Documentation for the USDA Regulations.gov Comment Downloader Script

---

Purpose of the Script

This script downloads comments from the Regulations.gov API for a specific docket or document and saves:

1. Comment data (ID, text, attachments metadata) into JSON files.
2. Attachments for comments, saving each file with a unique name.

This guide walks through:

- Required installations.
- Configuration steps (e.g., API key, file paths).
- How to use the script step by step.

Setup Instructions

Step 1: Install Required Libraries

The script uses Python 3 and the following libraries:

1. **`requests`**:
    - Handles HTTP requests to the Regulations.gov API.
    - **Installation**: Run:

      ```
      pip install requests
      ```

`os`:

- Provides utilities for file and directory operations.
- Built-in, no installation required.

`json`:

- Handles reading and writing JSON data.
- Built-in, no installation required.

`re`:

- Supports regular expressions for text matching.
- Built-in, no installation required.

`time.sleep`:

- Introduces pauses between requests to avoid overwhelming the API.
- Built-in, no installation required.

Add the API Key:

- Replace YOUR_API_KEY in the script with your actual API key

```
API_KEY = "YOUR_API_KEY"
```

Step 3: Set File Paths

The script saves:

- **JSON Files**: Cleaned comment data (`comment_id`, `text`, and `attachments`).
- **Attachments**: Files linked to comments (e.g., PDFs, images).

Default directories:

- **JSON Files**: `./Downloads/USDA_JSON/`
- **Attachments**: `./Downloads/USDA_JSON/attachments/`

You can update the base directory by modifying:

```
os.makedirs("./Downloads/USDA_JSON/", exist_ok=True)
```

Step 4: Input Document or Docket Link

The script requires a URL from the Regulations.gov website of a docket or document.

How the Script Works

1. Extract Docket or Document ID

- The script parses the provided link using `re` to extract:
  - `documentId` if it's a document link.
  - `docketId` if it's a docket link.

2. Fetch `objectId`

- Uses the API to query the document or docket details.
- Retrieves the `objectId`, a unique identifier required to fetch comments.

3. Fetch Comments

- Fetches comments in batches (default: 250 per page) using the `objectId`.

4. Save Comment Data

- Saves comment data (`comment_id`, `text`, and `attachments` metadata) incrementally to JSON files.

5. Download Attachments

- For each comment:
    - Queries attachment metadata.
    - Downloads files using `fileUrl` links.
    - Saves attachments with unique filenames

Code Flow Explanation

1. Extract IDs

The `extract_docket_id_or_document_id` function uses a regular expression to identify and extract the ID from the provided link.

2. Fetch `objectId`

The `fetch_object_id_from_document` function queries the API for the document's `objectId`.

3. Retrieve Comments

The `get_comments_by_object_id` function queries the API for comments using the `objectId`.

4. Process Each Comment

The `fetch_all_comments` function:

- Iterates through each page of comments.
- Saves comment data (ID, text, attachments).

5. Handle Attachments

The `fetch_attachments` function:

- Queries the API for attachment metadata.
- Downloads files using `download_file`.

- Ensures unique filenames with `attachment_index`.

6. Save Incrementally

The `save_comments_incrementally` function saves comment data to JSON after processing each page.