



2024

Python package to work with torob.com

Version : 1.0



Start work

import package :

```
import torob  
from torob import att  
from torob import city
```

att : To get a series of attributes of the desired product, we import this package.

city : To specify the city, to bring results related to the city (default = Tehran)



Product search

Simple search:

```
a = torob.Search('ps5')
```

A little more precise:

```
a = torob.Search('ps5',range_price=(20000000 , 35000000) , status='new' )
```

or

```
a = torob.Search('ps5' , city.MASHHAD , status='stock' ,unknown=True , order='cheap')
```

search parameters:

goods : The value to search [str]

number : Number of goods received (default 24) [int]

page : search page (default 0) [int]

city : Search city (default Tehran) [Select city from 'city' module]

in_person : In person (default True) [bool]

status : Product status (new or stock) (default None) [str]

order : Order type (cheap or popular) (default None) [str]

range_price : Price range (as a tuple , specifies minimum and maximum price) [tuple]

default = (None , None)

unknown : If it is true, no cookies will be sent by the package to torob.com, so the results will be different. (default False) [bool]



General information

Number of products found :

```
print(len(a))
```

Product category(ies) :

```
print(a.categorys)
```

lowest price:

```
print(a.min_price)
```

most expensive price:

```
print(a.max_price)
```



More details

Get product images:

```
a.get_images(0)
```

parameter :

1- index : Product number (0 to len(a) products have been found)

Output :

Returns a list of product image urls

Get product(s) page url :

```
a.get_links()
```

output:

Returns a list of product addresses

Get information about the product (specifications, descriptions, features):

```
a.get_info(0)
```

parameter :

1- index : Product number (0 to len(a) products have been found)

2-strip: If set to True, it removes extra spaces

3- out_put_type: Specifies the type of output (list or string) (default str)

4- up_to_page : It repeats this process up to a certain page

Output:

Returns product specifications (as a list or text)

Returns the information of the found products (on current page):

```
a.get_All(att.NAME_EN ,att.PRICE , only=True ,)
```

parameter :

1- args : With the help of the imported att class, get the attribute it needs from the products.

2-only : If it is equal to True, it returns as a list and otherwise it returns as a dictionary.

(If the number of results is less than 22, the dictionary key is Latin letters, and if the results are more than 22, the dictionary key is equal to the value you searched.)

3-up_to_page : Returns all product details up to the "up_to_page" page

Output:

The dictionary returns the product specifications of this page

seller details (phone, location, address):

```
a.get_seller(1)
```

parameter :

1- index : Product number (0 to len(a) products have been found)

2-up_to_page : Returns all product details up to the "up_to_page" page

Output:

It returns the contact number, SMS number, store address and location of the seller

Get related searches (similar):

```
a.get_suggestion()
```

output :

return list of suggestion

Goods in other stores:

```
a.get_more_store(0)
```

parameter :

1- index : Product number (0 to len(a) products have been found)

Output:

It returns the specifications of similar products in other stores.

Returns the specifications of similar products:

```
a.get_similar(0 , short=True)
```

parameter :

1- index : Product number (0 to len(a) products have been found)

2- short: If it is True; Shows the output more concisely

Output :

Returns a dictionary of similar products (containing the name, price, city and address of the product photo)



save products information

```
a.save(att.PRICE , att.NAME_FA , formating='excel'  
,add_row=True , add_info=True , replaced='ناموجود')
```

parameters:

1-args: With the help of the imported att class, get the attribute it needs from the products

2-up_to_page : Returns all product details up to the "up_to_page" page

3-formating : Specifies the saved format, it can be equal to the following:

A) 'excel' or 'xlsx' or 'xls' or 'e' = To save as Excel

B) 'xml' or 'x' = To save as XML

C) 'sql' or 'db' or 's' = To save as SQL

D) 'html' or 'h' or 'table' or 'htm' = To save as HTML

E) 'str' or 'txt' or 'text' or 'string' = To save as TEXT

F) 'hdf5' or 'h5' = To save as HD5

G) 'feather' or 'pandas' or 'f' = To save as PANDAS format

H) 'json' or 'j' = To save as JSON

I) 'csv' or 'c' or 'csvx' = To save as CSV

4-replaced : If the product does not have this feature, it replaces this value in the output file (default "-")

5-file_name : Output file name

6-add_info : If it is True, it will add product details to the file with the help of the get_info method (default False)

7- add_row : If it is True, it puts one row (one number) for each product



Professional features

Get the specifications of a product (first product):

```
a = torob.Search('ps5')(att.NAME_EN , goods_limit=(0,0))  
  
print(a)
```

help:

After defining the object, open the parentheses again, inside these parentheses, write the features you need with the help of **att**, the second input of this parentheses is **goods_limit**.

This parameter specifies from which product to which product to return

Other example : Returns the Latin name of the first 10 products on this page

```
a = torob.Search('ps5')(att.NAME_EN, goods_limit=(0,9))
print(a)
```

Returns the price of the 4th to 7th item:

```
b = a(att.PRICE , goods_limit=(4,7))
print(b)
```

Returns the price and name of the 4th to 7th item:

```
b = a(att.PRICE , att.NAME_EN, goods_limit=(4,7))
print(b)
```

...

Returns the price and name of all products:

```
b = a(att.PRICE , att.NAME_EN, goods_limit=(0,len(a)))
print(b)
```




Get store information

Get 3 stores whose names **include** "پارس":

```
c = torob.Shop(name='پارس' , count=3)
```

parameters:

1-name: The name of the desired store

2- count : Number of stores received

Getting store information:

```
print(c.get_information())
```

Getting the number of stores that have the same name:

```
print(c.get_count())
```

Get address of the store:

```
print(c.get_address(2))
```

parameters:

1-index: Which store? store index (from 0 to count-1) can be

Returns store's performance score:

```
print(c.get_score(1))
```

parameters:

1-index: Which store? store index (from 0 to count-1) can be

Returns the store's permissions and credentials:

```
print(c.get_license(1))
```

Returns the cooperation history of the store:

```
print(c.get_history(1))
```

Helped by:

