Kit de documentation

Dokumentation — HTML-Pipeline — DE

1 Einleitung

Dieses Dokument ist eine sinngemäße Übersetzung. Maßgeblich ist das Originaldokument auf Französisch; Erweiterungen und Änderungen werden stets dort eingepflegt.

Die Website veröffentlicht einen Teil der Kit-Dokumente, nicht alle: Nur die in der *Pipeline* angemeldeten werden in Seiten umgewandelt. Die übrigen bleiben im gedruckten *Ordner* und erscheinen nirgends online, auch nicht nach einer kürzlichen Aktualisierung.

Die Umwandlung geht von den Word-Dateien (.docx) aus, die das *Stylesheet* der Prosadokumente erzeugt, entnimmt ihnen die Struktur und gibt sie als HTML-Seiten mit Navigation, Glossar und *Startseite* wieder. Jede Veröffentlichung erzeugt ein Archiv, das zur Übertragung bereitsteht.

Die folgende Tabelle zeigt, was in jedem Projekt vollständig gilt und was vom Projekt abhängt und bei der Einrichtung festzulegen ist.

Geltung	Abschnitt	Projektwert
INVARIANT	§2.1 Auswahlkriterien der veröffentlichten Dokumente	—
INVARIANT	§2.3 Lieferregeln — Werkzeuge, Delta-Logik	—
INVARIANT	§3 *Registry.js* — Aufbau, Verwendung, Aktualisierungsregel	—
INVARIANT	§4 *Delta-Deployment* als ZIP — Logik, Benennung, lastDeploy	—
INVARIANT	§5 Aufbau der HTML-Website	—
INVARIANT	§6 PDF-*Erzeugung*	—
INVARIANT	§7 Robustheitsregeln	—
PROJEKT	§2.2 Rubriken und veröffentlichte Dokumente	Liste der Projektdokumente, HTML-Slugs, PDF-Aktivierung. In *Registry.js* im Abschnitt documents festlegen.
PROJEKT	§4.3 Benennung der ZIP-Dateien	Die Domain des Kits durch die der *Subsite* des Projekts ersetzen.

2 Veröffentlichungsumfang

Nicht alle Kit-Dokumente werden auf der Website veröffentlicht. Die Auswahl ist bewusst auf Dokumente beschränkt, die ein aktiver Kit-Nutzer lesen soll. Dateien der internen Konfiguration — Prompts, Stylesheets, *PCL*-Konstanten — sind von der direkten Veröffentlichung ausgeschlossen: Sie bleiben zur Information aufgeführt.

2.1 Auswahlkriterien

- Das Dokument hat für einen Kit-Nutzer einen eigenständigen Lesewert.
- Das Dokument legt keine interne Konfiguration des Kits offen.
- Das Dokument ist stabil — kein Entwurf und kein Arbeitsdokument.

2.2 Rubriken und veröffentlichte Dokumente

Die Website ist in nummerierte Rubriken gegliedert. Beim Kit übernimmt der *Papierordner* dieselben Rubriken, mit denselben Titeln und denselben Dokumenten — siehe §2.5. Die Stylesheets fehlen dort: Ihre Dokumente finden sich im herunterladbaren Archiv.

#	Titel	Dokumente	HTML-Datei	PDF
01	Bevor Sie anfangen	Kit — Documentation — Comment documenter ses projets	comment-documenter-LANG.html	ja
		Kit — Documentation — Reading Guide	reading-guide-LANG.html	ja
02	Arbeiten mit **Claude AI**	**Claude AI** — Documentation — Manuel	claude-manuel-LANG.html	ja
		Claude AI — Documentation — Guide Pratique	claude-guide-pratique-LANG.html	ja
		Claude AI — Documentation — Environnement	claude-environnement-LANG.html	ja
03	Arbeiten mit *Grok AI*	*Grok AI* — Documentation — Manuel	grok-manuel-LANG.html	ja

#	Titel	Dokumente	HTML-Datei	PDF
		Grok AI — Documentation — Guide Pratique	grok-guide-pratique-LANG.html	ja
		Grok AI — Documentation — Environnement	grok-environnement-LANG.html	ja
		Grok AI — Documentation — Prompts de dialogue	grok-prompts-de-dialogue-LANG.html	ja
04	Arbeiten mit **ChatGPT AI**	**ChatGPT AI** — Documentation — Guide Pratique	chatgpt-guide-pratique-LANG.html	ja
		ChatGPT AI — Documentation — Environnement	chatgpt-environnement-LANG.html	ja
05	Ein Projekt starten und pflegen	Kit — Projet — Guide d'initialisation	guide-initialisation-LANG.html	ja
		Kit — Projet — Guide de mise à jour	guide-maj-LANG.html	ja
		Kit — Projet — Convention de nommage	convention-nommage.html	ja
06	Die *künstliche Intell igenz* führen und das Kit anpassen	Kit — Documentation — Prompts de dialogue	prompts-de-dialogue-LANG.html	ja
		Kit — Documentation — Étendre le Kit	etendre-le-kit-LANG.html	ja
07	Dateien und Struktur	Kit — Documentation — Environnement d'exécution	environnement-execution-LANG.html	ja

#	Titel	Dokumente	HTML-Datei	PDF
		Kit — Projet — *Prompt*	kit-prompt-LANG.html	ja
		Kit — Projet — Structure commune	structure-commune-LANG.html	ja
		Kit — Documentation — *Pipeline* HTML	pipeline-html-LANG.html	ja
		Kit — Documentation — Quality Control	quality-control-LANG.html	ja
		Kit — Glossaire — Termes	glossaire-LANG.html	ja
08	Ein Projekt veröffentlichen und teilen	Kit — Documentation — Publication et reprise	publication-reprise-LANG.html	ja

Hinweis: *LANG steht für das Sprachkürzel: comment-documenter-fr.html, comment-documenter-de.html, comment-documenter-en.html. Ein nicht übersetztes Dokument trägt einen Slug ohne Kürzel.*

2.3 Lieferregeln

Jede *Erzeugung* liefert entweder einen vollständigen Durchlauf — erstes Deployment oder lastDeploy null — oder einen *Delta-Durchlauf* für alle folgenden Deployments. Die Logik wird vom *Registry* gesteuert. Einzelheiten in §4.

