참고 : find or find\_all ('태그', 'id name') 불가능

```
1 print(soup.find('a', 'link1'))
```

id name

None

```
1 print(soup.find_all('a', 'link1'))
```

[]

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

ID로 태그를 찾는다:

```
soup.select("#link1")  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>]
```

```
soup.select("a#link2")  
# [<a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>]
```

속성이 존재하는지 테스트 한다:

```
soup.select('a[href]')  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
# <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

## 1. Web Crawling 원리

## 2. BeautifulSoup4 패키지 확인 및 설치

## 3. bs4 모듈의 BeautifulSoup 에서 제공되는 메소드

### 3.1 find( )

### 3.2 find\_all( )

### 3.3 string

### 3.4 get\_text( )

### 3.5 select( )

## 4. 사용자 정의 식별자

## 5. 핵심 내용 실습

### Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

```
soup.select("#link1")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>]
```

```
soup.select("a#link1")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>]
```

```
soup.select("a[href]")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
<a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
<a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

속성 값으로 태그를 찾는다:

```
soup.select('a[href="http://example.com/elsie"]')  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>]
```

```
soup.select('a[href^="http://example.com/"]')  
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,  
# <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,  
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

```
soup.select("a[href='http://example.com/elsie']")
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>]
```

```
soup.select('a[href="http://example.com/elsie"]')
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>]
```

```
soup.select("a[href="http://example.com/elsie"]")
```

File "<ipython-input-91-81bcc105c168>", line 1

```
soup.select("a[href="http://example.com/elsie"]")
```

SyntaxError: invalid syntax

1. Web Crawling  
원리
  2. BeautifulSoup4  
패키지 확인 및  
설치
  3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  4. 사용자 정의  
식별자
  5. 핵심 내용 실습
- Exercise

soup

```
<html>
<head>
<title>
    The Dormouse's story
</title>
</head>
<body>
<p>
<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
    <a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
    <a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>
</p>
</body> </html>
```

```
soup.select('a[href^="http://example.com/"]')
```

```
[<a class="sister" href="http://example.com/elsie" id="link1"> Elsie </a>,
<a class="sister" href="http://example.com/lacie" id="link2"> Lacie </a>,
<a class="sister" href="http://example.com/tillie" id="link3"> Tillie </a>]
```

```
soup.select('a[href="http://example.com/"]')
```

```
[]
```

# 사용자 정의 식별자

- 1. Web Crawling  
원리
  - 2. BeautifulSoup4  
패키지 확인 및  
설치
  - 3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
    - 3.1 find( )
    - 3.2 find\_all( )
    - 3.3 string
    - 3.4 get\_text( )
    - 3.5 select( )
  - 4. 사용자 정의  
식별자
  - 5. 핵심 내용 실습
- Exercise

- 사용자 정의 식별자 정의 규칙 : 영문자 + 숫자 + \_(언더바)

## 1. id

- 문서 내에 유일한 객체 식별자
- 중복값 발생 X
- 모든 태그에 적용 가능
- HTML 보다는 CSS, JavaScript 에서 주로 사용

## 2. class

- 성격이 동일한 태그들을 그룹으로 나타내기 위한 식별자
- 중복값 발생 O
- 모든 태그에 적용 가능
- CSS, JavaScript 에서 주로 사용

## 3. name

- <a>,<img>,<form> 태그에만 적용 가능
- HTML에서만 사용
- CSS, JavaScript에서 사용 안함 (일부 사용 가능)



1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## ■ 핵심 내용 요약 실습

- 07\_(01\_summary) 강의용 BeautifulSoup.ipynb
- 07\_(02\_summary) 강의용 BeautifulSoup.ipynb
- 07\_(03\_summary) 강의용 BeautifulSoup.ipynb

1. Web Crawling  
원리
2. BeautifulSoup4  
패키지 확인 및  
설치
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
  - 3.1 find( )
  - 3.2 find\_all( )
  - 3.3 string
  - 3.4 get\_text( )
  - 3.5 select( )
4. 사용자 정의  
식별자
5. 핵심 내용 실습

## Exercise

“BS\_Exercise(ANSI).html” 과 “BS\_Exercise(UTF-8).html” 파일을 아래와 같이 BeautifulSoup을 테스트한 후,  
둘 중 하나를 선택하여 다음 페이지에서 제시하는 **지시사항**을 완성하십시오.

