# oerforge.verify — Accessibility Verification and Reporting

## Overview

oerforge.verify provides tools for automated accessibility checking, badge generation, and reporting for static HTML sites. It integrates with [Pa11y](https://pa11y.org/) for WCAG compliance checks, manages results in a SQLite database, and generates visual badges and detailed reports for each page. This module is designed for new users, educators, and developers seeking robust accessibility workflows in open educational resources.

* **Automated accessibility checks (WCAG AA/AAA) using Pa11y**
* **Badge and report generation for each HTML page**
* **Database integration for tracking results**
* **Jinja2-based report rendering**
* **Robust logging and error handling**

## Functions

### load\_pa11y\_config(yml\_path=’\_content.yml’)

Load Pa11y configuration and logo info from YAML and JSON files.

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

**Returns** - dict: Parsed configuration dictionary.

### run\_pa11y\_on\_file(html\_path, config\_path=None, wcag\_level=‘AA’)

Run Pa11y accessibility checks on a single HTML file.

**Parameters** - html\_path (str): Path to the HTML file. - config\_path (str, optional): Path to Pa11y config file. - wcag\_level (str): WCAG level (‘AA’ or ‘AAA’).

**Returns** - list[dict] or None: Parsed Pa11y JSON results, or None on error.

**Notes** - Handles errors robustly, logs issues, and attempts to parse output even on failure.

### get\_content\_id\_for\_file(html\_path, conn)

Get the content ID for a given HTML file from the database.

**Parameters** - html\_path (str): Path to the HTML file. - conn (sqlite3.Connection): Database connection.

**Returns** - int or None: Content ID if found, else None.

### store\_accessibility\_result(content\_id, pa11y\_json, badge\_html, wcag\_level, error\_count, warning\_count, notice\_count, conn=None)

Store the latest accessibility result for a page in the database.

**Parameters** - content\_id (int): Content ID for the page. - pa11y\_json (list[dict]): Pa11y results. - badge\_html (str): Badge HTML markup. - wcag\_level (str): WCAG level. - error\_count, warning\_count, notice\_count (int): Issue counts. - conn (sqlite3.Connection): Database connection.

### generate\_badge\_html(wcag\_level, error\_count, logo\_info, report\_link)

Generate badge HTML for a given WCAG level and error count.

**Parameters** - wcag\_level (str): WCAG level. - error\_count (int): Number of errors. - logo\_info (dict): Mapping of WCAG levels to badge/logo URLs. - report\_link (str): Link to the accessibility report.

**Returns** - str: Badge HTML markup.

### inject\_badge\_into\_html(html\_path, badge\_html, report\_link, logo\_info)

Inject the badge/button into the HTML file after <main>.

**Parameters** - html\_path (str): Path to the HTML file. - badge\_html (str): Badge HTML markup. - report\_link (str): Link to the accessibility report. - logo\_info (dict): Badge/logo info.

### generate\_nav\_menu(context)

Generate top-level navigation menu items from the content database.

**Parameters** - context (dict): Context dict, should include ‘rel\_path’.

**Returns** - list[dict]: List of menu item dicts: {‘title’: str, ‘link’: str}

### generate\_wcag\_report(html\_path, issues, badge\_html, config)

Generate a detailed HTML accessibility report for a file using Jinja2 templates.

**Parameters** - html\_path (str): Path to the HTML file. - issues (list[dict]): Pa11y issues. - badge\_html (str): Badge HTML markup. - config (dict): Page and site config.

**Returns** - str: Path to the generated report file.

### process\_all\_html\_files(build\_dir=‘build’, config\_file=None, db\_path=‘db/sqlite.db’)

Process all HTML files in the build directory: - Run Pa11y checks - Store results in DB - Generate badges and reports - Inject badges into HTML - Copy changed files to docs/

**Parameters** - build\_dir (str): Build directory. - config\_file (str, optional): Pa11y config file. - db\_path (str): Path to SQLite database.

### copy\_to\_docs()

Copy all changed files from build/ to docs/.

### main()

CLI entry point. Parses arguments, runs checks, stores results, and generates reports as needed.

## Logging

All major operations and errors are logged to log/pa11y.log for debugging and auditing.

## Error Handling

Robust error handling is implemented for subprocess calls, file I/O, database operations, and template rendering. All failures are logged with context.

## Example Usage

from oerforge import verify  
verify.process\_all\_html\_files()

## See Also

* [Pa11y Documentation](https://pa11y.org/)
* [WCAG Guidelines](https://www.w3.org/WAI/standards-guidelines/wcag/)
* [Jinja2 Templates](https://jinja.palletsprojects.com/)

## License

See LICENSE in the project root.