

Commands Cheat Sheet

Quick reference for the repository wrapper commands (run from repo root once `src/scripts/bin` is on your PATH).

Pack / Unpack

- `unpack <source-file> <dest-folder>`: Expand a packed Office file into an unpacked folder.
- `pack <expanded-folder> <output-file>`: Pack an expanded folder back into an Office file.

Create

- `create <type> <output-file>`: Create a new OpenXML file (examples: `word-template`, `xl-template`).
- `create-snippet-styles <expanded-folder> <snippet-id> <style-id,style-id>`: Create a styles snippet from an expanded template.

Create Types

- `xl-template` : Excel template (.xltx)
- `xl-mtemplate` : Excel macro-enabled template (.xltn)
- `xl-book` : Excel workbook (.xlsx)
- `xl-mbook` : Excel macro-enabled workbook (.xlsm)
- `word-template` : Word template (.dotx)
- `word-mtemplate` : Word macro-enabled template (.dotm)
- `word-doc` : Word document (.docx)
- `word-mdoc` : Word macro-enabled document (.docm)

Style utilities

- `style-generate-list <expandedPath>`: Extract and save style list from an unpacked template to its `docs/` folder.
- `style-import-map <target-styles.xml> [--mapping <path>] [--dry-run] [--backup]`: Import ordered styles into a `styles.xml` using a mapping YAML (prefers `docs/style-map.yml`, falls back to `style-mapping.yml`).
- `style-import-partial <target-styles.xml> <snippet-xml> [--dry-run] [--backup]`: Replace styles in `styles.xml` from a single snippet XML (match by styleId).
- `style-import-doc <target.docx> <source.docx> [--dry-run] [--backup]`: Replace the entire `styles` part in a docx using the Open XML SDK (reliable package-level operation).

Cleanup / Fixup

- `remove-tracking <target-file>`: Remove RSIDs, ParalId, textId from Word XML.
- `remove-noproof <target-file>`: Remove `noProof` attributes.
- `remove-styles-linkchar <path-to-styles.xml>`: Remove linked character styles.

Validation & Utilities

- `validate <file>`: Run OOXML validation helpers.
- `xpathsel <xml-file> <xpath>`: Run a quick XPath selection against an XML file.
- `inventory generate`: Generate the template inventory (see `inventory --help`).

Manifests

- `manifest <action> <resource>`: Unified manifest dispatcher (see `manifest --help`).
- `manifest-generate <agency>`: Generate/update an agency manifest.
- `manifest-generate-core`: Generate/update core manifest.
- `manifest-generate-partials-builds`: Generate partials manifest for `builds/`.
- `manifest-generate-partials-core`: Generate partials manifest for `core/`.
- `manifest-generate-sync`: Auto-create per-agency manifest-generator wrappers.
- `manifest-generate-jbc`, `manifest-generate-osa`, `manifest-generate-lcs`, `manifest-generate-olls`: Per-agency manifest generators (wrappers).
- `manifest-list`: List templates across agencies.
- `manifest-update-status <agency> <template> <status>`: Update template status in an agency manifest.
- `manifest-validate`: Validate manifest files.

Tips

- Use `--dry-run` where available to preview changes without writing files.
- Backups are opt-in with `--backup` for import/write commands (unless noted otherwise).
- Mapping files for `style-import-map` are in `docs/style-map.yml` next to the template.