Erzeugte Datei	Werkzeug	Vollständiger Durchlauf	Delta
html/*.html — Dokumentseiten	**Node.js**, *AST pandoc*, HTML-*Stylesheet*	Alle	Nur die seit lastDeploy geänderten .docx
glossaire.html	Kit-Helper kit_gen_glossaire_html.js aus dem Begriffsmodul, nie über den *AST*	Immer	Immer

Erzeugte Datei	Werkzeug	Vollständiger Durchlauf	Delta
index.html und index-{langue}.html — Lader	**Node.js**, pageChargeur()	Immer	Immer
assets/kit-style.css	**Node.js**, getCSS()	Immer	Wenn sich das HTML-*Stylesheet* seit lastDeploy geändert hat
assets/pdf/*.pdf	*LibreOffice* CLI aus der binären .docx	Wenn das PDF aktiviert ist	Mit der zugehörigen .html
assets/pieces/* — Anlagenstücke	Direkte Kopie aus dem Wurzelverzeichnis des Projekts, über den Schlüssel piece erklärt	Wenn erklärt	Das Merkblatt erzeugt sein eigenes PDF wie jedes Dokument
assets/png/{slug}/*.png	Von *pandoc* extrahiert	Mit der .html	Mit der zugehörigen .html
.htaccess	Nicht erzeugt — von der übergeordneten Website abgelegt	Außerhalb des Umfangs	Außerhalb des Umfangs
robots.txt	Nicht erzeugt — von der übergeordneten Website abgelegt	Außerhalb des Umfangs	Außerhalb des Umfangs

Hinweis: *Der Generator findet jede Quelldatei (.docx) über eine Teilübereinstimmung mit dem Dateinamen. Das Muster muss eindeutig sein. Apostrophe in Dateinamen sind immer gerade Apostrophe, nie typografische. Ausführliche Regeln: HTML Reference §14.10.*

Ein in mehreren Sprachen veröffentlichtes Dokument erzeugt eine Seite je Variante, jede mit ihrem Slug samt Sprachkürzel und ihrem eigenen Eintrag in Registry.documents. Das Delta vergleicht je Slug, daher wird jede Sprache unabhängig von den anderen neu veröffentlicht. Wo der Sprachwähler erscheint, richtet sich nach dem im *Registry* unter rendering.languageSelector erklärten Modus — Gemeinsame Struktur §15.

2.4 Startseite — Einleitungsblock

Die *Landing Page* zeigt über der Liste der Rubriken einen Zitatblock. Dieser Text ist die offizielle Wahrheitsquelle — der Index-Generator übernimmt ihn wörtlich in seine dafür vorgesehene Konstante.

```
Ce site vous aide à vous servir du kit de documentation pour documenter vos projets. Une fois appliqué, le kit produit des documents au format Word, des PDF et des pages web. Il sert de mode opératoire à un moteur d'intelligence artificielle, celui de votre choix.
```

Der Text wird dem Landing-Generator über das vorgesehene Konfigurationsfeld übergeben. Ist das Feld leer oder fehlt es, wird der Block automatisch ausgeblendet — kein verwaister orangefarbener Balken.

Dieselbe Regel gilt für *Anwenderprojekte*: Das Dokument *Pipeline HTML* jedes Projekts legt in §2.4 seinen eigenen Text fest, den sein Website-Generator wörtlich übernimmt.

2.5 Cover Sheet auf Papier und HTML-Landing Page — zwei unabhängige Verzeichnisse

Die beiden Verzeichnisse — §2 des Dokuments *Cover Sheet* und die Rubrikenkonstante des Website-Generators — werden getrennt deklariert. Kein Skript liest das eine, um das andere zu steuern.

Ihre Übereinstimmung ist eine Entscheidung des Projekts. Das Kit hält sie identisch: dieselben Rubriken, dieselben Dokumente in derselben Lesereihenfolge, wobei der *Ordner* nur die Basissprache trägt. Ein Projekt kann auch einen verschlankten *Ordner* zusammenstellen, der nur einen Teil dessen übernimmt, was die Website veröffentlicht.

Die HTML-*Landing Page* bleibt das Web-Verzeichnis: Sie ergänzt jeden Eintrag um Veröffentlichungsmetadaten, und was sich nicht online liest, fehlt dort — Stylesheets, *Registry*, *Cover Sheet*, *Datenmodule*.

Das gemeinsame Erscheinungsbild — Farben der Rubriken, Grundgedanke, Lesehilfe — bleibt in jedem Fall erhalten.

Hinweis: *Eine Abweichung zwischen den beiden Verzeichnissen ist keine Drift, die ein Skript korrigieren müsste. Die stillschweigende Synchronisierung aufgrund einer vermuteten Drift ist in Quality Control §6 als Anti-Pattern katalogisiert; beim Kit wird die Übereinstimmung beim Lesen geprüft.*

Der Website-Generator folgt dem Muster gen-{projet}-site.js, einer je Projekt. Das Kit hat seinen eigenen, getrennt von denen der *Anwenderprojekte*. Er ist kein wiederverwendbares *stabiles Artefakt* des Kits: Er ist seiner Natur nach spezifisch für das Projekt, das er veröffentlicht.

Hinweis: *Der Website-Generator des Kits bietet keine öffentliche API. Seine internen Helper und Konstanten sind an die Struktur des Kits gebunden und werden nicht exportiert. Anwenderprojekte schreiben ihren eigenen Generator von Grund auf — direkte Folge des Präfixes gen-, Namenskonvention §3.6.*

3 Registry — zentrale Wahrheitsquelle

Das *Registry* ist der einzige Konfigurationspunkt der *Pipeline*. Es enthält die Versionen der Stylesheets, die Erzeugungszeitstempel jedes Dokuments, die Render-

Flags, den Kompilierungsvermerk und das letzte Deployment. Jede Angabe, die sich aus Konsistenzgründen ableiten ließe, wird hier gelesen.

3.1 Aufbau

Das *Registry* exportiert ein Objekt, dessen Abschnitte in der folgenden Tabelle beschrieben sind. Die Datei trägt keinen *Zeitstempel* im Namen — sie ist die Quelle der *Zeitstempel*. Der Aufbau des *Registry* eines Anwenderprojekts weicht leicht ab und ist in Gemeinsame Struktur §5 dokumentiert: Er enthält keinen Abschnitt stylesheets, da die Projekte die Versionsverfolgung dem *Registry* des Kits überlassen.

