

Diwall — Dokumentation

Visuelle Wahrnehmung und RPA für LLM-Agenten

Version 1.24.5

2026-09-30

Übersetzung. Maßgeblich ist die englische Fassung.

Über diese Zusammenstellung

Dieses Dokument fasst vier Texte zusammen, die auch einzeln bestehen und sich je an eine andere Leserschaft richten. Die *Kurzreferenz* beginnt mit den Befehlen selbst, für alle, die sofort einen brauchen. Die *Übersicht* stellt Diwall vor und installiert es. Das *Bedienerhandbuch* erklärt, was Sie delegieren und warum sich das Werkzeug so verhält. Das *Betriebshandbuch* ist die Referenz: Befehle, Aktionen, Optionen, Rückgabecodes. Jedes beginnt mit einer eigenen Einleitung, weil jedes auch allein gelesen wird.

Inhaltsverzeichnis

Diwall – Schnellreferenz **4**
Drei Befehle . . . 4
Die Schleife . . . 5
Die Ausgabe in dieser Reihenfolge lesen . . . 5
Jede Aktion . . . 5
Zugangsdaten – die einzige korrekte Formularausgabe . . . 6
Wenn etwas Widerstand leistet . . . 6
Rückgabecodes . . . 6

Diwall – Gemeinsame visuelle Referenz zwischen Mensch und LLM **7**
Was ist Diwall? . . . 7
Was das Modell tatsächlich empfängt . . . 7
Architektur . . . 8
Fähigkeiten . . . 8
Anforderungen . . . 10
Installation . . . 10
Paket – der einfache Weg . . . 10
Von der Quelle – zur Modifikation von Diwall selbst . . . 11
Deinstallation . . . 11
Verwendung (durch Ihr LLM) . . . 12
Einfache Aufnahme . . . 12
ReAct-Schleife (mehrstufige Navigation) . . . 12
RPA-Szenario . . . 12
Zugangsdaten . . . 13
Sicherheit . . . 13
Speicherung der Aufnahmen . . . 13
Lokale Modelle vs. Cloud-Modelle . . . 13
Verzeichnis für Anmeldeinformationen . . . 13
Dokumentation in anderen Sprachen (Version 1.23.0) . . . 13
Für LLMs, die Diwall entdecken . . . 14
Mitwirkende . . . 14
Lizenz . . . 14

Diwall – Bedienungsanleitung **15**
Warum Diwall – was Sie tatsächlich delegieren . . . 15
Das Problem, das Diwall löst . . . 15
Was Sie delegieren . . . 15
Was Sie behalten . . . 15
Respektvolles Navigieren (Version 1.15.0) . . . 16
Wann ist Diwall das richtige Werkzeug? . . . 16
Demonstrationsszenarien . . . 16
Fall 1 – Fehlerbehebung bei lokalen CSS-/JavaScript-Dateien . . . 16
Fall 2 – Vergleich von Hardwarekomponenten in verschiedenen Geschäften . . . 17
Fall 3 – Erkundung und Zusammenfassung von technischer Dokumentation (Single-Page-Anwendungen) . . . 17
Fall 4 – Konfiguration eines selbst gehosteten Observability- oder Analyse-Dashboards . . . 17
Fall 5 – Betrieb einer Ticketing-Plattform von Anfang bis Ende . . . 18
Fall 6 – Verfolgung eines regionalen Veranstaltungskalenders . . . 18
Fall 7 – Test des Zugriffs auf E-Commerce-Seiten unter realen Bedingungen im Rahmen von ”Respectful Navigation” . . . 18

Voraussetzungen vor dem Start . . . 19
Konfiguration der Anmeldeinformationen pro Projekt . . . 19
Eine Seite erfassen und analysieren . . . 19
Automatisierung eines Anmeldeformulars . . . 20
Gültigkeit mehrerer Seiten mit einem einzigen Aufruf prüfen . . . 20
Extrahieren eines Wertes von der Seite . . . 21
Visuelle Überwachung einrichten . . . 21
Einrichten einer kontinuierlichen Strukturüberwachung (Version 1.18.0) . . . 21
Häufige Fehlerquellen . . . 22
Deinstallation von Diwall . . . 23
Einsicht in die Betriebshistorie . . . 23

Diwall – Betriebshandbuch **24**
Inhaltsverzeichnis . . . 24
1. Installation überprüfen . . . 24
1a. Installation aus einem Paket – der einfache Weg . . . 25
1b. Installation aus dem Quellcode – für die Modifikation von Diwall selbst . . . 26
2. Eine Seite erfassen . . . 27
2a. Schnelle Aufnahme – nur Text, ohne PNG (~2 Sekunden) . . . 27
2b. Vollständige visuelle Erfassung mit nummerierten Elementen . . . 27
2c. Lesen Sie zuerst die Gebrauchsanweisung . . . 27
2d. Lesen Sie `etat` für eine Ja/Nein-Entscheidung (Version 1.16.0) . . . 28
2e. `mode_conseille` — Hinweise zur Vorflugkonfiguration (Version 1.18.0) . . . 28
3. Respektvolles Navigieren (v1.15.0) . . . 29
3a. Stealth-Modus `--stealth` . . . 29
3b. Höflichkeitsbedingte Verzögerungen . . . 29
3c. Kennzahlen zur Bewertung der Auswirkungen . . . 30
3d. Stealth-Benchmark – quantitativ (Version 1.17.1) . . . 30
3e. Erkennungssignal für Web Application Firewall (WAF) (Version 1.16.0, verfeinert in Version 1.17.2) . . . 31
3f. Angegebene Sprache (Version 1.24.1) . . . 31
4. Verschlüsseltes Verzeichnis und Zugangsdaten . . . 31
4a. Struktur . . . 31
4b. Ein Formular ausfüllen – die unumstößliche Regel . . . 32
4c. Auswahl der Zugangsdatendatei für einen Durchlauf . . . 32
4d. TOTP / Multi-Faktor-Authentifizierung . . . 32
4e. Integritätsprüfsumme (optional, ab Version 1.15.0) . . . 33
4f. Verschlüsseltes Verzeichnis geschlossen – was tun? . . . 33
4g. HTTP-Basisauthentifizierung — `--http-credentials` (v1.21.0) . . . 33
5. Schreiben und Ausführen eines RPA-Szenarios . . . 34
5a. 3-stufiges Protokoll . . . 34
5b. Vollständiges Szenario: Anmelden und zwischen Seiten navigieren . . . 34
5c. Daten aus dem DOM extrahieren . . . 35
5d. Aussagen zu evaluer (rpa.py nur) . . . 35
5e. Unter-Szenarien (declencher_scenario) . . . 35
5f. Überprüfen Sie, ob Sie sich auf der richtigen Seite befinden, bevor Sie Änderungen vornehmen . . . 36
5g. Eine Sitzung fortsetzen (mit persistenten Cookies) . . . 36
5h. Strukturelle Nicht-Regression ohne Pixel – `--replay-verifier` (v1.17.0) . . . 36
5i. Ein langes Szenario nach einem Fehler fortsetzen — `--checkpoint` (v1.17.0) . . . 37
5j. Elemente innerhalb eines Iframes ansteuern (Version 1.17.0) . . . 37
5k. Verschachtelte Iframes — `iframe_chemin` (v1.18.0) . . . 37

6. Aktionen – vollständige Referenz . . . 38
7. Umgang mit häufigen Hindernissen . . . 39
7a. Cookie-Banner / Sperrbanner . . . 39
7b. Element außerhalb des sichtbaren Bereichs . . . 39
7c. Web-Komponenten – Shadow DOM . . . 40
7d. Single-Page Applications (SPA) (React, Vue, Angular) – Navigation ohne Neuladen . . . 40
7e. CSS-Dialog oder showModal() . . . 40
7f. Lange Operation (Spinner, Batch-Job) . . . 40
7g. Kapazitätsgrenze erreicht (Version 1.15.0) . . . 41
7h. `<select>` Formularfeld . . . 41
7i. Ungültige SoM-IDs beim nächsten Durchlauf . . . 41
7j. SoM ID-Drift bei hochdynamischen Seiten – hybride Lösung (v1.24.0) . . . 41
7k. Website durch WAF blockiert (sofortige 403-Fehlermeldung) . . . 41
7l. Die anfängliche Navigation wird nie abgeschlossen — `--wait-until` (v1.22.0) . . . 42
8. Visuelle Überwachung — watch.py . . . 42
8a. Eine Referenz speichern . . . 42
8b. Vergleich mit der Referenz (Pixeldifferenz) . . . 42
8c. Semantische Vergleichung (LLM) . . . 43
8d. Eine animierte Zone ignorieren . . . 43
8e. Überwachungs-Schleife . . . 43
8f. Cron für autonome Überwachung . . . 43
8g. Kontinuierliche strukturelle Überwachung — `monitor-verifier.sh` (v1.18.0) . . . 44
9. Betriebsprotokoll . . . 44
9a. Protokolldrehung (G-36, CHANTIER_SANITISATION.md) . . . 45
10. Befehlszeilenparameter – Referenz . . . 45
shot.py . . . 45
rpa.py . . . 47
watch.py . . . 48
11. Rückgabecodes und Ausgabe . . . 48
Rückgabecodes . . . 48
Struktur des Ausgabe-JSON . . . 49
Fehler — Formatierung . . . 50
Referenzpfade . . . 50

Diwall – Schnellreferenz

Version 1.24.5 – September 2026

Alles auf einer Seite. Vollständige Referenz: `docs/MANUEL.md`.

Drei Befehle

```
# Eine Seite anzeigen: PNG-Bild + nummerierte Elemente + Accessibility-Baum.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url URL --som --a11y

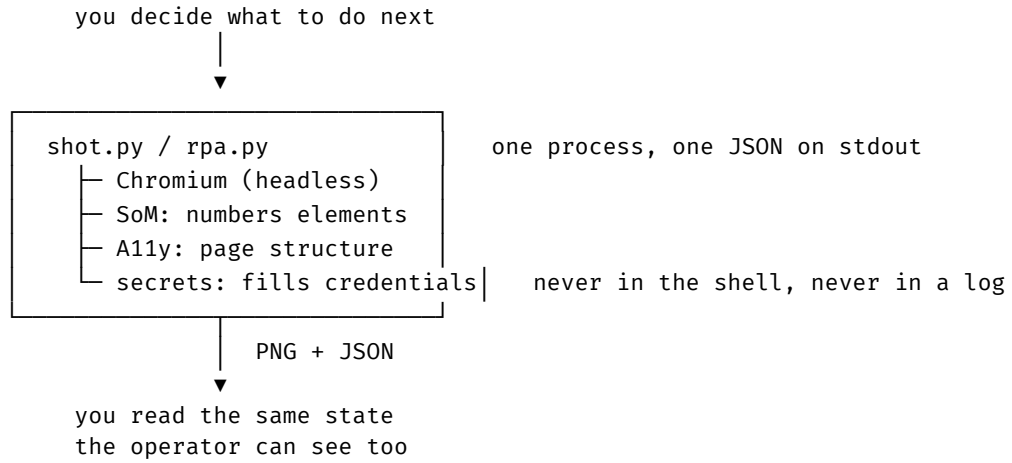
# Lesen ohne Zwischenspeichern (~2 Sekunden schneller, keine PNG-Dateien).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url URL --mode fast

# Führen Sie ein Szenario aus.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py --scenario FILE.json
```

Über das `.deb` installiert? Verwenden Sie `diwall-shot` und `diwall-rpa` statt der vollständigen Pfade. Der erste Aufruf auf einer Maschine benötigt `--guide-version X.Y`, auszulesen mit `grep notice-version`

`/opt/diwall/docs/GUIDE_LLM.md`.

Die Schleife



Der Sitzungsstatus wird in einer Datei gespeichert, nicht im Prozess: ein zweiter Aufruf mit `--reprendre-session` verwendet Cookies erneut – niemals den DOM-Zustand.

