Kit de documentation

Documentation — Dialogue Commands — EN

1 Introduction

This document is a translation in substance. The reference is the original document in French; extensions and changes are always made there.

A language *model* applies the rules it is given, in the order they were given, with the priority they carry. It does not arbitrate between contradictory instructions — it accumulates them. This is not ill will, it is logic.

It is also built to be helpful and accommodating. That combination sets a classic trap: the user makes a request that contradicts a Kit rule, the *AI* obliges, and the structure degrades silently. This is precisely why the Kit rules must be laid down before any request to produce something, and why cases the rules do not cover must be treated as evolutions of the Kit, never as one-off exemptions.

This document gathers the commands to paste into the *chat* to keep the session inside the Kit's frame. Four situations are covered: starting a production session, resetting a session that has drifted, authorising a one-off surgical modification, and the pre-check before any document is created or modified.

The blocks in this document are given in English. The French and German variants carry the same blocks in their own language, and say the same thing — use the one that matches the working language of the session.

Formatting convention for the blocks: the title is in capitals, followed by a separating line of thirty equals signs. Internal sections are separated by a line of dashes. Enumerations use dashed items, with continuations aligned under the text. Stop conditions are introduced by an ASCII arrow. Each new idea is separated by a blank line holding a *non-breaking space* — without it, the web converter removes the line.

2 Dialogue commands

2.1 Starting a production session

Use at the start of any session involving the *generation* or modification of a document.

```
Hello. New production session on the Kit.
Before any generation, here is the frame to work within.

ABSOLUTE RULE — ZERO FORMATTING FROM MEMORY
==========================

MANDATORY READING BEFORE ANY GENERATION :
------------------------------

1. READ Kit - Projet - Prompt in full.
2. READ the active stylesheet .js in full
   (General / Glossary / YAML / HTML as required).

3. READ the matching Reference .docx (same timestamp).
   §9.1 = makeTable row format.
   §9.3 = indentation constants and column widths.
   §13  = return contracts, spread, predecessors, successors.
```

```
4. READ the current project prompt.
5. READ Kit - Documentation - Quality Control §3.13 and §3.14
   — absolute call chain and environment prerequisites.

— Reference .docx — NEVER rebuilt from the .js alone.
  The .js holds the code. The .docx holds the application
  recipe: contracts, predecessors, successors, anti-patterns.
  These cannot be inferred from the .js.
  Without the source file: full stop.

No numeric value, function signature, colour or indentation
constant may be used without having been verified in the .js
read during this session.

VALIDATION CHAIN — ABSOLUTE ORDER :
------------------------------

0. PREREQUISITES :
   node -e "console.log(Object.keys(require('docx')).length)"
   -> must return more than 200.
   node -e "require('adm-zip')" — before a site publication
   only ; outside the document chain since General 1.83.

1. BEFORE any generation :
   python kit_check_registry.py
   -> exit 0 required.

2. BEFORE each node gen-*.js :
   python kit_check_setters.py gen-document.js
   -> exit 0 required.

3. AFTER .docx generation :
   node kit_validate_docx.js document.docx --version {version}

4. AFTER .xlsx generation :
   python kit_validate_xlsx.py document.xlsx

5. AFTER regenerating an existing document :
   python kit_check_markup.py source.docx produced.docx
   -> a text diff does not see a markup loss.

6. AFTER regenerating a language variant :
   python kit_check_ossature.py reference.docx variant.docx

7. AFTER a change to the extraction :
   python kit_check_fidelite.py over the corpus, --version {version}

8. BEFORE any delivery touching a member of a couplet :
   python kit_check_couplets.py

No file is delivered until the full chain has passed.

CHECKS BEFORE RUNNING THE SCRIPT :
------------------------------

— The level of a body element comes from the current
  heading. h1, h2 and h3 set it, the others read it.
```

```
  No suffix to check since General 1.70.

- Document section :
  properties: { ...style.pageProps, titlePage: true }
  Without this flag, Word ignores first-page headers and
  footers and the cover page loses its bare layout.

- makeTable receives cells in the correct format: a plain
  string, or a pair of text and italic flag. Never a numeric
  width inside a cell.

- Conditional loading of data modules according to
  [Prefix] - Registry.requires. A false flag means the file is
  absent: no require, no setter. A true flag means load and
  call the matching setter.

- style.getLuxTimestamp() for the timestamp. Never reuse an
  existing timestamp.

- Consult the contracts section of the Reference before any
  function call (General §13 / YAML §8 / HTML §13).

- makeImageRun(buffer, width, height, format) for any image —
  format among png, jpg, gif, bmp, svg, mandatory.

- File names without any underscore, except kit_* tools.
- Pair integrity: modifying one file of a pair requires the
  simultaneous update of its partner, even for a cosmetic bump.

- Pre-production check: ask explicitly what changes or is added.
  Confirm the exact file name before writing the first line.

STOP CONDITIONS :
-----------------------------

If any anomaly, inconsistency or ambiguity is detected at any
stage :

-> Full stop.
-> Report clearly and wait for explicit confirmation.
-> Absence of an answer is a stop signal.
-> Never report and continue in the same sentence.

If a formatting case is not covered by the stylesheet :

-> Report it as a Kit evolution need. Do not improvise.

If the document to modify already exists :

-> Request the .docx binary in the chat before starting.
-> The working tree file is plain text, not a binary.
-> Modify only what is explicitly requested.
```

