# oerforge.scan

Static Site Asset and Content Scanner for OERForge

## Overview

oerforge.scan provides functions for scanning site content, extracting assets, and populating the SQLite database with page, section, and file records. It supports Markdown, Jupyter Notebooks, and DOCX files, and maintains hierarchical relationships from the Table of Contents (TOC) YAML. Logging is standardized and all major operations are traced for debugging.

## Functions

### batch\_read\_files

def batch\_read\_files(file\_paths)

Read multiple files and return their contents as a dictionary {path: content}. Supports .md, .ipynb, .docx, and other file types.

**Parameters** - file\_paths (list[str]): List of file paths.

**Returns** - dict: Mapping of file paths to contents.

### read\_markdown\_file

def read\_markdown\_file(path)

Read a Markdown file and return its content as a string.

**Parameters** - path (str): Path to the Markdown file.

**Returns** - str or None: File content or None on error.

### read\_notebook\_file

def read\_notebook\_file(path)

Read a Jupyter notebook file and return its content as a dictionary.

**Parameters** - path (str): Path to the notebook file.

**Returns** - dict or None: Notebook content or None on error.

### read\_docx\_file

def read\_docx\_file(path)

Read a DOCX file and return its text content as a string.

**Parameters** - path (str): Path to the DOCX file.

**Returns** - str or None: Text content or None on error.

### batch\_extract\_assets

def batch\_extract\_assets(contents\_dict, content\_type, \*\*kwargs)

Extract assets from multiple file contents in one pass. Returns a dictionary {path: [asset\_records]}.

**Parameters** - contents\_dict (dict): Mapping of file paths to contents. - content\_type (str): Type of content (‘markdown’, ‘notebook’, ‘docx’, etc.). - \*\*kwargs: Additional arguments for DB connection/cursor.

**Returns** - dict: Mapping of file paths to lists of asset records.

### extract\_linked\_files\_from\_markdown\_content

def extract\_linked\_files\_from\_markdown\_content(md\_text, page\_id=None)

Extract asset links from Markdown text.

**Parameters** - md\_text (str): Markdown content. - page\_id (optional): Page identifier for DB linking.

**Returns** - list[dict]: File record dicts for each asset found.

### extract\_linked\_files\_from\_notebook\_cell\_content

def extract\_linked\_files\_from\_notebook\_cell\_content(cell, nb\_path=None)

Extract asset links from a notebook cell.

**Parameters** - cell (dict): Notebook cell. - nb\_path (str, optional): Notebook file path.

**Returns** - list[dict]: File record dicts for each asset found.

### extract\_linked\_files\_from\_docx\_content

def extract\_linked\_files\_from\_docx\_content(docx\_path, page\_id=None)

Extract asset links from a DOCX file.

**Parameters** - docx\_path (str): Path to DOCX file. - page\_id (optional): Page identifier for DB linking.

**Returns** - list[dict]: File record dicts for each asset found.

### populate\_site\_info\_from\_config

def populate\_site\_info\_from\_config(config\_filename='\_config.yml')

Populate the site\_info table from the given config file.

**Parameters** - config\_filename (str): Name of the config file (default: ’\_config.yml’).

### get\_conversion\_flags

def get\_conversion\_flags(extension)

Get conversion capability flags for a given file extension using the database.

**Parameters** - extension (str): File extension (e.g., ‘.md’, ‘.ipynb’).

**Returns** - dict: Conversion capability flags.

### scan\_toc\_and\_populate\_db

def scan\_toc\_and\_populate\_db(config\_path)

Walk the TOC from the config YAML, read each file, extract assets/images, and populate the DB with content and asset records. Maintains TOC hierarchy and section relationships.

**Parameters** - config\_path (str): Path to the config YAML file.

### get\_descendants\_for\_parent

def get\_descendants\_for\_parent(parent\_output\_path, db\_path)

Query all children, grandchildren, and deeper descendants for a given parent section using a recursive CTE.

**Parameters** - parent\_output\_path (str): Output path of the parent section. - db\_path (str): Path to the SQLite database.

**Returns** - list[dict]: Dicts for each descendant (id, title, output\_path, parent\_output\_path, slug, level).

## Usage Example

from oerforge import scan
scan.scan\_toc\_and\_populate\_db('\_content.yml')

## Requirements

* Python 3.7+
* SQLite3
* PyYAML
* python-docx

## See Also

* [Python-Markdown](https://python-markdown.github.io/)
* [Jupyter Notebook Format](https://nbformat.readthedocs.io/en/latest/)
* [python-docx Documentation](https://python-docx.readthedocs.io/en/latest/)

## License

See LICENSE in the project root.