Code Documentation: webscraping.py

1. Overview

This script is used for web scraping and file downloading. It connects to a specified URL, parses the HTML content, identifies files with specific patterns in their hyperlinks, and downloads them into a local directory.

2. Importing Libraries

The following libraries are used in this script:
- `**requests**`: To send HTTP requests and fetch web content.
- `**BeautifulSoup**`: To parse and extract data from the HTML content.
- `**os**`: To manage file directories and paths.

3. Code Breakdown

3.1 Define URL and Directory

```
url = 'https://soleil.i4ds.ch/solarradio/callistoQuicklooks/?date=20120122'
download_dir = 'downloaded_files'
os.makedirs(download_dir, exist_ok=True)
```

Explanation:

- `**url**`: Specifies the target web page to scrape.
- `**download_dir**`: Name of the directory where downloaded files will be stored.
- `**os.makedirs**`: Ensures the directory exists or creates it if it doesn't.

3.2 Fetch HTML Content

```
response = requests.get(url)
response.raise_for_status()
soup = BeautifulSoup(response.text, 'html.parser')
```

Explanation:

- Sends a GET request to the specified URL using `requests.get()`.
- `response.raise_for_status()` ensures there are no HTTP errors (e.g., 404, 500).
- Parses the HTML content using `BeautifulSoup`.

3.3 Find and Filter Links

```
links = soup.find_all('a', href=True)
for link in links:
    href = link['href']
```

```
    if href.startswith('../data/'):
        # Process the link
```

Explanation:

- Finds all `<a>` tags with `href` attributes using `soup.find_all()`.
- Filters the links where the `href` starts with `'../data/'` (specific path pattern).

3.4 Construct File URL and Download

```
file_url = f'https://soleil.i4ds.ch/solarradio/{href[3:]}'
file_name = href.split('/')[-1]
file_response = requests.get(file_url)
file_response.raise_for_status()
with open(file_path, 'wb') as file:
    file.write(file_response.content)
```

Explanation:

- `**file_url**`: Constructs the full URL for the file.
- `**file_name**`: Extracts the file name from the URL.
- Sends a GET request to download the file content.
- Saves the file locally using binary mode (`wb`).

3.5 Final Output

Once the script runs successfully, all matching files from the specified URL are downloaded into the `downloaded_files` directory, and a message is printed for each file.

4. Summary

This script efficiently automates the process of identifying and downloading specific files from a webpage. It can be customized by:

- Modifying the `url` to target a different web page.
- Adjusting the file path filter logic (e.g., `startswith('../data/')`) for other use cases.