Abschnitt	Schlüssel	Inhalt
project	docLanguage, documentAuthor, webAuthor, documentSiteBase	Identität und Sprache des Kits. Steuert die *L10N*-Auswahl der Stylesheets und die lokalisierten Konstanten des Cover-Sheet-Renderers. Enthält außerdem den Autor der Dokumente und der Seiten sowie die Wurzel der Verweise zwischen Dokumenten.
requires	coverSheet, readingGuide, glossary, textHighlights, documentTitles, brands, variableNames, hassEntities, colors	Flags für das Vorhandensein der *Datenmodule* und optionalen Dateien. Von den Generatoren gelesen, um require() und *Setter* zu steuern.
stylesheets	general, glossary, yaml, html	Semantische Version, *Zeitstempel* des Paars aus .js und .docx und für General die Untergrenze minDocumentVersion, unter der ein veröffentlichtes Dokument veraltet ist.
familyDomains	Array von Hostnamen	Domains, deren Links in der *WebView* der *Android*-App bleiben. Gelesen von setFamilyDomains().

Abschnitt	Schlüssel	Inhalt
documents	ein Slug je Dokument	*Zeitstempel* der letzten *Erzeugung*. Ein Eintrag je Sprachvariante bei übersetzten Dokumenten.
rendering	homeHref, languageSelector, coverSheet, toc	Flags zur Steuerung der *Erzeugung* von Liefergegenständen: Haus-Symbol der *Startseite*, die übrigen Seiten führen zum Verzeichnis ihrer Sprache, Platz des Sprachwählers, Höhenmodus der Rubriken des Deckblatts, *Erzeugung* und Ausblenden des HTML-Inhaltsverzeichnisses.
compiledWith	Objekt je Sprache oder null	Kompilierungsvermerk in der HTML-Fußzeile. null oder fehlend: kein Vermerk. Betrifft nie die .docx.
projectZip	version, filename, updated	Über die *Landing Page* herunterladbares Release. Versionsnummer nach Gemeinsame Struktur §11.
deploy	siteName, siteInfrastructure, lastDeploy	Name der veröffentlichten Website, der das Archiv benennt; Herkunft der Icons, der robots.txt und der .htaccess; ISO-*Zeitstempel* des letzten Deployments, ausschließlich vom Website-Generator geschrieben.

```
module.exports = {
  project:       { docLanguage: 'FR' },
  requires:      { coverSheet: true, readingGuide: true, glossary: true,
                   brands: true, variableNames: false, hassEntities: false },
```

```
  stylesheets: {
    general:  { version: '…', ts: 'AAAA-MM-JJ - HHhMM',
                minDocumentVersion: '…' },
    glossary: { version: '…', ts: 'AAAA-MM-JJ - HHhMM' },
    yaml:     { version: '…', ts: 'AAAA-MM-JJ - HHhMM' },
    html:     { version: '…', ts: 'AAAA-MM-JJ - HHhMM' },
  },
  familyDomains: ['sliver.lu', 'hexi.lu'],
  documents: {
    'reading-guide': { ts: 'AAAA-MM-JJ - HHhMM' },
    // un slug par document Kit ; un slug par variante de langue
  },
  rendering: {
    coverSheet: { sectionHMode: 'FIXED' },   // FIXED | DYNAMIC
    toc:        { generate: true, hidden: false },
  },
  compiledWith: null,                        // objet par langue, ou null
  projectZip:       { version: '…', filename: '…', updated: 'AAAA-MM-JJ' },
  deploy:       { lastDeploy: null },        // ISO 8601 UTC
};
```

3.2 Verwendung in den Generatoren

Jeder Generator, der Querverweise verwendet, lädt das *Registry* beim Start und liest die Werte dynamisch.

```
const registry = require('./Kit - Registry.js');

const v    = registry.stylesheets.general.version;   // version active
const ts   = registry.documents['reading-guide'].ts; // horodatage document
const mode = registry.rendering.coverSheet.sectionHMode;
const last = registry.deploy.lastDeploy;             // null ou ISO
```

Hinweis: *Versionsangaben im Fließtext verwenden String-Vorlagen. Eine Versionsänderung im Registry überträgt sich automatisch auf alle erzeugten Dokumente.*

3.3 Aktualisierungsregel

Das *Registry* wird von Hand aktualisiert, mit Ausnahme der Felder, die der Website-Generator selbst schreibt.

- **Stylesheet in neuer Version:** Version und *Zeitstempel* des Paars im Abschnitt stylesheets aktualisieren.
- **Dokument neu erzeugt:** den *Zeitstempel* des betreffenden Slugs aktualisieren. Ein übersetztes Dokument hat einen Eintrag je Variante.
- **Render-Flag angepasst:** den Abschnitt rendering aktualisieren. Die *Renderer* lesen dieses Flag bei der *Erzeugung* — für ein bloßes Umschalten muss kein Dokument neu erzeugt werden.
- **Kompilierungsvermerk:** compiledWith aktualisieren. Eine redaktionelle Entscheidung ohne Auswirkung auf die erzeugten Dokumente.

- **lastDeploy:** wird nach jeder ZIP-*Erzeugung* automatisch vom Website-Generator aktualisiert. Nie von Hand ändern.
- **Inhaltsdaten:** contentTs und contentHash im Abschnitt documents schreibt der Website-Generator. Bei jedem Durchlauf nimmt er die *Signatur* des reinen Textes — Überschriften, Absätze, Listenpunkte, Zellen, Hinweise, Tipps, Code- und *Prompt*-Blöcke, Bildunterschriften — ohne Einzug, Farbe, Kursivierung, Versionssignatur und *Zeitstempel*. Gleiche *Signatur*: nichts ändert sich. Andere oder fehlende *Signatur*: contentTs übernimmt den *Zeitstempel* des Dokuments. Die Verlaufsseite datiert nach contentTs, sodass eine *Stylesheet*-Erhöhung, die den Bestand ohne Textänderung neu erzeugt, dort keine Zeile hinzufügt. Nie von Hand ändern. Das Schreiben erhält die übrigen Schlüssel des Eintrags — etwa das PDF-Kennzeichen —, und ein Eintrag, dessen Wert eine Klammer enthält, wird mit seinem Namen gemeldet, statt neu geschrieben zu werden.
- **Symbole der Subsite:** die vier Symbole der *Subsite* — favicon.svg, favicon.ico, apple-touch-icon.png und index-icon.svg — liegen im Wurzelverzeichnis des Projekts, und der Durchlauf kopiert sie in die Ressourcen der Website. Die Elternseite zeichnet sie, das Projekt trägt sie: Eine allein aus ihrem Archiv übernommene *Subsite* behält ihre Identität. Der Durchlauf bricht ab und nennt die fehlende Datei; index-icon.svg wird nur erwartet, wenn rendering.indexIcon es erklärt. deploy.siteInfrastructure deckt nur noch robots.txt und .htaccess ab. Die Zeichnung von index-icon.svg zentriert sich in ihrer Anzeigefläche, senkrecht wie waagerecht, und füllt sie ganz aus: Das *Stylesheet* zentriert den Rahmen von 64 mal 64, nie den Inhalt der Datei, und eine verschobene Zeichnung erscheint genau so auf der Seite.
- **Projektarchiv:** ist projectShareable true, erzeugt der Durchlauf das Archiv des gesamten Baums und legt es unter assets/downloads/ ab, unter dem in projectZip.filename erklärten Namen. Das Veröffentlichungs-ZIP trägt es, sodass der Link der *Startseite* und projectZip.downloadUrl die angekündigte Version auflösen, ohne gesonderte Ablage. Der Website-*Ordner* und die Archive bleiben vom Paket ausgeschlossen.