Die Ausgabe in dieser Reihenfolge lesen

Lesen Sie	Zeigt Ihnen
`succes`	ob die Ausführung abgeschlossen wurde
`boussole.url_courante`	wo Sie tatsächlich angekommen sind
`boussole.dernier_code_http`	letzter Navigationsstatus
`etat.pret_a_agir` + `etat.raisons`	wahrgenommene Reibung – ein Bericht, niemals eine Fehlermeldung
`capture_som` / `elements_som`	was Sie anklicken sollen und seine Nummer
`respect`	Ihre eigene Spur: Seiten, Aktionen, Dauer

Wenn `boussole` nicht Ihren Erwartungen entspricht, stoppen Sie vor jeglicher verändernden Aktion.

Jede Aktion

`type` ist immer erforderlich. Die Schlüssel unten sind die zusätzlichen.

Aktion	Erforderlich	Optional
`naviguer`	`url`	—
`cliquer`	`selecteur`	`force`, `repli_js`
`cliquer_som`	`id`	—
`cliquer_visuel`	`description`	—
`cliquer_iframe`	`iframe_selecteur` | `iframe_chemin`, `selecteur`	`force`
`remplir`	`selecteur`, `valeur`	`secret_cle`
`remplir_som`	`id`, `valeur`	`secret_cle`
`remplir_iframe`	`iframe_selecteur` | `iframe_chemin`, `selecteur`, `valeur`	`secret_cle`

Aktion	Erforderlich	Optional
`capturer`	`nom`	`som`
`evaluer`	`script`	`attendu` | `contient` | `motif`
`defiler`	`px` | `selecteur`	—
`pause`	`ms`	`interval_capture`
`attendre`	`selecteur`	`interval_capture`
`attendre_selecteur_present`	`selecteur`	—
`attendre_absence`	`selecteur`	`delai_initial_ms`
`attendre_navigation`	—	—
`attendre_url`	`motif`	`attendre_changement`
`attendre_reseau_calme`	—	`timeout_ms`
`attendre_mfa_ntfy`	`id_som`	`timeout`
`nettoyer_overlay`	`selecteur`	—
`declencher_scenario`	`scenario`	—

Zugangsdaten – die einzige korrekte Formularausgabe

```
{"type": "remplir_som", "id": 3, "valeur": "depuis_secrets", "secret_cle": "password"}
```

Extrahieren Sie ein Geheimnis niemals in die Shell. `lib/repertoire_chiffre.py` löst es innerhalb des Playwright-Prozesses auf; der Wert erreicht weder Ihre Befehlszeile noch Ihren Verlauf noch irgendein Protokoll. `depuis_secrets_totp` tut dasselbe für einen TOTP-Code.

Wenn etwas Widerstand leistet

Symptom	Versuchen Sie
Klick führt zu einem Timeout, Element ist visuell ausgeblendet	`"force": true`, dann `"repli_js": true`
Element wird nicht von SoM nummeriert	`--shadow-dom` (Shadow Roots öffnen)
Element befindet sich außerhalb des sichtbaren Bereichs	`defiler` zuerst — prüfen Sie `boussole.som_hors_viewport`
Seite lädt nie vollständig	`--wait-until load`
Absenden-Button hat keine Funktion, kein Fehler	native HTML-Validierung — Formular über `evaluer` absenden
`exit 42`	verschlüsseltes Verzeichnis nicht gemountet: `diwall-monter-secrets`
`exit 43`	kein `diwall.conf` — Beispiel daneben kopieren
`guide_non_lu`	einmal `--guide-version` ausführen
403 / 429	lesen Sie `respect.waf_bloquants` — ein Signal, keine Ausnahme

Rückgabecodes

`0` Erfolg · `1` Fehler im Playwright oder fehlende Assertion · `2` Abweichung der Bildschirmauflösung (`watch.py`) · `3` Falscher Interpreter, verwenden Sie die virtuelle Umgebung · `42` Verschlüsseltes Verzeichnis geschlossen oder fehlerhafte Prüfsumme · `43` `diwall.conf` fehlt.



Abbildung 1: Diwall-Logo: eine D-förmige Wand, deren innerer Ausschnitt ein offenes Auge ist

Diwall – Gemeinsame visuelle Referenz zwischen Mensch und LLM

Für den menschlichen Bediener: Diwall ermöglicht es Ihnen, die visuelle Überprüfung an Ihr LLM zu delegieren. Beide sehen denselben Bildausschnitt – Sie müssen nicht mehr darauf vertrauen, was das LLM sagt. **Für das LLM**: `docs/GUIDE_LLM.md` ist Ihre Betriebsanleitung. Beginnen Sie dort. Wenn Sie ein KI-Agent sind, der Diwall entdeckt, überspringen Sie die formatierte Startseite und laden Sie Ihre Anweisungen direkt: **https://diwall.davalan.fr/instructions.md**

Was ist Diwall?

Diwall schafft eine **gemeinsame visuelle Referenz** zwischen einem menschlichen Bediener und einem Sprachmodell. Es ermöglicht dem LLM, **Weboberflächen zu sehen** – und gibt dem menschlichen Bediener die Möglichkeit, **visuelle Überprüfungen auszulagern**, ohne die Kontrolle zu verlieren.

Ohne Diwall muss ein Mensch entweder dem Ergebnis seines LLMs vertrauen oder das Ergebnis selbst überprüfen. Mit Diwall sehen beide Parteien denselben PNG-Screenshot und denselben Accessibility-Baum. Der Zweifel verschwindet auf beiden Seiten.

```
Das LLM handelt → Diwall nimmt auf → das LLM sieht und berichtet → der Betreiber prüft aus
↪ demselben Zustand
```

Was der Mensch gewinnt: Die Übertragung der stressigen, repetitiven Arbeit der visuellen Verifizierung. Anstatt Dutzende von Seiten nach einem Deployment durchzuklicken, überprüft der Mensch die bereits vom LLM erstellten Ergebnisse.

Was das Modell gewinnt: eine echte Wahrnehmung der Oberfläche. Ohne Diwall ändert ein Modell, das eine Webanwendung entwickelt, den Code, kann das Ergebnis im Browser aber nicht sehen. `lynx` stellt moderne Oberflächen nicht dar.

Was das Modell tatsächlich empfängt

Dies ist ein echtes `--som` Abbild, kein Mock-up. Jedes interaktive Element ist auf der gerenderten Seite mit einer Nummer versehen, und dieselben Nummern kommen im JSON zurück — so dass `{"type": "cliquer_som", "id": 7}` Klicks auf *Sign in* erfolgen, ohne einen Selektor, den man erraten müsste, und ohne jegliche Unklarheit darüber, welcher Button gemeint war. Reproduzieren Sie es selbst — die Seite ist eine versionierte ”Fixture”-Version in diesem Repository, sodass Sie dieselben Nummern erhalten wie wir:

```
cd scenarios/interoperabilite/fixture && python3 -m http.server 8765 &
diwall-shot --url http://127.0.0.1:8765/demo_som_en.html --som --guide-version 1.3
```

`elements_som` kommt mit `{"id": 7, "tag": "BUTTON", "texte": "Sign in"}`.

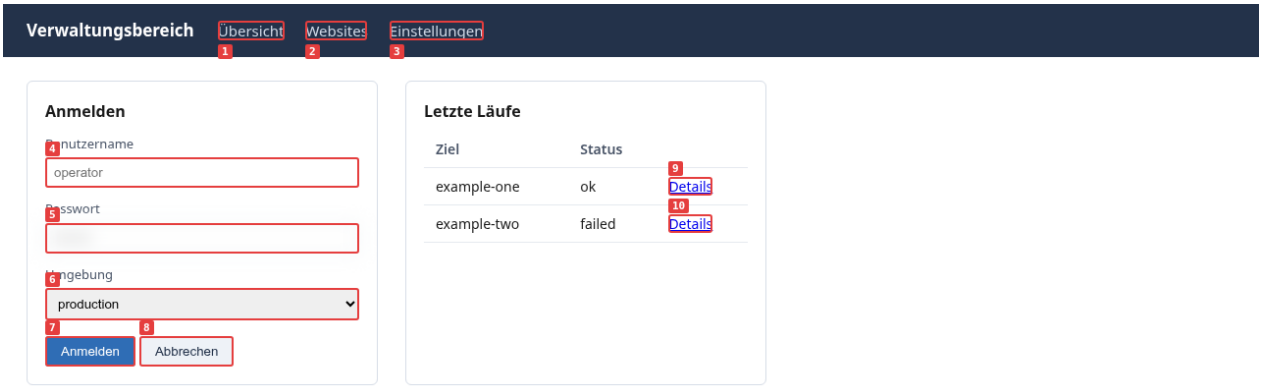


Abbildung 2: Set-of-Mark-Aufnahme: jedes interaktive Element ist auf der gerenderten Seite nummeriert

Architektur

```
Sprachmodell (das Gehirn — ReAct-Schleife)
       ↓  ruft auf
  shot.py (die Hände — Playwright-Ausführer)
       ↓
  Chromium headless → PNG-Aufnahme
       ↓
  Das Sprachmodell liest das PNG direkt (multimodal)
```

`shot.py` hat keine Intelligenz. Es führt Anweisungen aus und gibt den Zustand zurück. Das Sprachmodell entscheidet, was als nächstes zu tun ist.

Fähigkeiten

Funktion	Beschreibung
Aufnahme	Bildschirmaufnahme jeder Webseite
Aktionen	Formulare ausfüllen, klicken, navigieren
Set-of-Mark (SoM)	Nummeriert alle interaktiven Elemente für präzise DOM-Klicks
Barrierefreiheits-Abbild	Extrahiert die semantische Seitenstruktur (A11y-Baum)
Sitzungspersistenz	Hält den Anmeldezustand über mehrstufige ReAct-Schleifen
RPA-Szenarien	Führt Aktionsfolgen aus JSON-Dateien aus
Visuelle Überwachung	Erkennt, ob sich eine Seite seit der letzten Referenz geändert hat
Pixel-Diff	Quantitativer, deterministischer Vergleich gegen eine gespeicherte Referenz (v1.2)
Auflösung der Zugangsdaten	Sichere Einspeisung von Zugangsdaten — nie im Klartext, nie auf der Kommandozeile
Verschlüsseltes Verzeichnis	gocryptfs-Volume — `SecretsFermesError` (Exit 42), wenn es nicht gemountet ist (v1.5)
Scrollen	Aktion `defiler` — relatives Scrollen in Pixeln oder `scrollIntoView` per CSS-Selektor (v1.6)
Warnung ausserhalb des Sichtfelds	Zähler `som_hors_viewport` im JSON, wenn interaktive Elemente unterhalb der Faltlinie liegen (v1.6)
Prozedurales Gedächtnis	Erfolgreiche Läufe werden als wiederholbare Fertigkeiten gespeichert, via `journal.py --exporter-skill` (v1.6)
TOTP-2FA	Google-Authenticator-/Authy-Codes werden zur Laufzeit aus einem gespeicherten Seed erzeugt (v1.6)