```
1 from bs4 import BeautifulSoup
```



BS\_Exercise(ANSI).html

ANSI 코드 문서일 경우

```
1 with open("D:/ai/data/BS_Exercise(ANSI).html") as ex:  
2     soup = BeautifulSoup(ex, 'html.parser')
```



BS\_Exercise(UTF-8).html

UTF-8 코드 문서일 경우

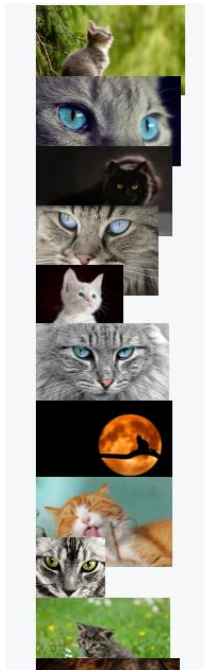
```
1 with open("D:/ai/data/BS_Exercise(UTF-8).html", encoding="utf-8") as ex:  
2     soup = BeautifulSoup(ex, 'html.parser')
```

1. Web Crawling  
원리
2. BeautifulSoup4  
패키지 확인 및  
설치
3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드
  - 3.1 find( )
  - 3.2 find\_all( )
  - 3.3 string
  - 3.4 get\_text( )
  - 3.5 select( )
4. 사용자 정의  
식별자
5. 핵심 내용 실습

## Exercise

## 지시 사항

1. 앞 페이지 코딩 결과로 soup 변수에 있는 html 코드 중에서 사진 부분의 정보를 모두 출력하고자 합니다. 결과는 아래 그림처럼 총 100 개가 출력되도록 빈칸의 코드를 완성하시오.



```
1 img_src = soup. ( 'div', 'flex_grid credits search_results' ). ( 'img' )
2
3 print( len( img_src ) )
```

100

1 img\_src

```
[,
,
,
,
,
,
]
```

100개

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## 지시 사항

2. for 반복문을 활용하여 아래와 같은 결과가 나오도록 코딩하시오.  
100개의 img\_src 목록에서 src 속성만 추출한 결과입니다.

`https://cdn.pixabay.com/photo/2017/02/20/18/03/cat-2083492__340.jpg`

`https://cdn.pixabay.com/photo/2016/03/28/12/35/cat-1285634__340.png`

`https://cdn.pixabay.com/photo/2015/03/27/13/16/cat-694730__340.jpg`

`https://cdn.pixabay.com/photo/2016/07/10/21/47/cat-1508613__340.jpg`

`⋮`

`https://cdn.pixabay.com/photo/2016/07/18/20/30/tiger-1526704__340.png`

`https://cdn.pixabay.com/photo/2015/03/26/10/21/cat-691175__340.jpg`

`https://cdn.pixabay.com/photo/2013/07/13/01/12/witch-155291__340.png`

`https://cdn.pixabay.com/photo/2016/11/14/03/56/tiger-1822535__340.jpg`

100개

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## 지시 사항

3. 추출된 결과 중에서 2017년 데이터를 `img_src_2017` 리스트에 저장하여 출력하는 코드를 작성하시오.

2017년 데이터는 총 20건 입니다