Hinweis: *Das Registry wird mit dem Kit geliefert und bei jeder Aktualisierung in den Arbeitsbaum zurückgelegt. Kein Zeitstempel im Dateinamen — es ist selbst dessen Quelle.*

4 Delta-Deployment

Der Website-Generator erzeugt je nach Stand von lastDeploy ein vollständiges ZIP oder ein Delta-ZIP. Ein vollständiges Deployment enthält alle Dateien. Ein *Delta-Deployment* enthält nur die seit dem letzten Deployment geänderten Dateien.

4.1 Delta-Logik

Der Vergleich stützt sich auf den lokalen *Zeitstempel* des Dokuments: Er wird aus dem *Registry* gelesen, in ISO 8601 UTC umgerechnet und dann mit lastDeploy verglichen. Ein Dokument mit späterem *Zeitstempel* löst die Neuerzeugung seiner Seite, seines PDF und seiner Bilder aus. Die übrigen sind vom Delta-ZIP ausgeschlossen.

```
// lastDeploy null = passe complète
const lastDeployIso = (mode === 'full')
  ? null
  : (registry.deploy && registry.deploy.lastDeploy);

function kitTsToIsoDateTime(ts) {
  // 'AAAA-MM-JJ - HHhMM' en heure locale Luxembourg vers ISO 8601 UTC.
  // L'offset local (CET +1h, CEST +2h) est recupere via Intl.DateTimeFormat
  // et soustrait pour obtenir l'UTC reel. Sans cette conversion, toutes
  // les comparaisons seraient decalees de une ou deux heures.
  const m = /^(\d{4})-(\d{2})-(\d{2})\s*-\s*(\d{2})h(\d{2})$/.exec(ts || '');
  if (!m) return null;
  const [, y, mo, d, h, mi] = m;
  const naive = new Date(Date.UTC(+y, +mo - 1, +d, +h, +mi, 0));
  const dtf = new Intl.DateTimeFormat('en-US', {
    timeZone: 'Europe/Luxembourg',
    timeZoneName: 'longOffset',
  });
  const tz = dtf.formatToParts(naive).find(p => p.type === 'timeZoneName');
  const m2 = /GMT([+-])(\d{2}):(\d{2})/.exec(tz ? tz.value : '');
  if (!m2) return naive.toISOString();
  const offsetMin = (m2[1] === '+' ? 1 : -1) * (+m2[2] * 60 + +m2[3]);
  return new Date(naive.getTime() - offsetMin * 60 * 1000).toISOString();
}

function isNewerThanLastDeploy(ts, lastDeployIso) {
  if (!lastDeployIso) return true;          // passe complete
  const iso = kitTsToIsoDateTime(ts);
  if (!iso) return true;                        // prudence : inclure si TS
incompatible
  return iso > lastDeployIso;
}

// Dans la boucle des documents :
const meta = registry.documents && registry.documents[slug];
if (!meta || !meta.ts) continue;
if (mode !== 'full' && !isNewerThanLastDeploy(meta.ts, lastDeployIso)) {
  console.log('- ' + slug + ' (inchange, exclu du delta)');
  continue;
}
```

Hinweis: *Die Delta-Logik nutzte lange das Änderungsdatum der Datei auf dem Datenträger. In einer Umgebung, in der die Datei in jeder Sitzung neu importiert wird, wird dieses Datum überschrieben und gibt nicht das tatsächliche Erzeugungsdatum an: Das Delta enthielt am Ende fast alle Dateien. Der Vergleich stützt sich nun auf den Zeitstempel des Registry, die maßgebliche Quelle in Ortszeit, mit Umrechnung nach UTC.*

4.2 Sonderfall — das Stylesheet

Die CSS-Datei der Website wird neu erzeugt, wenn sich die Version des HTML-Stylesheets seit dem letzten Deployment geändert hat, ermittelt über dessen *Zeitstempel* im *Registry*. Da das *Registry* eines Anwenderprojekts keinen Abschnitt stylesheets enthält, wird das CSS dort nur im vollständigen Durchlauf neu erzeugt.

4.3 Benennung der ZIP-Dateien

Muster: {domaine} (AAAA-MM-JJ - HHhMM).zip. Das ZIP der *Subsite* des Kits trägt also die Domain des Kits. *Anwenderprojekte* ersetzen die Domain durch die ihrer *Subsite*.

4.4 Automatische Aktualisierung von lastDeploy

Nach erfolgreicher *Erzeugung* eines ZIP schreibt der Generator den ISO-*Zeitstempel* des aktuellen Zeitpunkts in lastDeploy. Dieser Vorgang schreibt das *Registry* auf dem Datenträger neu.

Hinweis: *lastDeploy nie von Hand ändern. Um ein vollständiges Deployment zu erzwingen, den Wert auf null setzen und neu erzeugen: Der Generator erkennt null und wechselt in den vollständigen Durchlauf.*

5 Aufbau der Website

Alle HTML-Dateien der Dokumentseiten liegen in einem eigenen Unterordner. Nur die *Landing Page* bleibt im Stammverzeichnis der *Subsite*. Diese Trennung macht die Hierarchie lesbar und die relativen Pfade von allen Dokumenten aus einheitlich.