2.2 Hard reset — drifting session

Use when the session has drifted beyond what spot corrections can fix. Open a new *chat* and paste this as the first message.

```
The session has drifted. We go back to the sources and
reconstruct nothing from memory.

HARD RESET — BACK TO SOURCES
============================

CONTEXT :
-----------------------------

A new chat is mandatory when the session has grown too large.
Purge previous chats before starting over.

TECHNICAL NOTE :
-----------------------------

- File names shown with underscores in the working tree are
  a platform transformation. The canonical name uses spaces.

- The .docx files in the working tree are not Word binaries.
  They are text extractions produced by the platform. Always
  request the binary in the chat for any modification.

SESSION START — MANDATORY ORDER :
-----------------------------

1. Read Kit - Projet - Prompt in full.
2. Read the stylesheets .js and their Reference .docx.
3. Read the current project prompt.
4. Read every document not yet gone through in detail.
5. Purge anything that contradicts those documents.
6. Purge anything that could lead to an unsolicited initiative
   or to any shortcut.

7. Report what was eliminated, then what remains in memory.

PRODUCTION RULES :
-----------------------------

- Formatting exclusively through the active stylesheet. Never
  mix stylesheets in one document. Never format from memory.

- Generator script rebuilt from scratch every session.
- Conditional loading of data modules according to
  [Prefix] - Registry.requires.

- The .docx binary is requested in the chat before any
  modification. Modify only what is explicitly requested. No
  unauthorised loss of content.

- Work only from the most recent export provided in the
  session. Never infer or rebuild from an already generated
  document or from a previous session.

- Pair integrity: modifying one file requires the simultaneous
  update of its partner.

- Report any inconsistency before launching generation, and
```

```
  wait for the explicit go.

- Check that the validator points at the correct file version.
- Strict naming: spaces and hyphens only, never underscores.
  A fresh timestamp on every delivery, without exception.

- Never set the level other than by emitting a heading:
  h1, h2 and h3 are the only functions that write it.

MANDATORY PAIRS :
-----------------------------

- Stylesheet .js and its Reference .docx, same timestamp.
- Kit Cover Sheet .docx and .png, same timestamp.
- Project Cover Sheet .docx and .png, same timestamp,
  independent from the Kit one.

- Terms module and glossary .docx, same timestamp. A
  multilingual project produces one glossary per published
  language, all sharing that timestamp.

- Cover Sheet: every assembly instruction for the image lives
  in the matching .docx. Rendering by named constants only,
  never free composition.
```

2.3 Tightening — gradual drift

Today's *artificial intelligence* engines tend to favour what pleases the user over what was asked of them. The drift is slow and raises no alert: answers grow longer, unrequested analysis creeps in, and rules that are written down stop being reread. The engine therefore has to be tightened regularly, without waiting for the session to become unusable.

To be used as soon as answers grow longer or stray from the request. Unlike the previous block, this tightening does not call for a new *chat*: it is pasted into the session in progress.

```
TIGHTENING
==============================

Reread Kit Prompt §1 (checklist) and §2 (tone). Apply them
from now on.

BEFORE EVERY ANSWER :
-----------------------------

- Answer what was asked. Nothing else.
- No unrequested analysis, no summary of what I have just
  said, no lecture about the rules.
- A formatting value is read in the .js and the Reference of
  this session, never from memory.
- If you do not know, say so in one sentence.

Confirm in one line, then carry on.
```

2.4 Authorising a surgical modification

Use only when a targeted change to an existing document is needed and a full regeneration would be disproportionate.

```
Precise request: modify an existing .docx by surgical patch,
without regenerating it. Here are the conditions.

RULE — DIRECT XML MODIFICATION ON AN EXISTING .docx
============================

REQUIRED CONDITIONS (all at once) :
------------------------------

- The target .docx was generated by a script using the active
  stylesheet. Never on a file of unknown origin.

- The generator script is no longer available in the session.
  If it is, XML editing is forbidden: regenerate instead.

- Every modified value is taken from the active stylesheet .js
  read in the current session. No numeric value from memory.

- The change is surgical: only what is explicitly requested is
  touched. No rewriting, rephrasing or unsolicited addition.

- No content loss on an existing document. Everything present
  in the source file is fully preserved unless explicitly
  instructed otherwise.

- Before any modification: state precisely which nodes will be
  touched and wait for explicit confirmation.

- Regex is forbidden for writing to a structured file. Every
  modification goes through the format's native parser. Regex
  for pure reading remains allowed.

- To insert a note or a callout, use the helpers of
  kit_patch_helpers.py — never clone a fragment of the target
  document.

- After the change: validate with kit_validate_docx.js and
  report any warning before delivering.

STOP CONDITIONS :
------------------------------

If any of these conditions is not met :

-> Full stop.
-> Report the anomaly clearly.
-> Wait for explicit confirmation before continuing.
-> Never report and continue in the same sentence.
```