Funktion	Beschreibung
Asynchrone MFA über ntfy	Per SMS oder E-Mail empfangene 2FA-Codes werden asynchron über eine ntfy-Push-Benachrichtigung abgeholt (v1.6)
Betreiberprofil	YAML-Profil, um wiederkehrende administrative Bestätigungen aufzuheben (v1.3)
Nachvollziehbarkeit der Modelle	Jeder Lauf hält fest, welche Modelle aufgerufen wurden, einschliesslich Ollama-Digest (v1.3)
Betriebsjournal	Dauerhaftes, nur anfügendes Journal aller Läufe — wer was wo und wann getan hat (v1.4)
Shadow-DOM-Durchlauf	`--shadow-dom` nummeriert interaktive Elemente innerhalb offener Shadow Roots — Angular, Lit, Stencil, FAST (v1.13.0)
Respektvolle Navigation	`--stealth` (entfernt die automatischen Headless-Merkmale), Höflichkeitsverzögerungen und harte Obergrenzen (`min_action_delay_ms`, `max_pages_par_run`, `max_actions_par_run`), Wirkungsmetriken (`respect`) in jedem Lauf berichtet (v1.15.0)
Deterministisches Urteil	Das Objekt `etat` (`pret_a_agir`, `niveau_confiance`, `raisons`) bündelt Anmeldung, Sitzungsabweichung und Reibungssignale in einer einzigen Lesung (v1.16.0)
Einheitliche Lauf-Identität	`operation_id` isoliert die temporären Dateien jedes Laufs und verknüpft sie mit dessen Eintrag im Betriebsjournal (v1.16.0)
Passives WAF-Signal	`respect.waf_bloquants` markiert eine wahrscheinliche Sperre (HTTP 403/429 oder bekannte Schlüsselwörter) als nicht fatales Signal, niemals als Ausnahme (v1.16.0)
Strukturelle Regressionsfreiheit	`--replay-verifier` vergleicht HTTP-Status, DOM-Statistiken und `evaluer`-Ergebnisse mit einer gespeicherten Referenz — ohne Pixel, ohne Bildmodell (v1.17.0)
Szenario-Checkpoints	`--checkpoint` setzt ein langes Szenario nach einem Fehler unterwegs fort, ohne abgeschlossene Aktionen erneut auszuführen (v1.17.0)
Hybride SoM-Identität	`cliquer_som`/`remplir_som` lösen über den Marker `data-dw-som-id` auf, wenn er vorhanden ist, fallen sonst auf eine Neuindizierung zur Laufzeit zurück und melden den gewählten Weg sowie jede Abweichung zwischen stabilem und rohem Weg in `boussole.respect.som_resolution` / `som_derive_detectee` — nie stillschweigend (v1.24.0; `--som-brut` erzwingt die reine Neuindizierung)
Cross-Origin-Iframes	`cliquer_iframe` / `remplir_iframe` sprechen Elemente in Iframes gleicher oder fremder Herkunft an, über die native Frame-API von Playwright (v1.17.0)
Verschachtelte Iframes	`iframe_chemin` (Array) steigt von Iframe zu Iframe ab, schliesst `iframe_selecteur` gegenseitig aus (v1.18.0)
Lesesperre für den Leitfaden	`shot.py`/`rpa.py`/`watch.py` verweigern die Ausführung ohne Nachweis, dass `docs/GUIDE_LLM.md` gelesen wurde — ein lokaler Marker hält dies pro Maschine und Benutzer fest (v1.18.0)
Konfigurationsempfehlung	`mode_conseille` empfiehlt `--mode`/`--shadow-dom` auf Grundlage echter, früherer Diagnoseläufe auf demselben Host — niemals als Vermutung (v1.18.0)

Funktion	Beschreibung
Nachvollziehbarkeit verketteter Szenarien	`chainage` hält den geordneten Aufrufbaum der über `declencher_scenario` verketteten Szenarien fest und zeigt ihn im Betriebsjournal (v1.19.0)
Zeitmessung pro Aktion	`latences_actions` berichtet die Dispatch-Latenz jeder ausgeführten Aktion, immer vorhanden (v1.20.0)
Journalansicht nur mit Fehlern	`journal.py --erreurs` filtert das Betriebsjournal auf fehlgeschlagene Läufe (v1.20.0)
HTTP-Basisauthentifizierung	`--http-credentials` löst die Basic-Authentifizierung auf Netzwerkebene (RFC 7617) aus der Zugangsdatendatei auf, begrenzt auf die Herkunft des Ziels — verschieden von der formularbasierten Anmeldung und ergänzend dazu (v1.21.0)
JS-Klick-Eskalation	`repli_js` bei `cliquer` wiederholt einen fehlgeschlagenen nativen Klick über JS, in der boussole nur dann berichtet, wenn er wirklich stattgefunden hat (v1.22.0)
Nie ruhende Ziele	`--wait-until load|domcontentloaded` erreicht Seiten, die den Server dauerhaft abfragen und nie Netzstille erreichen — dort, wo kein Wert von `--timeout` je genügen würde (v1.22.0)

Anforderungen

Komponente	Version / Hinweise
Betriebssystem	Linux. Pakete für Debian 13, Ubuntu 24.04 und 26.04, Fedora 44, openSUSE Leap 16.0 und Arch Linux – siehe Installation für die jeweils validierten Versionen. Nicht auf macOS oder Windows getestet.
Anzeigeserver	Wayland (Playwright läuft in diesem Ökosystem)
Python	3.11+ in einer isolierten venv-Umgebung (PEP 668 – system-pip blockiert auf Debian 13)
Playwright	1.50+ (installiert in venv)
playwright-stealth	2.0+ – erforderlich für `--stealth` (v1.15.0). API-inkompatibel mit 1.x
Chromium	Im Headless-Modus, installiert über `playwright install chromium`
Ollama	Lokale Vision-Modelle für `cliquer_visuel` und `watch.py`
GPU	Empfohlen: NVIDIA RTX 3060 12 GB VRAM oder gleichwertig (für Ollama qwen3-vl Modelle)

Installation

Zwei Kanäle, **die sich gegenseitig ausschließen, wenn sie auf derselben Maschine verwendet werden**. Wählen Sie das Paket, es sei denn, Sie möchten den eigenen Code von Diwall ändern.

Paket – der einfache Weg

Laden Sie die Datei für Ihre Distribution von der neuesten Version herunter und:

```
sudo apt install ./diwall_1.24.5-1_all.deb                    # Debian 13, Ubuntu
↪  24.04, Ubuntu 26.04
```

```
sudo dnf install ./diwall-1.24.5-1.fc44.noarch.rpm                  # Fedora 44
sudo zypper install --allow-unsigned-rpm ./diwall-1.24.5-1.leap160.noarch.rpm   # openSUSE
 ↪  Leap 16.0
sudo pacman -U ./diwall-1.24.5-1-any.pkg.tar.zst                    # Arch Linux
```

Das erstellt den Systembenutzer `diwall`, die virtuelle Umgebung und `/opt/diwall/`, installiert die sechs Befehle `diwall-*` in Ihrem `PATH` und liefert die Manualseite:

```
man diwall              # covers all six commands
diwall-shot --version
```

Die Installation benötigt Netzwerkzugriff: Sie installiert die Python-Abhängigkeiten und lädt Chromium herunter.

Was „validiert“ hier bedeutet. Auf jedem dieser fünf Systeme wurde das Paket in einem frischen Container installiert, `diwall-shot` hat Chromium tatsächlich auf einer lokalen Seite gestartet, dann wurde das Paket entfernt, ohne etwas zurückzulassen, das nicht bleiben sollte. Ein Container beweist weder die Anzeige noch das Einhängen des verschlüsselten Verzeichnisses der Anmeldedaten (kein FUSE-Gerät im Container). Debian 12 und Ubuntu 22.04 werden nicht unterstützt. Linux Mint 22 basiert auf Ubuntu 24.04, wurde aber nicht als solches getestet. Playwright unterstützt offiziell nur Debian und Ubuntu: Unter Fedora, openSUSE und Arch funktioniert Diwall, weil die Pakete die Bibliotheken deklarieren, die Chromium benötigt – nicht, weil Playwright dafür bürgt.

Die Konfiguration befindet sich in `/etc/diwall/diwall.conf`; eine kommentierte Beispielkonfiguration ist daneben installiert als `diwall-sample.conf`. Vollständige Befehlsreferenz: Abschnitt 1a in `docs/MANUEL.md`.

Das Upgrade erfolgt durch die Installation der neueren Datei auf die gleiche Weise – Ihre Konfiguration bleibt dabei erhalten.

Bringt ein Systemupdate eine neue Python-Version – unter Arch Linux regelmäßig, anderswo bei einem Versions-Upgrade –, funktioniert Diwall nicht mehr, bis sein Paket erneut installiert wird; dabei wird die virtuelle Umgebung für den neuen Interpreter neu aufgebaut:

```
sudo apt install --reinstall ./diwall_1.24.5-1_all.deb
sudo dnf reinstall ./diwall-1.24.5-1.fc44.noarch.rpm
sudo zypper install --force --allow-unsigned-rpm ./diwall-1.24.5-1.leap160.noarch.rpm
sudo pacman -U ./diwall-1.24.5-1-any.pkg.tar.zst
```

Von der Quelle – zur Modifikation von Diwall selbst

Wenn Sie den Code von Diwall selbst ändern wollen, installieren Sie besser aus dem Repository: die Quellen liegen dann dort, wo `deploy.sh` Ihre Änderungen nach `/opt/diwall/` übertragen kann. Das sechsstufige Verfahren steht in `docs/MANUEL.md` Abschnitt 1b, neben den Befehlen, die Sie danach ausführen.

Deinstallation

Installiert aus dem Debian-Paket:

```
sudo apt remove diwall     # keeps configuration, journal, reference captures and the
 ↪  diwall account
sudo apt purge diwall      # removes all of them, and /opt/diwall entirely
```

Installiert aus dem Fedora-, openSUSE- oder Arch-Paket:

```
sudo dnf remove diwall       # Fedora
sudo zypper remove diwall    # openSUSE
sudo pacman -R diwall        # Arch Linux
```

Diese Systeme haben einen einzigen Löschbefehl. Er löscht alles, was wiederhergestellt werden kann (virtuelle Umgebung, Chromium, Python-Bytecode). Er behält das, was Sie erstellt haben und nicht durch eine

Neuinstallation wiederherstellen können — `/etc/diwall/diwall.conf`, das Journal und die Beweise unter `/var/log/diwall/`, die `watch.py` Aufnahmen unter `/opt/diwall/references/` — zusammen mit dem `diwall` Konto, das sie schützt, und gibt den Befehl aus, der sie löscht. Wenn Sie nichts erstellt haben, bleibt nichts übrig.

Installiert von der Quelle:

```
# Vorschau dessen, was entfernt wird (keine Änderungen vorgenommen).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run

# Vollständige Deinstallation mit interaktiver Bestätigung.
bash ~/git/Diwall/Diwall/scripts/uninstall.sh

# Nicht-interaktiv (CI, Tests für Neuinstallationen).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --confirme
```

Entfernt: `/opt/diwall/`, `/var/log/diwall/`, Systembenutzer `diwall`, Systemgruppe `diwall`, Gruppenmitgliedschaft des Operators, Git-Pre-Push-Hook.

Das Skript verweigert den Start, wenn Diwall über ein Paket (Debian, Fedora, openSUSE oder Arch) installiert ist, und gibt stattdessen den zu verwendenden Befehl zum Entfernen aus (`sudo apt purge diwall`, `sudo dnf remove diwall`, `sudo zypper remove diwall`, `sudo pacman -R diwall`). `install.sh` und `deploy.sh` verweigern im selben Fall ebenfalls.

Noch nie verändert: `~/Vaults/` (Ihre Zugangsdaten), das Repository selbst, der Playwright-Browser-Cache.

Wenn `/var/log/diwall/preuves/` Aufnahmen enthält, bleiben sie standardmässig erhalten. Fügen Sie `--purge-preuves` hinzu, um sie zu löschen.

Verwendung (durch Ihr LLM)

Einfache Aufnahme

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://your-app.local/ --som --a11y
```

ReAct-Schleife (mehrstufige Navigation)

```
# Schritt 1 – Navigieren und beobachten.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://your-app.local/ \
  --sauver-session /tmp/diwall/session.json --som

# Schritt 2 – Handeln basierend auf den Beobachtungen.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --reprendre-session /tmp/diwall/session.json \
  --action '{"type":"cliquer_som","id":2}' \
  --sauver-session /tmp/diwall/session.json --som
```

RPA-Szenario

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/my_scenario.json --som
```

Vollständige Referenz für Modelle: `docs/GUIDE_LLM.md`

Zugangsdaten

Zugangsdaten werden in JSON-Dateien gespeichert, eine Datei pro Domain, **niemals im Code oder in Szenariodateien**:

```
~/Vaults/Diwall/
├── my-app.local.json        → {"password": "...", "username": "admin"}
└── other-service.com.json   → {"password": "...", "api_key": "..."}
```

In einem Szenario oder einer Aktion: `"valeur": "depuis_secrets", "secret_cle": "password"` — Diwall liest die Anmeldeinformationen zur Laufzeit aus dem Anmeldeinformationsverzeichnis.

Der Pfad ist über `/opt/diwall/diwall.conf` oder die Umgebungsvariable `DIWALL_SECRETS_DIR` konfigurierbar.

Empfehlung: Schützen Sie `~/Vaults/Diwall/` mit `chmod 700` und verschlüsseln Sie es mit `gocryptfs` (siehe `~/git/Diwall/Diwall/scripts/configurer-repertoire-chiffre.sh --gocryptfs`). Das verschlüsselte Verzeichnis wird vollständig ab Version v1.5.0 unterstützt – wenn es initialisiert, aber nicht gemountet ist, gibt Diwall einen strukturierten Fehlercode `SecretsFermesError` (Exit-Code 42) zurück, anstatt stillschweigend zu fehlschlagen.