5.1 Dateihierarchie

Pfad	Inhalt
[sous-site]/	Stammverzeichnis der *Subsite*
index.html, index-{langue}.html	Lader zu den Übersichtsseiten. Die einzigen Dateien, die das Kit ins Stammverzeichnis schreibt; sie enthalten keinen Inhalt.
html/	*Ordner* aller Dokumentseiten und der Übersichtsseiten je Sprache
html/[slug].html	Dokumentseite — eine Datei je veröffentlichtem Dokument
html/[slug]-[lang].html	Sprachvariante eines übersetzten Dokuments — Namenskonvention §2.4
html/glossaire.html	HTML-Glossar — vom Kit-Helper aus dem Begriffsmodul erzeugt, nie über den *AST*
assets/	Gemeinsame Ressourcen

Pfad	Inhalt
assets/kit-style.css	HTML-*Stylesheet*, aus dem *Stylesheet* erzeugt
assets/favicon.svg	Hauptsymbol. Vom Projekt getragen — Kit - Projet - *Prompt* §16
assets/favicon.ico	Mehrgrößen-Rückfall. Vom Projekt getragen — Kit - Projet - *Prompt* §16
assets/apple-touch-icon.png	*iOS*-Startbildschirmsymbol. Vom Projekt getragen — Kit - Projet - *Prompt* §16
assets/index-icon.svg	Illustration der *Startseite*. Vom Projekt getragen — Kit - Projet - *Prompt* §16
assets/downloads/	Zum Herunterladen von der *Startseite* angebotenes Projektarchiv. *Ordner* vorhanden, wenn projectShareable im *Registry* true ist: Der Durchlauf erzeugt dort das Archiv unter dem in projectZip.filename erklärten Namen, und das Veröffentlichungs-ZIP trägt es.
assets/pieces/	Vom Projekt abgelegte Anlagenstücke, unter ihrem ursprünglichen Namen — nie der Slug, nie eine erzeugte Datei. HTML-*Pipeline* §6.3
assets/pdf/	Aus den .docx erzeugte PDF — eines je freigegebenem Dokument
assets/png/{slug}/	Von *pandoc* extrahierte Bilder — ein *Ordner* je Dokument
robots.txt	Crawling-Anweisungen. Von der übergeordneten Website abgelegt — Kit - Projet - *Prompt* §16
.htaccess	*Apache*-Konfiguration der *Subsite*. Von der übergeordneten Website abgelegt — Kit - Projet - *Prompt* §16

5.2 Relative Pfade

Alle HTML-Dateien des Unterordners verwenden relative Pfade, die eine Ebene nach oben führen: *Landing Page*, *Stylesheet*, Bilder und PDF werden alle über das übergeordnete Verzeichnis erreicht.

Hinweis: *Nie einen Pfad relativ zum aktuellen Ordner verwenden, um von einer Dokumentseite zur Landing Page zu gelangen: Er würde auf eine im Unterordner nicht vorhandene Datei zeigen. Immer eine Ebene nach oben gehen.*

5.3 Aufbau des Deployment-ZIP

Das ZIP bildet die obige Hierarchie genau ab. Das Entpacken auf dem Server stellt die erwartete Struktur ohne Handarbeit wieder her.

```
kit.sliver.lu (AAAA-MM-JJ - HHhMM).zip
├── index.html
├── index-fr.html
├── index-de.html
├── html/
│   ├── kit-index-fr.html
│   ├── kit-index-de.html
│   ├── reading-guide.html
│   ├── manuel.html
│   ├── glossaire.html
│   └── ...
└── assets/
    ├── kit-style.css
    ├── pdf/
    │   └── reading-guide.pdf
    └── png/
        └── {slug}/
            └── {slug}
```

Eine Website veröffentlicht je Sprache eine Übersichtsseite in html/, benannt {préfixe}-index-{langue}.html — das Präfix stammt aus deploy.siteName. Die Basissprache trägt ihr Kürzel wie die anderen: Keine hat zwei mögliche Adressen.

Hinweis: *Das Stammverzeichnis gehört dem Projekt. Das Kit schreibt dort nur Lader: einen je veröffentlichter Sprache, index-fr.html und seine Gegenstücke, dazu index.html, das die im Registry unter project.docLanguage erklärte Sprache ausliefert — nie die des Browsers, sonst würde ein geteilter Link je nach Empfänger nicht an denselben Ort führen. Diese Namen erwartet die .htaccess der übergeordneten Website, deren Sprachblock absolute Pfade ab dem Stammverzeichnis prüft und ausliefert: Nur unter dieser Bedingung bleibt sie von einer Subsite zur anderen identisch. Ein Lader enthält keinen Inhalt und verwendet location.replace: Ein href oder ein Refresh-Tag schreiben einen Verlaufseintrag, und wer von der Übersichtsseite zurückgeht, würde sofort wieder dorthin geschickt.*

Hinweis: *Der Generator weigert sich, eine Datei im Stammverzeichnis zu überschreiben, die er nicht erzeugt hat, und bricht unter Nennung der Datei ab. Ein Projekt, dessen Ergebnis eine Website ist, legt dort seine eigene Startseite ab: Sie kann nicht ohne Meldung verschwinden. Die vom Kit geschriebenen Dateien tragen am Anfang eine Markierung, die sie von denen des Projekts unterscheidet.*

- **Lokalisierte Inhalte.** Rubriktitel, Dokumentbezeichnungen, Untertitel, Begrüßungstext und Titel der Übersichtsseite folgen der Sprache der Seite. Die Bezeichnungen stammen aus der Titeltabelle — Gemeinsame Struktur §6.8; die Rubriktitel werden im Website-Generator deklariert.

- **Links auf die richtige Variante.** Die deutsche Übersichtsseite verweist auf die deutschen Seiten. Ein nicht übersetztes Dokument erscheint dort unter seiner deutschen Bezeichnung und verweist auf die vorhandene Variante.
- **Sprachwähler.** Jede Übersichtsseite verweist über den Sprachwähler auf die anderen; seine Position richtet sich nach rendering.languageSelector — Gemeinsame Struktur §15. Die Verlaufsseite trägt je Sprache einen eigenen Namen: historique, verlauf, history.

6 PDF-Erzeugung

Ein PDF lässt sich aus einem Word-Dokument direkt in der Sitzung mit *LibreOffice* auf der Kommandozeile erzeugen, das in der Umgebung verfügbar ist. Liegt die Quelldatei im *Arbeitsbaum* vor, kann das PDF ohne Zutun des Nutzers erzeugt werden.

6.1 Referenzbefehl

```
libreoffice --headless --convert-to pdf document.docx --outdir ./

# Exemple avec chemin complet :
libreoffice --headless \
  --convert-to pdf \
  'Kit - Documentation - Pipeline HTML (AAAA-MM-JJ - HHhMM).docx' \
  --outdir /mnt/user-data/outputs/
```

6.2 Einbindung in den Website-Generator

Ist das PDF für ein Dokument im *Registry* aktiviert, erzeugt der Generator es mit *LibreOffice* und nimmt es im PDF-*Ordner* in das ZIP auf.

