

Hola Mundo / Desarrollo de Software / Python / Automatizando con Python, Idea 3: Job Scrapper

Automatizando con Python, Idea 3: Job Scrapper

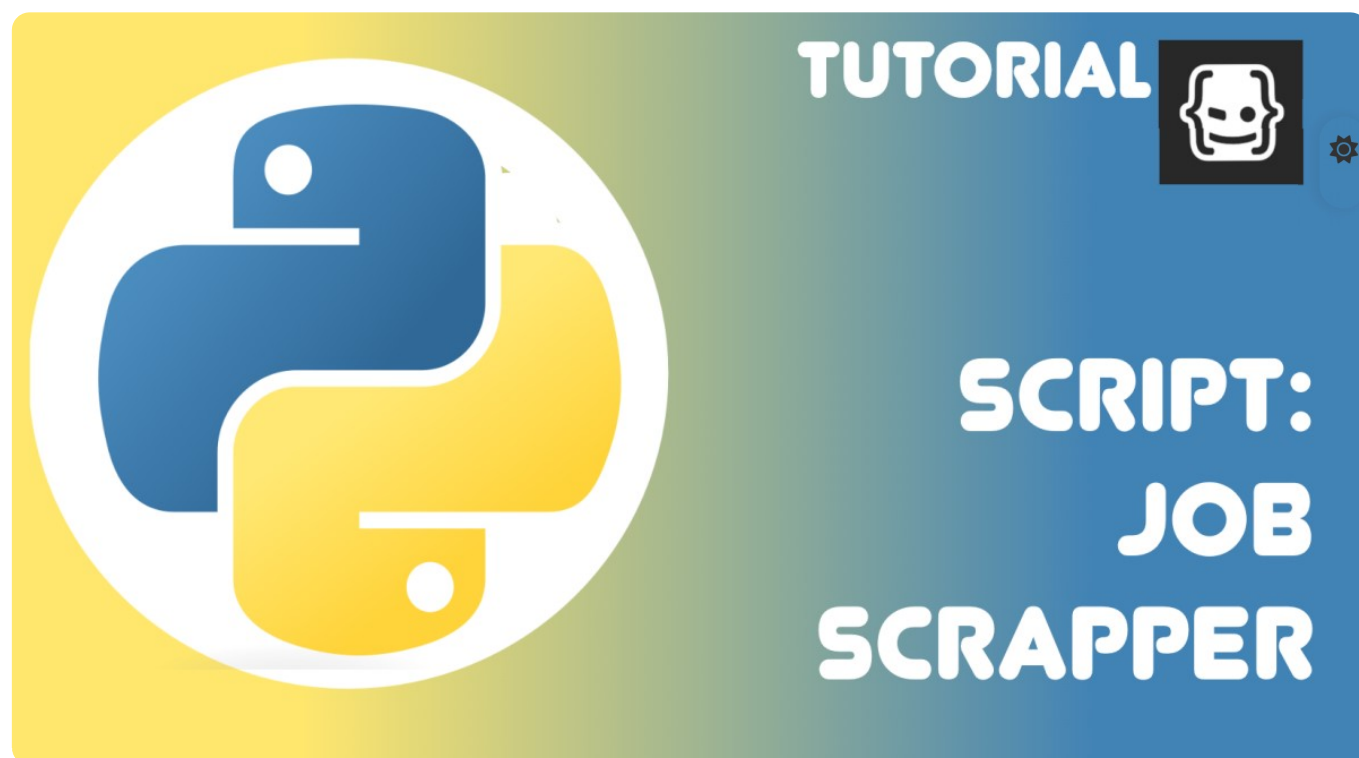


Nicolas Schurmann

• Desarrollo de Software, Python, Tutorial

• febrero 1, 2023

•  (1)



Cuando estás aprendiendo a programar lo mejor que puedes hacer es buscar formas de practicar. Con ideas de proyectos o incluso intentando automatizar algunas tareas diarias que puedes tener. Es por esta razón que en este post te traigo la tercera y última de tres ideas que puedes automatizar en Python. Aunque estas ideas, por supuesto que también las puedes automatizar en cualquier lenguaje.