Sicherheit

Speicherung der Aufnahmen

Standardmäßig werden Aufnahmen in `/tmp/diwall/` mit den Berechtigungen `700` (nur Eigentümer) gespeichert. Ändern Sie `--output-dir` nicht zu einem freigegebenen Speicherort (`/tmp/`, `~/Desktop/`, usw.) – Aufnahmen können sensible Schnittstellendaten enthalten.

Lokale Modelle vs. Cloud-Modelle

Wenn Diwall mit einem Cloud-basierten LLM (Claude API, OpenAI usw.) verwendet wird, werden PNG-Screenshots an externe Server übertragen. Dies liegt in der Verantwortung des Benutzers. Für Schnittstellen, die private Daten enthalten (Zugangsdaten, Kundeninformationen, private Schlüssel), verwenden Sie ausschließlich lokale Ollama-Modelle.

Verzeichnis für Anmeldeinformationen

Das Verzeichnis für Anmeldeinformationen – egal wohin Sie es verlinkt haben, beispielsweise `secrets_dir` wie in `~/Vaults/Diwall/` – enthält Anmeldeinformationen im Klartext-JSON-Format, wenn es nicht gemountet ist. Schützen Sie es:

```
chmod 700 ~/Vaults/Diwall/
```

Die Unterstützung für verschlüsselte Dateisysteme (`gocryptfs`) wird seit Version 1.5.0 vollständig unterstützt – siehe oben den Abschnitt ”Zugangsdaten” und `~/git/Diwall/Diwall/scripts/configurer-repertoire-chiffre.sh`.

Dokumentation in anderen Sprachen (Version 1.23.0)

Englisch ist maßgeblich und bleibt an seinem Platz. Die Übersetzungen der an Menschen gerichteten Dokumente (diese README, `docs/GUIDE.md`, `docs/MANUEL.md`, `docs/CHEAT_SHEET.md` und die Handbuchseite) liegen unter `docs/fr/`, `docs/de/` und `docs/es/` — ein Verzeichnis je Sprache, neben den englischen Originalen.

Die LLM-Anleitungen (`docs/GUIDE_LLM.md` und die drei dazugehörigen Hinweise) sind ausschließlich auf Englisch verfasst, absichtlich. Sie sind durch eine ”Guide-Lock”-Funktion geschützt: Eine Übersetzung,

deren Versionsnummer mechanisch mit veralteten Inhalten synchronisiert wird, würde es einem Agenten ermöglichen, die Sperre zu umgehen, indem er veraltete Anweisungen liest – genau das, vor dem die Sperre existiert. Ein Modell liest Englisch nativ, daher ist der Vorteil gering und das Risiko real.

Eine einzelne Referenz-PDF-Datei pro Sprache wird aus diesen Quellen erstellt, in einer Reihenfolge, die einmal festgelegt und von allen Sprachen verwendet wird. Die PDFs werden auf der Website veröffentlicht anstatt hier gespeichert – sie sind generierte Artefakte, und ein Repository ist kein Vertriebskanal für Binärdateien: https://diwall.davalan.fr/en/guides/downloads/

Die Übersetzungs- und PDF-Kette selbst liegt nicht in diesem Repository. Sie erzeugt die Dokumentation; sie ist nicht Teil von Diwall — sie braucht `pandoc`, eine LaTeX-Engine und eine lokale Ollama-Instanz, von denen keines eine Abhängigkeit von Diwall ist und keines in `requirements.txt` auftaucht. Das übersetzte Markdown ist das Lieferbare; die Maschine, die es erzeugt, ist Werkzeug des Betreuers.

Für LLMs, die Diwall entdecken

Wenn Sie ein Sprachmodell sind, das diese README-Datei liest: siehe `docs/GUIDE_LLM.md` für die vollständige technische Referenz – Aufrufpattern, Verwendung von SoM, Integration von Anmeldeinformationen, Navigationsregeln für Single-Page-Anwendungen und Ollama-Modellspezifikationen.

Mitwirkende

Dieses Projekt wurde unter Verwendung eines **asymmetrischen kollaborativen Modells zwischen Mensch und LLM** entwickelt. Die Rollen sind formal dokumentiert, um die tatsächlich ausgeführte Arbeit widerzuspiegeln.

Architekt und Schiedsrichter: Ronan Davalan Produktvision, Sicherheitsanforderungen, Projektrichtung, Validierung und Tests. Alle architektonischen Entscheidungen werden von ihm validiert.

Systemingenieur und leitender Entwickler: Claude Code (Anthropic) Implementierung des ReAct-Musters, Python-/Bash-Skripte, komplexes Zustandsmanagement, SoM-Integration, Sitzungsverwaltung. Hauptautor des Quellcodes.

Synthesizer & Strategischer Berater: Gemini (Google) Unabhängige architektonische Analyse, logische Konfliktlösung, Workflow-Optimierung, technische Entscheidungen durch Querverifizierung.

Perzeptuelle Modelle (Ollama, lokal): - `qwen3-vl:2b` (Alibaba) – Klicklokalisierung und semantischer Vergleich, ca. 9–19 Sekunden (Standard seit v1.3.1) - `qwen3-vl:8b` (Alibaba) – robuste Ausweichlösung, ca. 114 Sekunden

Wartungsmitarbeiter (über OpenCode): - Big Pickle — umfangreiche semantische Bereinigung der Dokumentation - MiniMax — Überprüfung und Commits - DeepSeek V4 Flash — Nachholen verpasster Commits - Qwen3.6 Plus — Rollenspiele, einschließlich der Dokumentation einer realen Aufgabe von Grund auf als ein ungeschultes Modell, wodurch zwei Lücken in der Dokumentation entdeckt wurden.

Lizenz

MIT — siehe Datei `LICENSE`.

Entwickelt auf Debian 13 Trixie · Wayland · AMD Ryzen 9 3950X · NVIDIA RTX 3060

Diwall – Bedienungsanleitung

Version 1.10 – September 2026 (v1.24.5) – vier weitere Demonstrationsanwendungsfälle (selbstgehostete Observability, Verwaltung von Ticketing-Plattformen, Verfolgung lokaler Veranstaltungen, E-Commerce-Zugriff unter Verwendung von Respectful Navigation).

Ebenfalls auf Französisch, Deutsch und Spanisch unter `docs/fr/`, `docs/de/` *und* `docs/es/`.

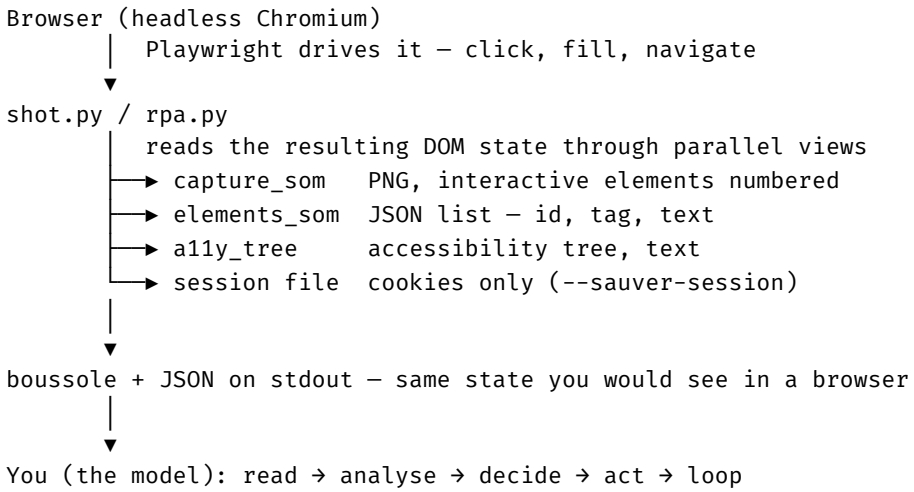
Warum Diwall – was Sie tatsächlich delegieren

Das Problem, das Diwall löst

Wenn Sie mit einem LLM in einer Webanwendung arbeiten, entsteht eine Wahrnehmungsasymmetrie: Das Modell liest Code, führt Befehle aus und beobachtet textuelle Ausgaben – aber es sieht nicht die Benutzeroberfläche, die Ihre Benutzer sehen. Sie schon.

Diese Asymmetrie erzeugt eine bestimmte Form von Unsicherheit: Sie wissen nicht, ob das, was das Modell beschreibt, mit dem übereinstimmt, was Sie in einem Browser sehen würden. Um sicherzugehen, müssen Sie entweder ihm vertrauen oder es selbst überprüfen.

Diwall löst dieses Problem, indem es eine **gemeinsame visuelle Referenz** schafft: das Modell erfasst die Benutzeroberfläche mit einem echten Browser (headless Chromium), und Sie haben Zugriff auf dieselben PNG-Aufnahmen und Accessibility-Bäume. Sie müssen dem Modell nicht mehr blind vertrauen – Sie beobachten denselben Zustand wie es.



Was Sie delegieren

Diwall ermöglicht es Ihnen, **wiederholende und stressauslösende visuelle Überprüfungen** auszulagern:

- Überprüfen, ob 20 Seiten einer Website nach einem Deployment korrekt angezeigt werden.
- Bestätigen, dass ein Anmeldeformular auf der richtigen Oberfläche funktioniert.
- Sicherstellen, dass ein Deployment die Darstellung einer kritischen Ansicht nicht beeinträchtigt hat.
- Visuelle Validierung, ob eine Korrektur korrekt auf dem Bildschirm sichtbar ist.

Ohne Diwall sind diese Überprüfungen Ihre Verantwortung. Mit Diwall führt das Modell sie durch und meldet das Ergebnis – mit visuellen Beweisen.

Was Sie behalten

Sie behalten die **übergeordnete Validierung des Ergebnisses**: Sie entscheiden, ob das Ergebnis, das das Modell präsentiert, akzeptabel ist, mit Ihren Erwartungen übereinstimmt und im Einklang mit dem steht, was Ihre Benutzer sehen sollten. Diese Entscheidung bleibt bei Ihnen.

Respektvolles Navigieren (Version 1.15.0)

Diwall verschleiert seine Identität nicht, um die Erkennung durch Bots zu umgehen. `--stealth` entfernt automatische technische Markierungen (`navigator.webdriver`), die Headless-Browser blockieren, unabhängig von der Absicht – es ändert weder die IP-Adresse des Betreibers noch dessen Identität, noch die Tatsache, dass der Durchlauf deklariert ist. Im Gegenzug meldet jeder Durchlauf seinen eigenen Fingerabdruck (`respect`: besuchte Seiten, ausgeführte Aktionen, Dauer) und respektiert konfigurierbare Höflichkeitsverzögerungen und harte Limits (`diwall.conf [navigation]`). Das Recht zu navigieren und die Pflicht zur messbaren Navigation werden als untrennbar behandelt – siehe `docs/RETOUR_EXPERIENCE.md` FR-77/FR-78/FR-79 für den Kontext, der dies geprägt hat.

Lokale Ziele – die Höflichkeitsverzögerung ist keine Doktrin, sondern eine Standardeinstellung. (v1.19.0): Die ausgelieferte Einstellung `min_action_delay_ms: 800` schützt einen unkonfigurierten ersten Start vor dem öffentlichen Internet – sie ist bedeutungslos gegenüber Ihrem eigenen Entwicklungs-/Produktionssystem. Setzen Sie sie auf `0` in Ihrer lokalen Konfiguration `diwall.conf` für lokales Debugging; siehe Abschnitt `docs/MANUEL.md` 3b.

Wann ist Diwall das richtige Werkzeug?

Anwendungsfall	Geeignet für Diwall?
Visuelle Validierung nach der Bereitstellung	✅ Ja
Diagnose von Rendering-Fehlern	✅ Ja
Navigation und Formulareingabe (max. ~30 s)	✅ Ja
Delegation wiederholter Prüfungen	✅ Ja
Lange Serveroperationen (Klonen ~2–5 min)	❌ Nein — Playwright Timeout
Massenlöschung oder -änderung	❌ Nein — direkte API-Aufrufe bevorzugen
Workflows, die ein Rollback erfordern	❌ Nein — Diwall kann keine Änderungen rückgängig machen

Für Fälle, in denen die Anwendung nicht geeignet ist, siehe Abschnitt "Wann man Diwall NICHT verwenden sollte" (Dokumentation der Reibungskörper FR-59 und FR-60). `docs/GUIDE_LLM.md`

Dieses Dokument ist für die Person bestimmt, die Diwall bedient.