Hinweis: *LibreOffice ist in der Sitzung verfügbar. Den Nutzer nie bitten, die PDF selbst zu erzeugen, sobald die Quelldatei des Dokuments im Chat vorliegt. Diese Möglichkeit steht immer zur Verfügung, ohne Bedingung.*

6.3 Anhänge und Handbücher — externes PDF

Die Dokumente der Kategorien Annexe und Manuel eines Anwenderprojekts haben ein Quell-PDF, das nie von *LibreOffice* erzeugt wird. Der Nutzer legt das PDF von Hand im *Arbeitsbaum* ab, mit demselben Basisnamen wie das zugehörige Dokument. Der Generator kopiert es unverändert — für diese Dokumente wird keine Umwandlung aufgerufen.

Zuordnungsregel über den Basisnamen: Für jedes Dokument dieser Kategorien sucht der Generator ein PDF, dessen Name mit dem des Dokuments ohne Zeitstempelsegment und Endung übereinstimmt. Exakte Übereinstimmung, ohne Toleranz.

- **Dateipaar:** Das Dokument trägt einen *Zeitstempel*, das PDF nie — es ist eine eingefrorene Quelldatei, die unverändert abgelegt wird.

- **HTML-Slug:** mit der Kategorie als Präfix, um jede Kollision zu vermeiden, falls zwei gleichnamige Dokumente in zwei verschiedenen Kategorien existieren. Das zugehörige PDF folgt derselben Regel.
- **Verhalten der Pipeline:** direkte Kopie des PDF in den PDF-*Ordner*; das Dokument durchläuft die normale Kette aus HTML-*Erzeugung* und Validierung.
- **Lautes Scheitern bei Fehlen:** Fehlt das erwartete PDF zum Zeitpunkt der *Erzeugung* im *Arbeitsbaum*, bricht der Generator ab und nennt die fehlende Datei. Kein stiller Rückgriff auf ein erzeugtes PDF, keine Seite ohne Icon.
- **Rubriken der Landing Page:** Die ständigen Rubriken Anhänge und Handbücher erscheinen am Ende der *Landing Page* der *Subsite*, nach den nummerierten Rubriken. Jede wird ausgeblendet, wenn sie kein Dokument enthält. Interne Reihenfolge alphabetisch.
- **Gattung des gekoppelten Dokuments:** das Dokument, das das Stück begleitet, ist ein Feststellungsdokument — *Kit Prompt* §12. Es berichtet über das Stück, ohne es abzuschreiben, benennt, was nicht stimmt, und schließt mit dem Abschnitt der festgestellten Punkte. §6.3 regelt die Mechanik; die Gattung des Textes steht im *Prompt*.
- **Das Merkblatt wird veröffentlicht, die Schaltfläche liefert das Stück:** das Anlagenmerkblatt wird wie jede andere Seite veröffentlicht, und seine Download-Schaltfläche liefert das Originalstück, nie das aus ihm erzeugte PDF. Das Stück wird im *Registry* über den Schlüssel piece des Dokumenteintrags erklärt und liegt im Wurzelverzeichnis des Projekts; der Durchlauf kopiert es nach assets/pieces/, unter seinem ursprünglichen Namen und nie unter dem Slug, und bricht ab, indem er es nennt, wenn es fehlt. Der *Ordner* trennt das Abgelegte vom Erzeugten: Ein Stück und das PDF eines Merkblatts können nicht kollidieren.
- **Das PDF des Merkblatts wird weiter erzeugt:** es kommt in das ZIP der Website und in das Archiv des Projekts, für den *Ordner*, und keine Seite verweist darauf: So muss der Leser nie zwischen zwei PDF wählen, von denen nur eines gilt. Die Schaltfläche wechselt weder Symbol noch Beschriftung je nach Seite — der Hinweis am Anfang des Merkblatts sagt, was sie liefert.

Hinweis: *Das PDF-Kennzeichen gilt für ein Anlagenmerkblatt wie für jedes andere Dokument: Sein PDF entsteht aus seiner .docx und bleibt in assets/pdf, ohne dass eine Seite darauf verweist. Das Stück wird gesondert aus assets/pieces geliefert.*

Referenzmuster für den Website-Generator: Erkennung nach Kategorie, Zuordnung über den Basisnamen, Kopie des Quell-PDF.

```
// gen-{prefix}-site.js — extrait pour une fiche d'annexe
const fs   = require('fs');
const path = require('path');

// La piece se declare au Registry, par la cle piece de l'entree du
// document ; elle vit a la racine du projet. Aucun appariement par nom
// de base : le Registry dit le fichier, et lui seul.
function pieceDeclaree(slug, registry, projectDir) {
  const meta = (registry.documents || {})[slug] || {};
  if (!meta.piece) return null;
  const src = path.join(projectDir, meta.piece);
  if (!fs.existsSync(src)) {
    throw new Error('Piece d\'annexe manquante : ' + meta.piece);
  }
  return src;
}

// Dans la boucle de generation de chaque document :
const src = pieceDeclaree(slug, registry, PROJECT_DIR);
if (src) {
  fs.mkdirSync(path.join(OUTPUT_DIR, 'assets/pieces'), { recursive: true });
  fs.copyFileSync(src,         path.join(OUTPUT_DIR,         'assets/pieces',
path.basename(src)));
  // pdfHref de renderDocument pointe la piece, jamais le PDF de la fiche
}
// Le .docx suit la chaine ordinaire : page HTML, et PDF si le drapeau
// du document l'autorise. Ce PDF reste dans assets/pdf, sans lien.
```

7 Robustheitsregeln

Dieser Abschnitt dokumentiert die gefährlichsten Fallen der *Pipeline* und die zwingenden Regeln, die daraus folgen. Jede Regel geht auf einen tatsächlichen Fehler zurück, der zu Inhaltsverlust oder beschädigter Darstellung geführt hat.

7.1 Falle doc.paragraphs — stiller Verlust der Tabellen

Liest ein Aktualisierungsskript ein bestehendes Dokument mit einer *Python*-Bibliothek und iteriert über dessen Absätze, um es neu aufzubauen, verschwinden alle Tabellen stillschweigend. Das erzeugte Dokument ist syntaktisch gültig, aber um seinen tabellarischen Inhalt beschnitten.