Hoy tenemos un tutorial para automatizar con Python, puedes leer este post o ver y escuchar esta información en formato de video en nuestro canal de youtube en la que encontraras además de esta, dos ideas más para automatizar con python, [te dejamos el link por si prefieres este formato.](#) 😊👍

Hola mundo y bienvenidos a este post,

Puede parecer algo complejo si aún no tenemos las bases del lenguaje, para esto te sugerimos aprender con el curso de de python que tenemos en la **Academia de Hola Mundo** para aprender los fundamentos y como construir aplicaciones web, has click qui para ver el curso: [Python: HTML, CSS, Flask y MySQL](#).

Descripción de nuestro proyecto “Job Scraper”

A continuación, vamos a hablar sobre la mejor alternativa para encontrar el trabajo remoto de tus sueños.

Este script buscará trabajos sobre algún patrón de búsqueda, como por ejemplo javascript o python, y guardará los resultados en un archivo de texto, este ustedes si desean lo pueden modificar para que envíe por correo el archivo o incluso agregar más condiciones como por ejemplo que solo muestre los trabajos que tengan un salario ofertado.

Preparación

Primero crearemos una nueva carpeta donde guardaremos nuestro proyecto y dentro de esta carpeta crearemos un nuevo archivo, lo llamaremos **scraper.py**. Esta estructura la podemos lograr con la secuencia de comandos para tener el archivo necesario para trabajar con este script.



```
1 | mkdir python_projects
```

```
1 | cd python_projects/
```

```
1 | touch scraper.py
```

Con esto tendríamos la carpeta y el archivo creado, tú puedes elegir la ruta donde crear estos archivos.

Comenzando el script

Ahora abriremos nuestro editor de código como lo es VsCode, pero tú puede utilizar el de tu preferencia para trabajar nuestro proyecto.

Dentro del archivo **scraper.py** tenemos que importar los módulos:

request, el cual nos permitirá hacer peticiones a la web, y el módulo

bs4, de este usaremos a **BeautifulSoup**, el cual nos servirá para hacer scrapping con más facilidad,

Pero antes tendremos que instalarlos y lo haremos con el comando:

esto sí estás usando comandos Unix.

```
1 | python3 -m pip install requests beautifulsoup4
```

Nuestras importaciones deberian quedar de la siguiente manera:

```
1 | import request
2 | from bs4 import BeautifulSoup
```

Definiremos una variable con la url de donde vendran los datos, en este caso usaremos el sitio [seek.co.nz](https://www.seek.co.nz) y la URL será la siguiente: <https://www.seek.co.nz/python-jobs?salaryrange=100000-999999&salarytype=annual>

```
1 | url = "https://www.seek.co.nz/python-jobs?salaryrange=100000-999999&salarytype=annual"
```

Ahora inicializamos nuestra aplicación con y todo nuestro script va dentro de este bloque de código:

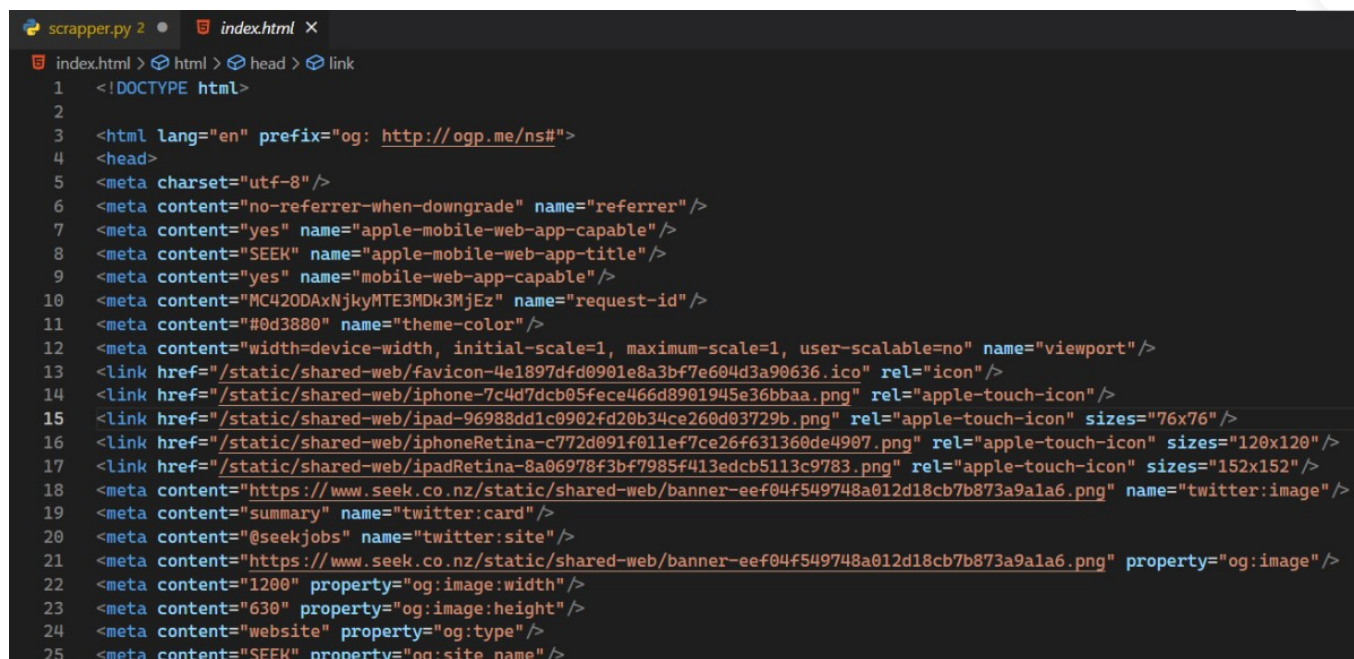
```
1 | if __name__ == "__main__":
```

Obteniendo los datos

Comenzaremos con traer los datos de Internet o haremos con las siguientes líneas de código, primero trayendo la información con `request`, guardaremos este resultado en una variable `page` y después con `BeautifulSoup` interpretaremos el contenido de la petición y lo guardaremos en una variable llamada `soup`,

```
1 | page = requests.get(url)
2 | soup = BeautifulSoup(page.content, "html.parser")
```

Si el resultado de esta variable `soup` lo guardáramos en un HTML, veríamos algo como lo siguiente, te lo dejo por aquí para ver el resultado de ejecutar estas dos líneas de código, esto no lo tenemos que hacer:



```

1 | <!DOCTYPE html>
2 |
3 | <html lang="en" prefix="og: http://ogp.me/ns#">
4 | <head>
5 | <meta charset="utf-8" />
6 | <meta content="no-referrer-when-downgrade" name="referrer" />
7 | <meta content="yes" name="apple-mobile-web-app-capable" />
8 | <meta content="SEEK" name="apple-mobile-web-app-title" />
9 | <meta content="yes" name="mobile-web-app-capable" />
10 | <meta content="MC420DAXNjkyMTE3MDk3MjEz" name="request-id" />
11 | <meta content="#0d3880" name="theme-color" />
12 | <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no" name="viewport" />
13 | <link href="/static/shared-web/favicon-4e1897dfd0901e8a3bf7e604d3a90636.ico" rel="icon" />
14 | <link href="/static/shared-web/iphone-7c4d7dcb05fece466d8901945e36bbaa.png" rel="apple-touch-icon" />
15 | <link href="/static/shared-web/ipad-96988dd1c0902fd20b34ce260d03729b.png" rel="apple-touch-icon" sizes="76x76" />
16 | <link href="/static/shared-web/iphoneRetina-c772d091f011ef7ce26f631360de4907.png" rel="apple-touch-icon" sizes="120x120" />
17 | <link href="/static/shared-web/ipadRetina-8a06978f3bf7985f413edcb5113c9783.png" rel="apple-touch-icon" sizes="152x152" />
18 | <meta content="https://www.seek.co.nz/static/shared-web/banner-ee04f549748a012d18cb7b873a9a1a6.png" name="twitter:image" />
19 | <meta content="summary" name="twitter:card" />
20 | <meta content="@seekjobs" name="twitter:site" />
21 | <meta content="https://www.seek.co.nz/static/shared-web/banner-ee04f549748a012d18cb7b873a9a1a6.png" property="og:image" />
22 | <meta content="1200" property="og:image:width" />
23 | <meta content="630" property="og:image:height" />
24 | <meta content="website" property="og:type" />
25 | <meta content="SEEK" property="og:site_name" />