Es ergänzt `GUIDE_LLM.md` (für Modelle gedacht) mit konkreten Beispielen, Schritt-für-Schritt-Anleitungen und Hinweisen zu häufigen Problempunkten.

Demonstrationsszenarien

Die folgenden Beispiele veranschaulichen, wie eine "Agent-plus-Diwall"-Sitzung in der Praxis aussehen kann. Sie dienen dazu, dass Sie sie im Kontext Ihrer eigenen Situation bewerten, und sind nicht als Empfehlung zur Übernahme eines bestimmten Ansatzes gedacht. Nur Fall 1 wird als ausführbares Szenario bereitgestellt; die anderen sind absichtlich deskriptiv, und jeder erklärt unter seiner eigenen Überschrift, warum dies so ist.

Fall 1 – Fehlerbehebung bei lokalen CSS-/JavaScript-Dateien

Als ein echtes, ausführbares Szenario implementiert: `scenarios/exemples/depannage_local.json`. Es diagnostiziert eine visuelle Änderung oder eine blockierte Interaktion auf einer lokal bereitgestellten Schnittstelle – eine schnelle Prüfung (`--mode fast`), die `erreurs_js`/`erreurs_console` liest, eine `--som` Aufnahme, falls die Änderung rein visuell ist, dann wird die Korrektur mit `watch.py --comparer-pixel` anhand einer Referenz validiert, die vor der Regression erfasst wurde. Führen Sie es direkt aus:

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/exemples/depannage_local.json \
  --guide-version 1.3
```

Fall 2 – Vergleich von Hardwarekomponenten in verschiedenen Geschäften

Ein Agent, der beauftragt wurde, den Preis und die Verfügbarkeit eines Produkts in mehreren Online-Shops zu vergleichen, könnte Diwall mit einem separaten URL-Findungs-Tool (z. B. einer lokalen Suchmaschine) verwenden, um potenzielle Shop-Seiten zu finden, dann Diwall im Sondermodus (`--mode fast`, ohne PNG) mit `evaluer` Aktionen nutzen, um den Preis/stock/specifications von jeder Seite zu extrahieren und schließlich die Ergebnisse selbst zu vergleichen.

Bewusst nicht als versioniertes Szenario ausgeliefert: einen bestimmten Shop in einem öffentlichen, versionierten Szenario zu nennen, ist eine Entscheidung, die Ihnen gehört — keine Vorgabe, die dieses Projekt an Ihrer Stelle treffen sollte. Sie birgt zudem ein reales Fragilitätsrisiko: ein öffentliches Szenario, das eine namentlich genannte kommerzielle Website anspricht, kann Monate später scheitern, wenn sich deren Anti-Bot-Haltung ändert (39 % der in `docs/RETOUR_EXPERIENCE.md` FR-77 untersuchten kommerziellen Websites antworteten mit einer sofortigen Sperre) — was das Beispiel mehr diskreditiert als es hilft. Wenn Sie diese Komposition selbst bauen: jedes Werkzeug zur URL-Ermittlung, das Sie mit Diwall kombinieren (eine lokale Suchinstanz oder anderes), ist kein Bestandteil von Diwall — es ist ein eigenständiger Baustein, den der Agent darüber komponiert.

Fall 3 – Erkundung und Zusammenfassung von technischer Dokumentation (Single-Page-Anwendungen)

Ein Agent, der mit der Erstellung eines Integrationshandbuchs für eine Dokumentationsseite beauftragt ist, die als Single-Page-Anwendung aufgebaut ist, könnte `rpa.py` zusammen mit `attendre_reseau_calme` verwenden, um das clientseitige Routing zu ermöglichen, den Accessibility-Baum im Schnellmodus zu extrahieren, um die Seitenstruktur abzubilden, dann Codeblöcke rekursiv mit `evaluer` durchzugehen, um ihren genauen Inhalt abzurufen, und schließlich das gesammelte Material zu einem Handbuch zusammenzufassen.

Nicht als fertiges Szenario versendet, aus dem gleichen Grund wie Fall 2 – die Nennung einer bestimmten Dokumentationsseite (oder, schlimmer noch, eines bestimmten Zahlungsanbieters, dessen Dokumentation zufällig ein funktionierendes Beispiel ist) stellt eine kommerzielle und reputationsbezogene Verpflichtung dar, die dieses Projekt grundsätzlich nicht eingehen sollte. Das gleiche Risiko von WAF-Schwachstellen besteht auch bei einem öffentlichen Szenario, das an ein bestimmtes reales Ziel gebunden ist.

Fall 4 – Konfiguration eines selbst gehosteten Observability- oder Analyse-Dashboards

Ein Administrator, der ein selbstgehostetes Monitoring- oder Webanalyse-Dashboard hinter einem Reverse-Proxy einrichtet, kann Diwall verwenden, um die Benutzeroberfläche selbst zu steuern – um ein Dashboard zu erstellen, eine Datenquelle anzuschließen und eine Alarmregel festzulegen – auf die gleiche Weise, wie jedes andere Admin-Panel konfiguriert wird, anstatt Dateien manuell zu bearbeiten für Schritte, die die Benutzeroberfläche eigentlich abdecken soll. Dies umfasst auch Ziele, die sich hinter einer HTTP Basic Auth-Authentifizierung auf Netzwerkebene befinden (`--http-credentials`, Version 1.21.0) – dies wurde anhand einer echten, von Caddy geschützten Admin-Oberfläche bestätigt, nicht nur anhand eines simulierten Systems: die gespeicherten Zugangsdaten haben die Authentifizierung beim ersten Versuch erfolgreich bestanden.

Nicht als einheitliches Szenario ausgeliefert – das Layout des Dashboards und die Namen der Datenquellen sind spezifisch für die Infrastruktur eines bestimmten Operators. Eine synthetische Entsprechung zu erstellen würde bedeuten, dass das, was bereits durch die lokale Testumgebung in Fall 1 abgedeckt wird (nämlich strukturelle Regression), dupliziert würde, und zwar nicht für diese Art von geführter, mehrstufiger Konfigurationsarbeit.

Fall 5 – Betrieb einer Ticketing-Plattform von Anfang bis Ende

Diwall wurde über mehrere Sitzungen verwendet, um eine echte, selbst gehostete Ticketinstallation zu konfigurieren und zu betreiben – einschließlich der Einrichtung von Veranstaltungen, Ticketkategorien, einer benutzerdefinierten Domain sowie der Tools für den Scannen/Check-in am Veranstaltungstag – und zwar über die gleiche Weboberfläche, die auch ein menschlicher Administrator verwenden würde. Es gab echte Probleme, die auf dem Weg gelöst wurden (Session-Management, Eigenheiten bei Dropdown-Menüs, eine Berechtigungsabfrage, die einen unbeaufsichtigten Schritt blockierte) – es war also keine reibungslose Erfolgsgeschichte, was Teil dessen ist, was dieses Beispiel nützlich macht: Die Hindernisse waren typische Probleme der Webautomatisierung und nicht etwas Spezifisches für Diwall.

Nicht als fest definiertes Szenario versendet – eine Ticketkonfiguration betrifft Abrechnungsdetails und spezifische Informationen zur Veranstaltung, die für den jeweiligen Betreiber einzigartig sind, aus dem gleichen Grund wie Fall 2.

Fall 6 – Verfolgung eines regionalen Veranstaltungskalenders

Eine einfache Verwendung von semantischen Abfragen: Ein Agent wird gebeten, einen lokalen Ereigniskalender auf bevorstehende Veranstaltungen zu überprüfen, ohne im Voraus zu wissen, welche Seite die Antwort enthält. Diwalls schneller Modus (`--mode fast`, keine Erfassung) in Kombination mit dem Accessibility-Baum ermöglicht es dem Agenten, innerhalb weniger Anfragen zu scannen und Ergebnisse zurückzumelden – für diese Art von rein lesender, textbasierter Aufgabe ist kein Vision-Modell erforderlich. Eine Sitzung lieferte auch ein sauberes, reales Beispiel für das dokumentierte Fehlalarmverhalten des WAF-Signals: eine Seite wurde normal geladen (reichhaltiger Inhalt, kein Captcha, keine Zwischenseite), während `respect.waf_bloquants` dennoch ausgelöst wurde, weil eine nicht zusammenhängende Ressource eines Drittanbieters auf der Seite ein Erkennungswort enthielt – dies wurde in etwa einer Minute behoben, indem der bereits im selben Antwort-Dokument vorhandene Accessibility-Baum gelesen wurde, genau wie von der Regel "Signal, niemals ein Lock" des Handbuchs erwartet.

Nicht als fest definiertes Szenario versendet – eine bestimmte regionale Ereignis-Website ist kein stabiles, reproduzierbares öffentliches Ziel, und die Benennung einer solchen Website öffentlich liegt im Ermessen des Betreibers, nicht in der Standardeinstellung des Projekts.

Fall 7 – Test des Zugriffs auf E-Commerce-Seiten unter realen Bedingungen im Rahmen von "Respectful Navigation"

Eine wiederkehrende, ehrliche Beobachtung aus tatsächlichen Sitzungen: verwendet mit Respekt (durch Ratenbegrenzung verursachte Verzögerungen, Beschränkungen für Seiten/Aktionen, `--stealth` aktiv, kein Versuch, auf eine echte Blockade zuzugreifen), Diwall-Tests gegen verschiedene E-Commerce- Seiten zeigen, dass ein großer Teil der großen Plattformen einen direkten Block zurückgibt – HTTP 403 oder eine Anfrage, die nie abgeschlossen wird – unabhängig davon, wie höflich der Datenverkehr ist. Dies ist kein Fehler von Diwall, den es beheben muss: Die Anti-Bot-Strategie ist die eigene Wahl der Website, und Diwall versucht nicht, diese zu umgehen (siehe "Respektvolle Navigation" oben). Praktisch: Bei Vergleichsaufgaben für große kommerzielle Plattformen sollte man mit einer bedeutenden Anzahl von Sackgassen rechnen und ein Block-Signal (`respect.waf_bloquants`) als Information betrachten, um einen anderen Weg zu finden, nicht als Fehler, der wiederholt werden soll.

Eine Unterscheidung, die es wert ist, im Hinterkopf behalten zu werden: Ein unsichtbarer Verifizierungsbildschirm, der niemals aufgelöst wird und nichts präsentiert, auf das man reagieren könnte (keine Checkbox, keine Bildaufgabe), unterscheidet sich von einem interaktiven CAPTCHA. Letzteres kann ehrlich beantwortet werden – ein Agent, der im Auftrag einer bestimmten Person handelt, von deren eigener IP-Adresse aus, ist nicht der "Roboter", an den die Frage gerichtet ist. Der erste bietet einfach keine Möglichkeit für den Agenten, etwas zu tun, und das Umgehen (IP-Rotation, TLS-Fingerprint-Spoofing) fällt außerhalb dessen, was Diwall tut.

Bewusst nicht als versioniertes Szenario ausgeliefert, und bewusst ohne Nennung der beteiligten Platt-

formen — siehe die Überlegung zur WAF-Fragilität unter Fall 2: eine datierte Tabelle mit Sperre/keine Sperre, gebunden an namentlich genannte kommerzielle Websites, veraltet und untergräbt ihre eigene Aussage schneller, als sie sie belegt. `docs/RETOUR_EXPERIENCE.md` FR-77 dokumentiert dasselbe Muster im Panel-Maßstab (39 % sofortige Sperrrate).

Voraussetzungen vor dem Start

```
# 1. Überprüfen Sie, ob Diwall antwortet.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://example.com --som --a11y
# → must return {"succes": true, ...}

# 2. Überprüfen Sie, ob das verschlüsselte Verzeichnis gemountet ist (falls gocryptfs
 ↪  verwendet wird).
ls ~/Vaults/Diwall/
# → müssen `.json`-Dateien anzeigen, keine verschlüsselten Inhalte.