Messung an einem echten Dokument: 300 Absätze im XML, davon nur 84 für die *Iteration* sichtbar — 28 Prozent. Die übrigen 216 lagen in den Tabellen, und alle 18 Tabellen gingen ohne die geringste Warnung verloren.

Hinweis: *Nie ein bestehendes Dokument durch Iteration über seine Absätze aktualisieren. Das Generatorskript immer von Grund auf aus der Quelldatei neu schreiben. Die Quelldatei wird im Chat bereitgestellt, um das Skript neu zu schreiben, nie, um das XML direkt zu bearbeiten.*

7.2 Escaping bei Pfaden

Die Escape-Funktion des HTML-Stylesheets schützt Sonderzeichen bei der Darstellung von Fließtext. Sie darf nie auf ein Pfadattribut angewendet werden.

- **Verboten:** eine Bildquelle, ein Linkziel oder einen Dateipfad escapen — das beschädigt den Pfad.
- **Erlaubt:** einen Zelltext oder einen Absatzinhalt escapen — nur Fließtext.

Hinweis: *Das Escapen eines Bildpfads erzeugt ein beschädigtes Attribut: Das Bild wird unerreichbar, ohne ausdrückliche Fehlermeldung im HTML.*

7.3 Vorab-Lektüren speziell für die HTML-Pipeline

Die allgemeine Regel, die Reference vor jeder *Erzeugung* zu lesen, steht in *Prompt* §4.1 und §5.1. Sie gilt hier ausnahmslos. Die folgenden Lektüren sind speziell erforderlich, bevor die erste Zeile eines *AST*-Konverters oder eines Website-Generators geschrieben wird.

- **HTML Reference:** vollständig, insbesondere die Funktionsverträge und die *AST*-Umwandlungspipeline.
- **General Reference:** §1.2 für den Grundaufbau eines Dokuments und §13 für die Rückgabeverträge — der Konverter erzeugt HTML, arbeitet aber mit Strukturen aus dem Word-*Modell* (.docx).

Regel des strikten Abbruchs: Wurde eines dieser Dokumente in der laufenden Sitzung nicht gelesen, vollständiger Abbruch. Keine Frage stellen, nichts annehmen, nicht fortfahren.

7.4 Validierung des Quelldokuments vor der Umwandlung

Ein Dokument gelangt erst in die HTML-*Pipeline*, nachdem es die Validierungskette durchlaufen hat. Der ausführliche Vertrag des Validators — Liste der Prüfungen, Verwendung, Rückgabecodes — steht in *Quality Control* §3.1. Die folgenden Bedingungen sind der *Pipeline* eigen und werden am Eingang geprüft. *Signatur* und Version stellen nicht dieselbe Frage: Die *Signatur* sagt, ob das Dokument vom Kit erzeugt wurde, die Untergrenze, ob es noch getreu dargestellt wird.

Bedingung	Erwartet	Folge bei Fehlen
***Signatur* vorhanden**	StylesheetVersion und GeneratedAt in den Dokumenteigenschaften	Fehlt sie: Das Dokument wird bei der Umwandlung abgewiesen
Version aktuell	StylesheetVersion größer oder gleich minDocumentVersion, der im *Registry* erklärten Untergrenze	Kleiner: Das Dokument ist veraltet und gesperrt
***Validator* bestanden**	kit_validate_docx.js exit 0 auf der Quell-.docx	Ein nicht validiertes Dokument darf nie in die *Pipeline* gelangen

Hinweis: *Der Website-Generator weist jedes Dokument ohne Signatur oder mit einer Version*

unterhalb der Untergrenze zurück. Ein veröffentlichtes Dokument ist also durch eine Formatierungs-Engine gegangen, deren Darstellung noch als getreu gilt. Eine neue Stylesheet-Version macht nichts mehr veraltet: Der Bestand kann mehrere Versionen zugleich tragen, einheitlich in der Darstellung, ohne es in der Nummer zu sein. Das Verschieben der Untergrenze ist der bewusste Schritt, der die allgemeine Neuerzeugung erzwingt, und diese Einschätzung fängt keine Prüfung ab.

7.5 Die Ebene einer Überschrift kommt aus dem Dokument

Die Konvertierung liefert einen Überschriftenblock mit seiner Ebene, und der Generator liest sie unverändert. Dies beruht auf der Vererbungskette der Stile, die in General v1.94 durch die Deklaration des Stils Normal wiederhergestellt wurde: ohne sie erkannte ein Konverter, der die Vererbung auflöst, keine Überschrift, und alles Folgende blieb auf der Ebene der vorigen Überschrift.

Der Text einer Überschrift entscheidet nichts. Ein Absatz, der mit einer Nummer beginnt, bleibt ein Absatz; eine Überschrift, deren Text mit einer Ziffer beginnt, bleibt eine Überschrift, was *Kit Prompt* §12 im Übrigen der Lesbarkeit wegen untersagt.

Hinweis: *Ein Dokument, das mit einem Stylesheet vor General v1.94 neu erzeugt wurde, deklariert den Stil Normal nicht: Seine Überschriften fallen bei der Konvertierung auf Absätze zurück, und die Ebene alles Folgenden fällt mit ihnen. Der Bestand wird daher vor seiner nächsten Veröffentlichung neu erzeugt.*

7.6 Absatz mit fetter Einleitung — das Trennzeichen bleibt

Trifft der *AST*-Konverter auf einen Absatz, der mit Fettschrift beginnt, erkennt er eine fette Einleitung und ruft den entsprechenden Helper mit dem fetten Text und dem Rest des Satzes auf.

Der Rest geht unverändert weiter. Das Trennzeichen nach der Einleitung — Doppelpunkt, Geviertstrich, einfacher Bindestrich — gehört zum Dokument: Es steht auf dem Papier und muss auf der Seite stehen. Das HTML-*Stylesheet* setzt seinerseits keines, also gibt es keine Verdopplung.

Regel: kein Entfernen am Anfang des Restes, weder Trennzeichen noch Leerzeichen. Ein Entfernen kostet ein Zeichen, das der Textkonservierungsschutz danach einfordert, und die Veröffentlichung bricht an einem Verlust ab, der vom Generator und nicht vom Dokument kommt. Diese Regel gilt für jeden von Grund auf neu geschriebenen *AST*-Konverter.

7.7 Familien- und externe Hyperlinks

Der HTML-Generator muss Links zu den Domains der Familie und ihren Subdomains, die in der *WebView* der mobilen App bleiben, von externen Links unterscheiden, die im Systembrowser öffnen müssen. Diese Unterscheidung trägt ein interner Helper des HTML-Stylesheets, der die Attribute für das externe Öffnen nur bei den Letzteren setzt.