```

Resultado de la variable soup

Ahora definimos una función para buscar según las tags de `HTML` en esta página específicamente se guardan las ofertas con el atributo `data-search-sol-meta`, entonces vamos a buscar aquellos elementos en este `HTML` que contengan estas etiquetas, lo haremos de la siguiente manera:

```
1 | def has_data_search(tag):
2 |     return tag.has_attr("data-search-sol-meta")
```

Ahora en una lista vamos a obtener solo los resultados de las ofertas laborales, ya que no necesitamos todo el `HTML`, vamos a guardar esto en una variable llamada `results`, la cual va a usar a la variable `soup` y vamos a buscar en esta cuando encuentre el tag que colocamos en la función `has_data_search`:

```
1 | results = soup.find_all(has_data_search)
```

Extrayendo los datos que necesitamos

Ahora vamos a obtener los datos que necesitamos de cada oferta laboral, vamos a buscar:

- Título de la vacante,
- Empresa,
- Salario, y
- Link de la vacante

Entonces usando el ciclo `for` vamos a recorrer la lista de los elementos obtenidos en `results`, a cada resultado le llamaremos `job` y usaremos un bloque `try/except` ya que estamos usando código que podría salir mal por el tipo de ejecución que hacemos en el script

Vamos a buscar dentro de cada `job` mediante un `find`, la etiqueta con el atributo específico de cada elemento que buscamos y después con `.get_text()` obtendremos su texto.

Después definiremos a `job` como el resultado del siguiente texto: `job = "Titulo: {}\nEmpresa: {}\nSalario: {}\nLink: {}a\n"` y sustituiremos los valores con nuestros valores obtenidos del paso anterior con el método `format()` y lo hacemos de la siguiente manera: `job = job.format(title, company, salary, joblink)`

Al final imprimimos cada `job`. Pero no olvidemos que debemos colocar el bloque de `except`. Todo este código.

Todo este proceso nos debe quedar así:

```
1 | for job in results:
2 |     try:
3 |         titleElement = job.find("a", attrs={"data-automation": "jobTitle"})
4 |         title = titleElement.get_text()
5 |         company = job.find("a", attrs={"data-automation": "jobCompany"}).get_text()
6 |         joblink = "https://www.seek.co.nz" + titleElement["href"]
7 |         salary = job.find("span", attrs={"data-automation": "jobSalary"})
8 |         salary = salary.get_text() if salary else 'n/a'
9 |
10 |        job = "Titulo: {}\nEmpresa: {}\nSalario: {}\nLink: {}a\n"
11 |
12 |        job = job.format(title, company, salary, joblink)
13 |
14 |        print(job)
```

Guardando las ofertas

Ahora podremos ejecutar nuestro script con el comando:

```
1 | python3 scrapper.py
```