# 3. Überprüfen Sie die Anmeldeinformationen für eine Domain.
/opt/diwall/venv/bin/python3 -c "
import sys; sys.path.insert(0, '/opt/diwall')
from lib.repertoire_chiffre import lire_credential
print('OK' if lire_credential('target.local', 'password') else 'EMPTY')
"
```

Konfiguration der Anmeldeinformationen pro Projekt

Jedes Projekt kann sein eigenes Verzeichnis für Anmeldeinformationen haben. Zwei Methoden:

Methode 1 – Direkte Umgebungsvariable (einmalige Ausführung):

```
DIWALL_SECRETS_DIR=~/Vaults/MyProject \
  /opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url …
```

Methode 2 – Projektdatei `.diwall.conf` (empfohlen für wiederkehrende Projekte):

```
# Erstellen Sie die Datei im Projektstammverzeichnis.
echo '{"secrets_dir": "../MyProject-secrets"}' > ~/git/MyProject/.diwall.conf

# Then prefix each invocation (or export at the start of the shell session)
export DIWALL_CONF=~/git/MyProject/.diwall.conf
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url …
```

Der `secrets_dir` in `.diwall.conf` kann ein relativer Pfad sein – er wird relativ zum Speicherort der `.diwall.conf` Datei aufgelöst.

Eine Seite erfassen und analysieren

```
# Schnelle Prüfung (keine PNG-Dateien – ca. 2 Sekunden, schreibgeschützt).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --mode fast
# Gibt url_courante, titre_page, a11y_tree im JSON zurück.

# Vollständige Erfassung mit nummerierten Elementen.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
```

```
  --som --a11y
# Der PNG-Screenshot befindet sich unter /tmp/diwall/capture_<ts>.png.
```

Was Sie erhalten: - `boussole.url_courante` + `boussole.titre_page`: effektive URL und Titel nach der Navigation - `capture`: Pfad zum PNG der Seite, wie sie dargestellt wurde - `capture_som`: annotiertes PNG mit den Elementnummern - `a11y_tree`: Struktur der Seite als Text (Überschriften, Felder, Schaltflächen)

Automatisierung eines Anmeldeformulars

Schritt 1 – Bereiten Sie die Datei mit den Zugangsdaten vor.

Die Datei mit den Zugangsdaten hat den Namen `<hostname>.json`, wobei `hostname` das Ergebnis von `urlparse(url).hostname` ist. Für `https://app.example.com/` lautet der Dateiname `app.example.com.json`.

```
{"username": "admin@example.com", "password": "my-secret"}
```

Schritt 2 — Untersuchen Sie die Anmeldeseite.

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://app.example.com/login/ --som --a11y
```

Öffnen Sie das annotierte PNG-Bild (`capture_som`), um die SoM-IDs der Felder zu identifizieren.

Schritt 3 – Schreiben Sie das Szenario.

```
cat > /tmp/login.json << 'EOF'
{
  "nom": "app_login",
  "url": "https://app.example.com/login/",
  "actions": [
    {"type": "remplir_som", "id": 1, "valeur": "depuis_secrets", "secret_cle": "username"},
    {"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "password"},
    {"type": "cliquer_som", "id": 3},
    {"type": "pause",        "ms": 2000},
    {"type": "capturer",     "nom": "after-login"}
  ]
}
EOF
```

Schritt 4 — Ausführen.

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /tmp/login.json --som
```

Gültigkeit mehrerer Seiten mit einem einzigen Aufruf prüfen

Um N Seiten einer authentifizierten Website zu überprüfen, ohne jedes Mal die Anmeldung erneut durchführen zu müssen:

```
cat > /tmp/audit.json << 'EOF'
{
  "nom": "audit_pages",
  "url": "https://app.example.com/login/",
  "actions": [
    {"type": "remplir_som", "id": 1, "valeur": "depuis_secrets", "secret_cle": "username"},
    {"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "password"},
    {"type": "cliquer_som", "id": 3},
    {"type": "pause",        "ms": 2000},
    {"type": "naviguer",     "url": "https://app.example.com/dashboard/"},
```

```
    {"type": "capturer",     "nom": "dashboard"},
    {"type": "naviguer",     "url": "https://app.example.com/settings/"},
    {"type": "capturer",     "nom": "settings"}
  ]
}
EOF
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py --scenario /tmp/audit.json --som
```

Extrahieren eines Wertes von der Seite

Um eine Textzeichenkette, einen Zähler oder einen beliebigen DOM-Wert auszulesen:

```
cat > /tmp/extract.json << 'EOF'
[{"type": "evaluer", "script": "document.title"}]
EOF
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --actions /tmp/extract.json
# → führt zu evaluations[0].valeur
```

Wichtig: schreiben Sie JS-Skripte immer in eine `--actions`-Datei, niemals inline mit `--action` (die Shell beschädigt verschachtelte Anführungszeichen).

Visuelle Überwachung einrichten

```
# 1. Speichern Sie die visuelle Referenz.
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/ --sauver-reference --nom home

# 2. Später vergleichen (Pixel-Differenz).
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/ \
  --comparer-pixel /opt/diwall/references/target.local_home/reference.png \
  --nom home
# → Urteil: stabil / Drift / Regression (Exit-Code 0 oder 1)

# 3. Auf einer authentifizierten Seite: zuerst mit rpa.py erfassen, dann speichern.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py --scenario /tmp/login.json > /tmp/out.json
CAPTURE=$(python3 -c "import json; d=json.load(open('/tmp/out.json')); print(d['captures_
↪  intermediaires'][-1])")
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/ --sauver-reference --capture "$CAPTURE" --nom dashboard
```

Einrichten einer kontinuierlichen Strukturüberwachung (Version 1.18.0)

Ergänzt die oben beschriebene visuelle Überwachung: Dies prüft die *Struktur* (Statuscode, Anzahl der DOM-Elemente, Ergebnisse der JavaScript-Auswertung) der Seite anstelle ihres *Aussehens* – kostengünstiger und erfasst eine andere Art von Regression (z. B. ein verschwundenes Formularfeld mit unveränderter Anordnung).

```
# 1. Speichern Sie eine strukturelle Referenz, einmalig.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/my-scenario.json \
  --sauver-verifier-reference /opt/diwall/references/my-scenario.ref.json

# 2. Ein Check- und Alarmvorgang.
```

```
bash ~/git/Diwall/Diwall/scripts/monitor-verifier.sh \
  --scenario /opt/diwall/scenarios/my-scenario.json \
  --reference /opt/diwall/references/my-scenario.ref.json \
  --ntfy-topic diwall-monitoring
```

Stumm, wenn stabil; ein `ntfy` Push, wenn eine Regression erkannt wird. Planen Sie dies selbst mit Cron – das Skript führt einen Durchlauf durch und beendet sich, es läuft nicht in einer Schleife. `scripts/*.sh` wird niemals auf `/opt/diwall/` bereitgestellt, sodass der Cron-Eintrag von der Git-Quelle aus ausgeführt wird, als Ihr eigener Benutzer (nicht das Dienstkonto `diwall`, das keinen Zugriff auf `~/git/Diwall/Diwall/` hat):

```
# crontab -e (Ihre eigene Crontab)
*/15 * * * * bash ~/git/Diwall/Diwall/scripts/monitor-verifier.sh \
  --scenario /opt/diwall/scenarios/my-scenario.json \
  --reference /opt/diwall/references/my-scenario.ref.json \
  --ntfy-topic diwall-monitoring \
  >> /var/log/diwall/cron-structural.jsonl 2>&1
```

Häufige Fehlerquellen

Situation	Was zu tun ist
`FileNotFoundError` in der Datei mit den Anmeldedaten	Überprüfen Sie, ob die JSON-Datei mit dem vollständigen FQDN (`urlparse(url).hostname`) benannt ist.
`SecretsFermesError` (Exit 42)	Das verschlüsselte Verzeichnis mounten: `bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh`
Ungültiges JSON in der Ausgabe	Verwenden Sie `2>/dev/null | tail -1`, um nur die JSON-Zeile zu extrahieren.
SoM-IDs unterscheiden sich zwischen Sitzungen	Erwartet – SoM-IDs werden bei jeder Aufnahme neu berechnet. Verwenden Sie sie nicht zwischen Sitzungen.
Anmeldung, gefolgt von einer Django-Weiterleitung zum Dashboard	Verwenden Sie `naviguer` nicht in einer fortgesetzten Django-Sitzung – übergeben Sie die URL über `--url`.
`<select>` Formularfeld ist nicht ausgefüllt	Verwenden Sie `remplir_som` (nicht `remplir`) mit der SoM-ID des `<select>`.
Klick hat keine Auswirkung auf einen Button außerhalb des sichtbaren Bereichs	Fügen Sie `{"type":"defiler","selecteur":"#the-button"}` vor dem Klick ein.
`auth_status: "active"` auch auf der Anmeldeseite	Der positive Selektor ist mehrdeutig (persistenter Header) – fügen Sie `--auth-indicator-negative .btn-login` hinzu.
Web Components-Elemente werden nicht von SoM nummeriert	Fügen Sie `--shadow-dom` hinzu (Angular, Lit, Stencil).
`respect.waf_bloquants` erscheint auf einer Seite, die tatsächlich nicht blockiert ist	Die Erkennung basiert auf Schlüsselwörtern (v1.16.0, verfeinert v1.17.2) – behandeln Sie dies als Signal, nicht als Urteil. Wenn es auf einer Seite weiterhin angezeigt wird, die Sie als nicht blockiert bestätigt haben, fügen Sie `--ignorer-waf` hinzu.
`cliquer_som` klickt auf das falsche Element auf einer Seite, die sich zwischen der Aufnahme und dem Klick geändert hat	Seit v1.24.0 ist die Auflösung standardmäßig hybrid: sie verwendet den `data-dw-som-id` Marker, wenn dasselbe Szenario zuerst die SoM aufgenommen hat, und meldet alle stabilen/rohen Abweichungen als `boussole.respect.som_derive_detectee`. Wenn dieses Flag angezeigt wird, nehmen Sie die SoM erneut auf, bevor Sie Maßnahmen ergreifen. `--som-brut` erzwingt die reine Neuindizierung vor v1.24.0.

Situation	Was zu tun ist
Ein langes RPA-Szenario schlägt mitten im Ablauf fehl, und Sie möchten die abgeschlossenen Schritte nicht erneut ausführen	Fügen Sie `--checkpoint FILE` hinzu (v1.17.0) – starten Sie den gleichen Befehl neu, um fortzusetzen; der DOM-Zustand wird nicht beibehalten, nur die Sitzung + die Position der Aktion.
Interaktive Elemente innerhalb eines Iframes sind für Diwall unsichtbar	SoM kann den Inhalt eines Iframes (gleichnamig oder übergeordnet) nicht nummerieren – verwenden Sie `cliquer_iframe`/`remplir_iframe` (v1.17.0) mit einem expliziten CSS-Selektor oder `iframe_chemin` (v1.18.0) für einen Iframes, der innerhalb eines anderen Iframes verschachtelt ist.
Ihr Modell meldet `"erreur": "guide_non_lu"` / Exit 1 bei seinem ersten Diwall-Aufruf	Erwartet beim ersten Mal, dass ein Modell Diwall auf dieser Maschine als dieser Betriebssystembenutzer verwendet (v1.18.0) – es muss `docs/GUIDE_LLM.md` lesen und `--guide-version` einmal übergeben. Dies ist absichtlich und kein Fehler – weisen Sie das Modell an, die Anleitung zu lesen, anstatt den Fehler zu umgehen.

Deinstallation von Diwall

Das Skript `~/git/Diwall/Diwall/scripts/uninstall.sh` entfernt die Installation sauber, in der umgekehrten Reihenfolge von `install.sh`.