2.5 Pre-check — document creation or modification

Paste before any document creation or modification. This pre-check has the sources of truth declared as read, the source file as inventoried, and every content change as approved, before the script is touched.

```
Before creating or modifying a document, here is the check
to run point by point.

PRE-CHECK — DOCUMENT CREATION / MODIFICATION
============================

MANDATORY READING BEFORE ANY ACTION :
----------------------------

1. READ Kit - Projet - Prompt in full.
2. READ the active stylesheet .js in full.
3. READ the matching Reference .docx (same timestamp).
   This .docx is the application recipe and takes precedence
   over any inference from the code.
   §9.1 = makeTable row format.
   §9.3 = constants and column widths.
   §13  = return contracts, spread, predecessors, successors.

4. READ the current project prompt.
5. READ Kit - Documentation - Quality Control §3.13 and §3.14.

– Reference .docx — NEVER rebuilt from the .js alone.
  Without the source file: full stop.

READING REPORT — ANSWER EVERY POINT :
----------------------------

– Kit Projet Prompt read in full: yes / no.
– Stylesheet .js read in full: yes / no, and version.
– Reference .docx read in full: yes / no, and timestamp.
– Project prompt read: yes / no.
– Quality Control §3.13 and §3.14 read: yes / no.

-> If any point is "no": full stop. Read before continuing.

IF MODIFYING AN EXISTING DOCUMENT :
----------------------------

– Request the .docx binary in the chat if not already provided.
– Pair integrity: if the modified file belongs to a pair, the
  partner must be assessed AND updated in the same session.
  Partner source file missing: full stop.

– Inventory the source file: paragraphs, tables, images.
  Record the exact counts before writing the script.

– Extract the images from the binary before any script. No
  image omitted without explicit instruction.

– The script is rebuilt from scratch from the binary content.
  Never from memory or from a previous script.
```

```
- List every planned content change against the source
  version. Wait for explicit approval before writing.

VALIDATION CHAIN :
------------------------------

0. PREREQUISITES: docx must resolve ; adm-zip only before a
   site publication.
1. BEFORE any generation : python kit_check_registry.py.
2. BEFORE node gen-*.js :
   python kit_check_setters.py gen-document.js
   -> exit 0 required.

3. AFTER .docx generation :
   node kit_validate_docx.js document.docx --version {version}

4. AFTER .xlsx generation :
   python kit_validate_xlsx.py document.xlsx

5. AFTER regenerating an existing document :
   python kit_check_markup.py source.docx produced.docx
   -> a text diff does not see a markup loss.

6. AFTER regenerating a language variant :
   python kit_check_ossature.py reference.docx variant.docx

7. AFTER a change to the extraction :
   python kit_check_fidelite.py over the corpus, --version {version}

8. BEFORE any delivery touching a member of a couplet :
   python kit_check_couplets.py

CHECK BEFORE FINAL DELIVERY :
------------------------------

Compare the produced document with the source inventory :

- Number of sections and headings: identical ?
- Number of tables: identical ?
- Number of images: identical ?
- Content of each section: nothing lost ?
- Image captions: present and accurate ?

-> If a gap is found: report, correct, re-check.
-> Deliver only after explicit zero-loss confirmation.

FORMATTING RULES :
------------------------------

- Zero formatting from memory. Every value verified in the .js
  or the Reference read in this session.

- makeTable: rows are plain strings or text-and-italic pairs.
  Never cell objects inside rows.

- Column widths: sum equals the level width exactly. Number of
```

```
  widths equals number of columns.

- Element level consistent with the parent heading.
- Section: properties with titlePage set to true.

NOTE AND CALLOUT RULES :
-----------------------------

- Note: only if the information is important, non-obvious and
  absent from the body text. Never directly under a heading,
  never without a preceding paragraph. When in doubt -> plain
  paragraph.

- Callout: only if the advice brings what the reader cannot
  find in the running text. If it brings nothing new ->
  remove it.

STOP CONDITIONS :
-----------------------------

If any anomaly, inconsistency or ambiguity is detected :

-> Full stop.
-> Report clearly and wait for explicit confirmation.
-> Never report and continue in the same sentence.

If a formatting case is not covered by the stylesheet :

-> Report it as a Kit evolution need. Do not improvise.
```