Konfiguration: Die Liste der Familiendomains ist im *Registry* deklariert, und jeder Website-Generator ruft den zugehörigen *Setter* am Anfang auf. Wird der *Setter* nie aufgerufen, gelten alle absoluten Links als extern.

Zuordnungsmechanik. Relative URL und *Anker* gehören per Definition zur Familie. Nicht-http-Schemata — E-Mail, Telefon — gehören per Definition zur Familie, da ihre Behandlung dem System obliegt. Bei absoluten URL wird der Hostname extrahiert und mit der Liste verglichen: exakte Übereinstimmung oder Subdomain. Querverweise zwischen Websites der Familie bleiben daher in der *WebView*.

Auf externe Links wird als Ansatzpunkt eine CSS-Klasse gesetzt, die bewusst leer deklariert ist. Sie erlaubt einem Projekt, externe Links anders zu gestalten, ohne das *Stylesheet* anzufassen. Der Helper wird bei der URL-Erkennung im Fließtext und für Tabellenzellen mit Link aufgerufen.

Hinweis: *Abhängigkeit von der mobilen App. Die Attribute für das externe Öffnen wirken nur, wenn die App das zugehörige Routing in ihrer WebView-Komponente umsetzt. Andernfalls werden sie stillschweigend ignoriert und die externen Links sind tot.*

7.8 Einspaltige Blöcke bei der Konvertierung

Ein Code- oder Dialogblock kann aus der Konvertierung mit seiner ersten Zeile in der Kopfzeile seiner Tabelle hervorgehen. Der Generator liest daher für diese Blöcke alle Zeilen und nimmt für Hinweis und Tipp die erste dieser Menge.

Die Trennung von Kopf und Rumpf gilt nur noch für die gewöhnliche Tabelle. Allein am Rumpf gelesen, ging die erste Zeile eines Blocks verloren, und die Erhaltungswache brach den Durchlauf ab, indem sie ein unauffindbares Fragment nannte.

Hinweis: *Abhängigkeit von der mobilen App. Die Attribute für das externe Öffnen wirken nur, wenn die App das zugehörige Routing in ihrer WebView-Komponente umsetzt. Andernfalls werden sie stillschweigend ignoriert und die externen Links sind tot.*

7.9 Pflicht-Setter am Anfang des Website-Generators

Jeder HTML-Website-Generator muss am Anfang, vor dem ersten erzeugten Element, eine Reihe von Settern aufrufen. Fehlt ein erforderlicher *Setter*, wird die zugehörige *Pipeline* stillschweigend deaktiviert: Der Generator läuft fehlerfrei, erzeugt aber eine semantisch beschnittene Website.

#	Setter	Quelle	Wirkung bei Fehlen
1	setLanguage	Registry.project.docLanguage	Standard EN — alle lokalisierten Zeichenketten auf Englisch
2	setFamilyDomains	Registry.familyDomains	Alle absoluten Links gelten als extern

#	Setter	Quelle	Wirkung bei Fehlen
3	setGlossaryTerms	Begriffsmodul des Projekts	Kein Glossarlink auf irgendeiner Seite — Website semantisch beschnitten
4	setBrands	Markenmodul des Projekts	Keine Marke in Kapitälchen dargestellt
5	setVariableNames	Variablenmodul des Projekts	Kein Variablenname in grüner Kursivschrift dargestellt
6	setHassEntities	Entitätenmodul des Projekts	Keine Entität in violetter Kursivschrift dargestellt
7	setGlossaryHref	Slug des Glossars der dargestellten Sprache	Alle Begriffslinks zielen auf dieselbe Seite — ein nicht französischsprachiger Leser erhält die Definitionen in der Sprache des Projekts
8	setDocumentTitles	[Préfixe] - Document Titles (TS).js — titleEntries und categoryLabels	Bezeichnungen in der Projektsprache eingefroren, kein Dokumentverweis als Hyperlink, *Deckblatt*-Untertitel nicht zusammengesetzt
9	setLanguageSelector	Registry.rendering.languageSelector	Standard „document“: Icons auf Dokumentebene statt des Menüs in der Leiste
10	setTextHighlights	Text-Highlights-Modul des Projekts	Kein deklariertes Fragment hervorgehoben

Die Daten-*Setter* hängen von den Flags des *Registry* ab — ein Projekt ohne Glossar ruft den zugehörigen *Setter* nicht auf, da das Modul nicht existiert. setLanguage und setFamilyDomains sind universell und immer erforderlich. setGlossaryHref betrifft nur Projekte, die ihr Glossar in mehreren Sprachen veröffentlichen: Sein Standardwert passt für ein einsprachiges Projekt, und er wird einmal je dargestelltem Dokument aufgerufen, da sich das Ziel mit der Sprache des Dokuments ändert.

Der Generator übergibt dem *Stylesheet* außerdem den aus dem *Registry* gelesenen Kompilierungsvermerk und für jedes übersetzte Dokument die Liste seiner veröffentlichten Sprachen. Diese Liste wird im Generator deklariert, nie im *Registry*: Der Übersetzungsumfang ist eine redaktionelle Entscheidung je Dokument, kein stabiles Projekt-Flag.

```
// Patron canonique en tete de gen-{prefix}-site.js — ordre recommande
const registry = require('./[Prefixe] - Registry.js');
const style    = require('./Kit - Stylesheet - HTML - Code (TS).js');

// 1. Langue active — depuis Registry
style.setLanguage(registry.project.docLanguage || 'EN');

// 2. Domaines famille — toujours appele, meme si la liste est vide
style.setFamilyDomains(registry.familyDomains || []);

// 3 a 6. Setters conditionnels selon Registry.requires
if (registry.requires && registry.requires.glossary) {
  const { glossarySearchTerms } = require('./[Prefixe] - Glossary -
Terms.js');
  style.setGlossaryTerms(glossarySearchTerms);
}
if (registry.requires && registry.requires.brands) {
  const { brandEntries } = require('./[Prefixe] - Brands.js');
  style.setBrands(brandEntries);
}
// idem variableNames et hassEntities
```

Hinweis: *Der Linter kit_check_setters.py prüft statisch, dass alle erforderlichen Setter tatsächlich aufgerufen werden. Er läuft vor jeder Website-Erzeugung. Da ein fehlender Setter zur Laufzeit keine Meldung erzeugt, ist dieser Linter das einzige automatische Netz — siehe Quality Control §3.4.*