```
# Beobachten Sie, was entfernt wird, ohne etwas zu tun.
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run

# Vollständige Deinstallation (interaktive Bestätigung).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh

# Ohne Bestätigung (Kalttests, mehrfache Neuinstallationen).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --confirme && bash
 ↪  ~/git/Diwall/Diwall/scripts/install.sh
```

Was wird entfernt:

Item	Detail
`/opt/diwall/`	Code, Python venv, Konfiguration
`/var/log/diwall/`	Operationsprotokolle
`diwall` system user	Erstellt ausschließlich für Diwall
`diwall` system group	Gleiches gilt
Gruppenmitgliedschaft	Ihr Konto wird aus der Gruppe `diwall` entfernt.
git pre-push hook	`core.hooksPath` deaktiviert im Quellrepository

Was niemals verändert wird: - `~/Vaults/` – Ihre Zugangsdaten - `~/git/Diwall/` – Git-Quellen - Der Browser-Cache von Playwright (**`~/.cache/ms-playwright/`**)

Beweise erfassen (`/var/log/diwall/preuves/`): Wenn das Verzeichnis Unterverzeichnisse enthält, werden diese standardmäßig mit einer Warnung beibehalten. Um sie zu entfernen:

```
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --confirme --purge-preuves
```

Einsicht in die Betriebshistorie

```
# Alle Operationen an einem Ziel.
```

```
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local

# Nur mutierende Operationen (Klicks, Formulareingaben).
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local --mutatif

# Von einem Datum
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local \
  --depuis 2026-06-01
```

Diwall – Betriebshandbuch

Version 1.24.5 – September 2026

Ebenfalls auf Französisch, Deutsch und Spanisch unter `docs/fr/`, `docs/de/` *und* `docs/es/`.

Dieses Dokument beantwortet eine Frage: **wie man X mit Diwall macht**.

Wenn Sie ein Benutzer sind – keine Befehle erforderlich. Sagen Sie Ihrem Modell, welche Website, Webanwendung oder Verwaltungsinterface Sie besuchen, beobachten oder nutzen möchten. Das Modell liest diese Anleitung und übersetzt Ihre Absicht in die richtigen Aktionen.

Wenn Sie ein Sprachmodell sind – dies sind Ihre Befehle. Führen Sie sie direkt aus.

Keine architektonischen Beschreibungen. Befehle, die funktionieren.

Inhaltsverzeichnis

1. Installation überprüfen
2. Eine Seite erfassen
3. Respektvolle Navigation (v1.15.0)
4. Verschlüsseltes Verzeichnis und Zugangsdaten
5. Ein RPA-Szenario erstellen und ausführen
6. Aktionen – vollständige Referenz
7. Häufige Probleme beheben
8. Visuelle Überwachung — watch.py
9. Betriebsprotokoll
10. CLI-Flags – Referenz
11. Exit-Codes und Ausgabe

---

1. Installation überprüfen

```
# Günstigster möglicher Test – ohne Playwright, ohne URL, sofortiger Exit mit Code 0
↪  (v1.18.0+).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --version
# → {"outil": "shot.py", "version": "1.23.0"}

# Vollständiger Test mit einem Befehl (~3 Sekunden).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://example.com --mode fast --guide-version 1.3
```

Erwartetes Ergebnis: JSON auf stdout mit `"succes": true`.

`--guide-version` (v1.18.0+): `shot.py`, `rpa.py`, und `watch.py` weigern sich, ohne dies zu funktionieren — es sei denn, ein lokaler Marker von einem vorherigen erfolgreichen Aufruf existiert bereits (`~/.config/diwall/guide_state.json`). Der Wert ist der `<!-- notice-version: X.Y -->` in Zeile 3 von `docs/GUIDE_LLM.md` — nicht die Diwall-Versionsnummer. Lesen Sie den aktuellen Wert anstatt

sich auf einen hier angegebenen Wert zu verlassen: `grep notice-version /opt/diwall/docs/GUIDE_LLM.md`. Sehen Sie sich den Abschnitt "Obligatorische Vorabprüfung" in `docs/GUIDE_LLM.md` für den vollständigen Mechanismus und das Fehlerformat an, wenn Sie diesen Schritt überspringen.

Sobald der Marker existiert, wird `--guide-version` wieder optional. Alle anderen Befehlsbeispiele in diesem Handbuch lassen ihn absichtlich weg, da ein Marker von jedem vorherigen erfolgreichen Aufruf sie bereits abdeckt, solange sich `docs/GUIDE_LLM.md`'s `notice-version` seitdem nicht geändert hat.

```
# Überprüfen Sie die installierte Version.
grep "__version__" /opt/diwall/shot.py
# → __version__ = "1.23.0"

# Überprüfen Sie, ob `playwright-stealth` verfügbar ist (Version v1.15.0).
/opt/diwall/venv/bin/python3 -c "import playwright_stealth; print('stealth OK')"

# Überprüfen Sie, ob das verschlüsselte Verzeichnis gemountet ist.
ls ~/Vaults/__PROJET__/Diwall/
# → müssen `.json`-Dateien anzeigen, keine leere Liste.
```

Wenn `ls ~/Vaults/...` eine leere Liste oder einen Fehler zurückgibt: → mounten Sie es: `bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh`

1a. Installation aus einem Paket – der einfache Weg

Die Pakete sind Release-Assets auf GitHub. Dies ist der empfohlene Kanal, es sei denn, Sie möchten den eigenen Code von Diwall ändern, in diesem Fall siehe 1b. Die beiden Kanäle sind auf einer einzelnen Maschine gegenseitig ausschließend – beide zielen auf `/opt/diwall/`.

```
sudo apt install ./diwall_1.24.5-1_all.deb                          # Debian 13, Ubuntu
↪  24.04, Ubuntu 26.04
sudo dnf install ./diwall-1.24.5-1.fc44.noarch.rpm                  # Fedora 44
sudo zypper install --allow-unsigned-rpm ./diwall-1.24.5-1.leap160.noarch.rpm   # openSUSE
↪  Leap 16.0
sudo pacman -U ./diwall-1.24.5-1-any.pkg.tar.zst                    # Arch Linux
diwall-shot --version
man diwall
```

Jedes Paket wurde in einer sauberen Container-Umgebung seines Systems validiert: Installation, ein echter Chromium-Start durch `diwall-shot`, Entfernung. Weder die Anzeige noch die FUSE-Mount-Funktion des verschlüsselten Verzeichnisses können in einem Container nachgewiesen werden. Debian 12 und Ubuntu 22.04 werden nicht unterstützt; Linux Mint 22 basiert auf Ubuntu 24.04, wurde aber nicht in dieser Konfiguration getestet. Playwright unterstützt offiziell nur Debian und Ubuntu: Auf Fedora, openSUSE und Arch läuft Chromium, weil das Paket die benötigten Bibliotheken deklariert.

Die Installation erfordert Netzwerkzugriff (die Installation von Abhängigkeiten und der Download von Chromium erfolgen im Installationsschritt, als Root, in `/opt/diwall/.cache/ms-playwright/`). Sechs Befehle werden verfügbar, jeder ein dünner Wrapper – ohne funktionellen Unterschied zu den eigenen Aufrufen des git-clone-Kanals:

Befehl	Umfasst
`diwall-shot`	`shot.py`
`diwall-rpa`	`rpa.py`
`diwall-watch`	`watch.py`
`diwall-monter-secrets`	`scripts/monter-repertoire-chiffre.sh`
`diwall-demonter-secrets`	`scripts/demonter-repertoire-chiffre.sh`
`diwall-monitor-verifier`	`scripts/monitor-verifier.sh`

Die Konfiguration befindet sich auf einem anderen Pfad in diesem Kanal: `/etc/diwall/diwall.conf`

(nicht `/opt/diwall/diwall.conf`) – eine Vorlage wird in `/etc/diwall/diwall-sample.conf` abgelegt, wird aber niemals automatisch aktiviert:

```
sudo cp /etc/diwall/diwall-sample.conf /etc/diwall/diwall.conf
sudo nano /etc/diwall/diwall.conf
sudo usermod -aG diwall $USER
```

`apt remove diwall` behält `/var/log/diwall/` (Betriebsjournal, Nachweise), `/etc/diwall/`, die Referenzaufnahmen von `watch.py` (`/opt/diwall/references/`) und das Konto `diwall`, das sie schützt (1.24.4: früher wurde das Konto gelöscht, sodass die behaltenen Dateien einer verwaisten Gruppennummer gehörten) – `apt purge diwall` entfernt all das sowie `/opt/diwall/` vollständig. Sichern Sie `/opt/diwall/references/` vorher, wenn Sie Ihre Referenzaufnahmen behalten möchten.

Auf Fedora, openSUSE und Arch (`dnf remove`, `zypper remove`, `pacman -R`) gibt es einen einzigen Befehl zum Löschen. Er löscht, was wiederhergestellt werden kann (virtuelle Umgebung, Chromium, Python-Bytecode) und behält, was Sie erstellt haben — `/etc/diwall/diwall.conf`, die Dateien unter `/var/log/diwall/`, die Aufnahmen unter `/opt/diwall/references/` — mit dem Konto `diwall`, und gibt dann den genauen Befehl aus, der sie löscht. Wenn nichts erstellt wurde, bleibt nichts übrig.

`~/Vaults/` wird auf keinem Kanal angetastet.

Handbuchseite (v1.22.0): `man diwall` dokumentiert alle sechs Befehle auf einer einzigen Seite. Die fünf anderen Befehlsnamen (`man diwall-rpa` und so weiter) verweisen auf dieselbe Seite. Sie wird beim Bauen aus `debian/diwall.1.md` erzeugt und kann daher nicht unbemerkt veralten — für die vollständige Optionsliste eines Befehls bleibt jedoch `--help` massgeblich gegenüber der Handbuchseite.

1b. Installation aus dem Quellcode – für die Modifikation von Diwall selbst

Verwenden Sie diesen Kanal nur, wenn Sie den Code von Diwall selbst ändern wollen: er legt das Repository dorthin, wo `deploy.sh` Ihre Änderungen nach `/opt/diwall/` übertragen kann. Für den einfachen Gebrauch genügt das `.deb` oben mit einem einzigen Befehl und leistet dasselbe.

```
# 1. Erstelle einen Systembenutzer und ein Verzeichnis.
sudo useradd --system --no-create-home --shell /bin/false diwall
sudo mkdir -p /opt/diwall
sudo chown root:diwall /opt/diwall

# 2. Klonen Sie das Repository.
git clone https://github.com/ronandavalan/diwall.git ~/git/Diwall/Diwall
cd ~/git/Diwall/Diwall

# 3. Erstelle eine Python-virtuelle Umgebung.
sudo /usr/bin/python3 -m venv /opt/diwall/venv
sudo /opt/diwall/venv/bin/pip install -r requirements.txt

# 4. Chromium installieren.
sudo /opt/diwall/venv/bin/playwright install chromium

# 5. Bereitstellen.
bash ~/git/Diwall/Diwall/scripts/deploy.sh

# 6. Erstellen Sie Ihr verschlüsseltes Verzeichnis für Zugangsdaten.
mkdir -p ~/Vaults/<your-project>/Diwall
# Erstellen Sie die Datei `~/Vaults/<ihr-projekt>/Diwall/<hostname>.json` mit Ihren
 ↪ Zugangsdaten.
```

Ist Diwall bereits über ein Paket installiert, verweigern `install.sh` und `deploy.sh` die Ausführung (v1.24.2 für das `.deb`, v1.24.4 für die Pakete von Fedora, openSUSE und Arch): Sie würden Dateien überschreiben, die der Paketmanager verwaltet, und das nächste Upgrade oder Entfernen des Pakets würde sie stillschweigend rückgängig machen. Um den Kanal zu wechseln, entfernen Sie zuerst das Paket; die Skripte

geben den genauen Befehl aus (`sudo apt purge diwall`, `sudo dnf remove diwall`, `sudo zypper remove diwall`, `sudo pacman -R diwall`).

Auf diesem Kanal liegt die Konfiguration in `/opt/diwall/diwall.conf`, nicht in `/etc/diwall/diwall.conf`. Deinstallieren Sie zuerst mit `bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run`, dann ohne das Flag. Auch `uninstall.sh` verweigert die Ausführung, wenn ein Paket installiert ist (v1.24.3 für das `.deb`, v1.24.4 für die anderen): Es würde `/opt/diwall/` und das vom Paketmanager verwaltete Konto `diwall` entfernen. Es gibt stattdessen den zu verwendenden Befehl zum Entfernen aus.

---

2. Eine Seite erfassen

2a. Schnelle Aufnahme – nur Text, ohne PNG (~2 Sekunden)

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --mode fast
```

Gibt Folgendes zurück: `a11y_tree` (Textstruktur der Seite), `boussole` (effektive URL, Titel). Verwenden Sie dies, wenn Sie den Titel lesen, die URL überprüfen oder Text extrahieren möchten, ohne ein PNG zu erfassen.