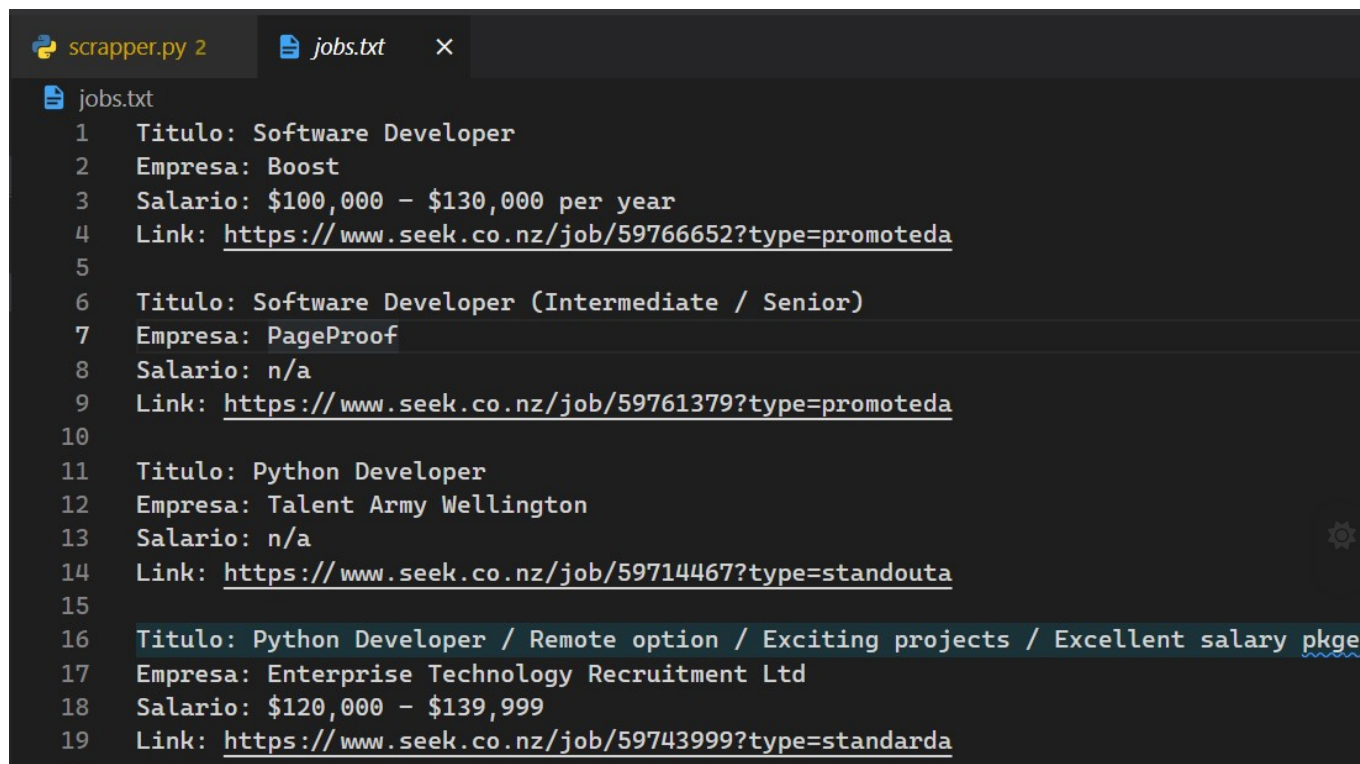
Con esto solo imprimiremos en consola el resultado, pero si queremos guardarlo en algún lugar, podemos hacer esto en la terminal de comandos, haremos uso de las pipes de la siguiente manera:

```
1 | python3 scrapper.py >> jobs.txt
```

Con esto guardaremos este resultado en un archivo que llamamos `jobs.txt`

¡Resultados!

Ha sido un camino largo, pero ahora después de ejecutar el script tenemos en el archivo las ofertas que ha encontrado nuestro scraper, este se ve de la siguiente manera:



```

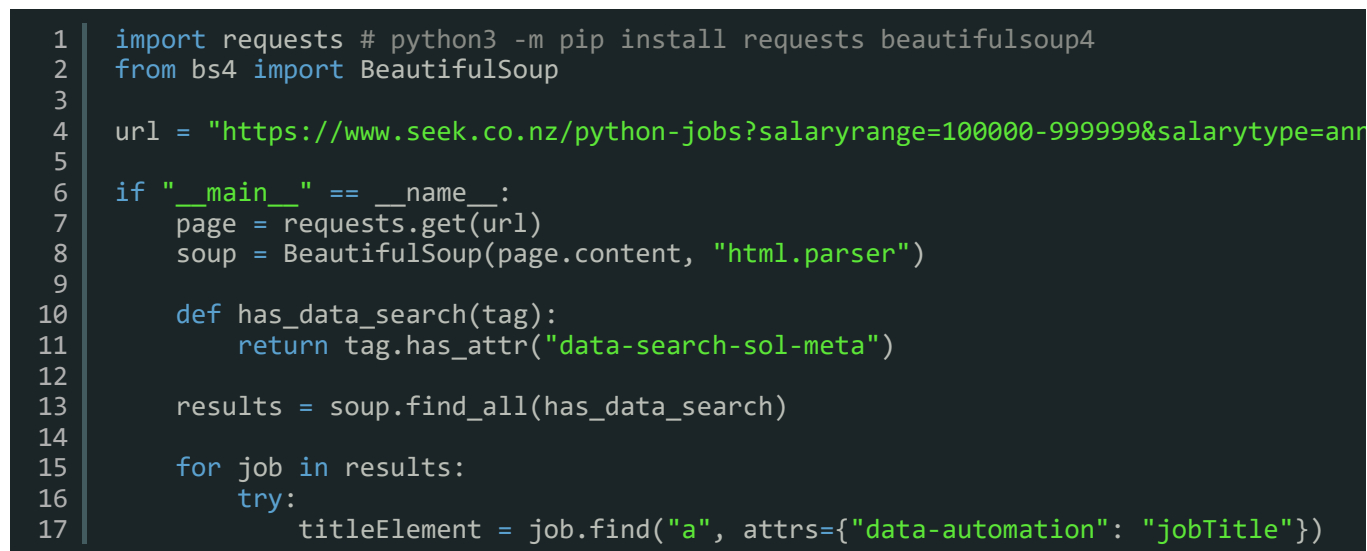
1  Titulo: Software Developer
2  Empresa: Boost
3  Salario: $100,000 - $130,000 per year
4  Link: https://www.seek.co.nz/job/59766652?type=promoteda
5
6  Titulo: Software Developer (Intermediate / Senior)
7  Empresa: PageProof
8  Salario: n/a
9  Link: https://www.seek.co.nz/job/59761379?type=promoteda
10
11 Titulo: Python Developer
12 Empresa: Talent Army Wellington
13 Salario: n/a
14 Link: https://www.seek.co.nz/job/59714467?type=standouta
15
16 Titulo: Python Developer / Remote option / Exciting projects / Excellent salary pkge
17 Empresa: Enterprise Technology Recruitment Ltd
18 Salario: $120,000 - $139,999
19 Link: https://www.seek.co.nz/job/59743999?type=standarda
20