```
https://cdn.pixabay.com/photo/2017/02/20/18/03/cat-2083492__340.jpg
https://cdn.pixabay.com/photo/2017/11/14/13/06/kitty-2948404__340.jpg
https://cdn.pixabay.com/photo/2017/07/24/19/57/tiger-2535888__340.jpg
https://cdn.pixabay.com/photo/2017/11/06/09/53/animal-2923186__340.jpg
https://cdn.pixabay.com/photo/2017/07/25/01/22/cat-2536662__340.jpg
https://cdn.pixabay.com/photo/2017/05/17/12/42/tiger-2320819__340.jpg
https://cdn.pixabay.com/photo/2017/11/09/21/41/cat-2934720__340.jpg
https://cdn.pixabay.com/photo/2017/12/09/21/33/sunset-3008779__340.jpg
https://cdn.pixabay.com/photo/2017/03/14/14/49/cat-2143332__340.jpg
https://cdn.pixabay.com/photo/2017/08/23/08/33/cats-eyes-2671903__340.jpg
https://cdn.pixabay.com/photo/2017/12/11/15/34/lion-3012515__340.jpg
https://cdn.pixabay.com/photo/2017/04/30/18/33/cat-2273598__340.jpg
https://cdn.pixabay.com/photo/2017/11/13/07/14/cat-eyes-2944820__340.jpg
https://cdn.pixabay.com/photo/2017/10/30/19/41/puma-2903312__340.jpg
https://cdn.pixabay.com/photo/2017/01/12/21/42/amur tiger-1975790__340.jpg
https://cdn.pixabay.com/photo/2017/01/16/23/10/snow-leopard-1985510__340.jpg
https://cdn.pixabay.com/photo/2017/08/07/18/57/dog-2606759__340.jpg
https://cdn.pixabay.com/photo/2017/02/15/12/12/cat-2068462__340.jpg
https://cdn.pixabay.com/photo/2017/03/29/09/59/cat-2184682__340.jpg
https://cdn.pixabay.com/photo/2017/11/22/08/07/cat-2969932__340.jpg
```

1. Web Crawling  
원리

2. BeautifulSoup4  
패키지 확인 및  
설치

3. bs4 모듈의  
BeautifulSoup  
에서 제공되는  
메소드

3.1 find( )

3.2 find\_all( )

3.3 string

3.4 get\_text( )

3.5 select( )

4. 사용자 정의  
식별자

5. 핵심 내용 실습

Exercise

## 지시 사항

4. 추출된 결과 중에서 2018년 데이터를 `img_src_2018` 리스트에 저장하여 출력하는 코드를 작성하시오.

2018년 데이터는 총 18건 입니다

```
https://cdn.pixabay.com/photo/2018/07/31/22/08/lion-3576045__340.jpg
https://cdn.pixabay.com/photo/2018/03/26/20/49/tiger-3264048__340.jpg
https://cdn.pixabay.com/photo/2018/01/25/14/12/nature-3106213__340.jpg
https://cdn.pixabay.com/photo/2018/01/28/12/37/cat-3113513__340.jpg
https://cdn.pixabay.com/photo/2018/05/04/16/50/cat-3374422__340.jpg
https://cdn.pixabay.com/photo/2018/04/13/21/24/lion-3317670__340.jpg
https://cdn.pixabay.com/photo/2018/07/08/14/16/cat-3523992__340.jpg
https://cdn.pixabay.com/photo/2018/01/04/18/58/cats-3061372__340.jpg
https://cdn.pixabay.com/photo/2018/11/29/23/34/cat-3846780__340.jpg
https://cdn.pixabay.com/photo/2018/07/13/10/20/cat-3535404__340.jpg
https://cdn.pixabay.com/photo/2018/05/03/22/34/lion-3372720__340.jpg
https://cdn.pixabay.com/photo/2018/06/03/08/57/cat-3449999__340.jpg
https://cdn.pixabay.com/photo/2018/03/27/17/25/cat-3266673__340.jpg
https://cdn.pixabay.com/photo/2018/03/26/02/05/cat-3261420__340.jpg
https://cdn.pixabay.com/photo/2018/04/20/17/18/cat-3336579__340.jpg
https://cdn.pixabay.com/photo/2018/01/11/23/16/woman-3077180__340.jpg
https://cdn.pixabay.com/photo/2018/05/30/19/29/cat-3442257__340.jpg
https://cdn.pixabay.com/photo/2018/07/31/11/14/lion-3574819__340.jpg
```