```

Resultado en el archivo jobs.txt

Código y repositorio

Con esto está terminado nuestro script, te dejo el código completo del script así como el [repositorio en GitHub](#) donde puedes ver otros dos scripts con más ideas para automatizar con python, si no sabes qué es o te falta profundizar en la tecnología Git para entender que es un repositorio y cómo tener los tuyos, te recomendamos el curso de [Git: Sin Fronteras](#) de la **Academia de Hola Mundo**.



```

1  import requests # python3 -m pip install requests beautifulsoup4
2  from bs4 import BeautifulSoup
3
4  url = "https://www.seek.co.nz/python-jobs?salaryrange=100000-999999&salarytype=ann"
5
6  if "__main__" == __name__:
7      page = requests.get(url)
8      soup = BeautifulSoup(page.content, "html.parser")
9
10     def has_data_search(tag):
11         return tag.has_attr("data-search-sol-meta")
12
13     results = soup.find_all(has_data_search)
14
15     for job in results:
16         try:
17             titleElement = job.find("a", attrs={"data-automation": "jobTitle"})

```



```
18     title = titleElement.get_text()
19     company = job.find("a", attrs={"data-automation": "jobCompany"}).get_t
20     joblink = "https://www.seek.co.nz" + titleElement["href"]
21     salary = job.find("span", attrs={"data-automation": "jobSalary"})
22     salary = salary.get_text() if salary else 'n/a'
23
24     job = "Titulo: {}\nEmpresa: {}\nSalario: {}\nLink: {}a\n"
25
26     job = job.format(title, company, salary, joblink)
27
28     print(job)
29 except Exception as e:
30     print("Exception: {}".format(e))
31     pass)
```

Esta ha sido la tercera y última idea para automatizar con python, ¿qué más se te ocurre que podemos hacer con python?, déjanos tu opinión en los comentarios.

Y ahora te dejamos aquí los post de las otras dos ideas anteriores para automatizar con python:

- [Compressor](#)
- [Automove File](#)

¿Cómo aprender python?



Puedes crear muchas más herramientas para automatizar tu día a día, pero lo primero es aprender las bases del lenguaje de programación, de nuevo te invitamos a tomar nuestro curso [Python: HTML, CSS, Flask y MySQL](#), pero si prefieres otras tecnologías, te dejamos el [link de la Academia de Hola Mundo para ver todos los cursos](#) que tenemos disponible en diferentes tecnologías para formarte en un excelente desarrollador o desarrolladora.

Para más contenido y no perderte nada suscríbete a este blog, e igualmente síguenos en todas las redes como [youtube](#), [twitter](#) e [Instagram](#). y por último, te invitamos a escuchar nuestra música **"Hola Beats"** diseñada para ayudarte a concentrarte y acompañarte en tu aprendizaje o trabajo, estamos en [Spotify](#) y en [Apple Music](#).

¡Hasta la próxima!, y chao mundo

Share this:



#Tutorial

Python