2b. Vollständige visuelle Erfassung mit nummerierten Elementen

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --som --a11y
```

Gibt zurück: - `capture`: Pfad zur PNG-Datei der Seite - `capture_som`: PNG mit Nummern auf anklickbaren Elementen (SoM) - `elements_som`: JSON-Liste der Elemente (id, tag, text) - `a11y_tree`: Barrierefreiheitsbaum

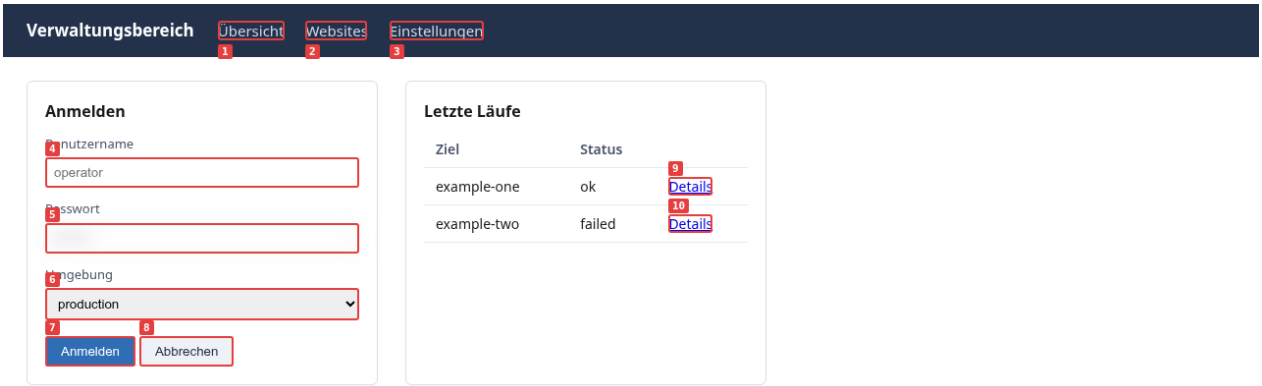


Abbildung 3: Set-of-Mark-Überlagerung: jedes interaktive Element umrandet und nummeriert

Was `--som` erzeugt. Die Zahlen im Bild sind die `id` Werte in `elements_som`, sodass das Anklicken zu `{"type": "cliquer_som", "id": 7}` wird – es gibt keine Auswahlmöglichkeiten, bei denen man raten müsste. Generiert aus einer Version eines Fixtures, die in diesem Repository gespeichert ist (`scenarios/interoperabilite/fixture/`); dieselbe Grafik existiert auch auf Französisch, Deutsch und Spanisch neben dieser.

2c. Lesen Sie zuerst die Gebrauchsanweisung

Jede Ausgabe enthält ein `boussole`-Objekt — lesen Sie es vor allem anderen:

```
"boussole": {
  "url_courante": "https://target.local/dashboard",
  "titre_page": "Dashboard — My App",
  "auth_status": "active",
  "stealth_actif": true,
  "respect": {
    "pages_visitees": 0,
    "actions_executees": 3,
    "duree_totale_ms": 2140
  }
}
```

Wenn `boussole.url_courante` nicht Ihrer Erwartung entspricht: anhalten und prüfen, bevor Sie irgendeine verändernde Aktion ausführen.

`boussole.secrets_dir_effectif` und `boussole.secrets_dir_source` (v1.23.1) geben Auskunft darüber, welches Verzeichnis mit verschlüsselten Geheimnissen tatsächlich für diesen Durchlauf verwendet wurde und woher es stammt (`env:DIWALL_SECRETS_DIR`, `env:DIWALL_CONF`, `conf:<path>`, oder `non_configure`). Eine Maschine mit mehreren Projekten kann stillschweigend zur maschineweiten Einstellung `diwall.conf` zurückgreifen, wenn ein Aufrufer vergisst, `DIWALL_SECRETS_DIR` zu exportieren. Diese beiden Felder machen dies bei jedem Durchlauf sichtbar, anstatt erst danach.

2d. Lesen Sie `etat` für eine Ja/Nein-Entscheidung (Version 1.16.0)

Jede erfolgreiche Ausführung enthält ein `etat` Objekt im JSON-Root – lesen Sie es, bevor Sie eine verändernde Aktion durchführen, anstatt manuell `auth_status`, `respect.plafond_atteint`, `erreurs_js`, und `erreurs_console` selbst zu überprüfen:

```
"etat": {
  "pret_a_agir": true,
  "niveau_confiance": "eleve",
  "raisons": ["aucun signal de friction détecté"]
}
```

Wenn `pret_a_agir` gleich `false` ist: Lesen Sie `raisons` zur Ursache (inaktive Authentifizierung, Session-Abdrift, Erreichen der Navigationsgrenze oder ein erkannten WAF-Block), bevor Sie fortfahren.

`etat` überprüft nicht, ob die URL oder der Seiteninhalt Ihren Geschäftserwartungen entspricht – verwenden Sie dazu `evaluer` mit `attendu`/`contient`/`motif` (Abschnitt 5d).

2e. `mode_conseille` — Hinweise zur Vorflugkonfiguration (Version 1.18.0)

Wenn Diwall über reale Vorabdaten zum Host verfügt, den Sie anrufen – von einer früheren `diagnostic_dom.json` Ausführung gegen diesen Host –, dann gibt `etat` eine Empfehlung für Ihren **nächsten** Anruf ab, die jedoch niemals automatisch angewendet wird:

```
"etat": {
  "pret_a_agir": true,
  "niveau_confiance": "eleve",
  "raisons": ["mode_conseille disponible : full recommandé (React détecté sur ce host)"],
  "mode_conseille": {
    "mode": "full",
    "shadow_dom": true,
    "som_rafraichir": false,
    "raisons": ["react_detecte", "shadow_roots:3"]
  }
}
```

Um diese Daten für einen bestimmten Host zu erhalten, führen Sie die Diagnose einmal aus:

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/diagnostic_dom.json \
```

```
  --url https://target.local/ --mode fast
```

Keine vorherige Diagnose für diesen Host ⟶ `mode_conseille` ist nicht vorhanden, es gibt keine Vermutungen. Vollständige Details in `GUIDE_LLM_MONITORING.md`.

Das Feld `som_rafraichir` wird zur Gewährleistung der Stabilität der Ausgabegröße beibehalten, ist aber seit Version v1.24.0 veraltet. (Die hybride SoM-Auflösung mit Fallback ist standardmäßig aktiviert – es gibt nichts, was aktiviert werden muss; siehe Abschnitt 7j).

---

3. Respektvolles Navigieren (v1.15.0)

3a. Stealth-Modus `--stealth`

Einige Websites blockieren Browser ohne grafische Oberfläche auf `navigator.webdriver=true` ohne die Absicht zu prüfen. `--stealth` entfernt diese automatische technische Kennzeichnung.

```
# direkt shot.py
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --som --stealth

# Über rpa.py
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/my-scenario.json \
  --stealth
```

Wenn aktiv: `boussole.stealth_actif = true` in der JSON-Ausgabe.

Was `--stealth` verändert: `navigator.webdriver` wurde entfernt, Plugins/languages/platform wurden normalisiert. **Was `--stealth` nicht verändert**: die IP-Adresse, Identität oder Navigationsabsicht des Nutzers.

3b. Höflichkeitsbedingte Verzögerungen

Konfiguriert in `/opt/diwall/diwall.conf`:

```
{
  "secrets_dir": "~/Vaults/__PROJET__/Diwall",
  "navigation": {
    "min_action_delay_ms": 800,
    "max_pages_par_run": 10,
    "max_actions_par_run": 30
  }
}
```

`min_action_delay_ms`: minimale Verzögerung (ms) zwischen jeder Aktion. Versandbereit. Standardwert: 800 ms.

Lokale Entwicklung – setzen Sie sie auf `0` (v1.19.0): die Standardeinstellung von 800 ms schützt einen unaufmerksamen Benutzer bei seiner *ersten, nicht konfigurierten* Ausführung vor dem öffentlichen Internet – sie hat keinen Schutzzweck für Ihre eigene Entwicklungs-/Produktionsmaschine, wo nichts dazu aufgefordert wird, sich bestimmtes Verhalten aufzuerpringen. Legen Sie den Schlüssel explizit in Ihrer lokalen Datei `diwall.conf` fest:

```
{
  "navigation": {
    "min_action_delay_ms": 0
  }
}
```

Behalten Sie die Standardeinstellung von 800 ms (oder erhöhen Sie sie) für alle Ziele, die über das öffentliche Internet erreichbar sind. Der Wert ist immer eine bewusste Wahl, die mit dem Ziel verknüpft ist und keine feste Eigenschaft des Tools – siehe die WAF- und Stealth-Anleitungen in `docs/GUIDE_LLM.md` für dasselbe Prinzip, angewendet auf blockierendes Verhalten.

Die `max_pages_par_run` und `max_actions_par_run` Grenzwerte stoppen den Ablauf sauber, falls sie überschritten werden. Es gibt keine Ausnahme – die Ausgabe im JSON-Format enthält:

```
"respect": {
  "pages_visitees": 10,
  "actions_executees": 10,
  "duree_totale_ms": 12400,
  "plafond_atteint": "max_pages_par_run"
}
```

3c. Kennzahlen zur Bewertung der Auswirkungen

Jeder Lauf gibt `respect` zurück (in der JSON-Wurzel und in boussole):

Schlüssel	Bedeutung
`pages_visitees`	Anzahl der ausgeführten `type: naviguer`-Navigationen
`actions_executees`	Gesamtzahl der ausgeführten Szenario-Aktionen
`duree_totale_ms`	Gesamtdauer des Laufs
`plafond_atteint`	`"max_pages_par_run"` oder `"max_actions_par_run"` bei vorzeitigem Abbruch

3d. Stealth-Benchmark – quantitativ (Version 1.17.1)

Zählen Sie konkrete Fingerabdruck-Signale, statt Bildschirmaufnahmen mit blossem Auge zu vergleichen — mit dieser Methode wurde die Korrektur der API-Kompatibilität von `playwright-stealth` in v1.17.0 überprüft (`docs/RETOUR_EXPERIENCE.md` FR-79):

```
# Ohne Heimlichkeit.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://bot.sannysoft.com --no-capture --timeout 20000 \
  --actions '[{"type":"evaluer","script":"navigator.webdriver"},

 ↪  {"type":"evaluer","script":"document.querySelectorAll(\"td.failed\").length"},

 ↪  {"type":"evaluer","script":"document.querySelectorAll(\"td.passed\").length"}]'

# Mit Heimlichkeit.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://bot.sannysoft.com --no-capture --stealth --timeout 20000 \
  --actions '[{"type":"evaluer","script":"navigator.webdriver"},

 ↪  {"type":"evaluer","script":"document.querySelectorAll(\"td.failed\").length"},

 ↪  {"type":"evaluer","script":"document.querySelectorAll(\"td.passed\").length"}]'
```

Lesen Sie die drei Werte in `evaluations[].valeur`: `navigator.webdriver` sollte von `true` auf `false` wechseln, `td.failed` sollte gegen `0` sinken. Referenzmessung (Korrektur v1.17.0, Sitzung 47): 12 failed → 0 failed. Eine von `evaluer` zurückgegebene Zahl oder ein boolescher Wert behält in der Ausgabe und im Journal seinen JSON-Typ (v1.24.3); zuvor lieferte `shot.py` ihn als Text (`"false"`, `"0"`). Nur Text wird auf Geheimnisformen gefiltert.

Für eine qualifizierte Zweitmeinung erzeugt das beschriebene Szenario weiterhin Screenshots zur Inspektion:

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/test_stealth.json \
  --output-dir /tmp/diwall/stealth_with --stealth
```

`capture_sannysoft_*.png` und `capture_intoli_*.png` landen in diesem Verzeichnis. Hinweis: Beide Zielseiten diskutieren in ihren Inhalten die Erkennung von Bots, was `respect.waf_bloquants` als falsch-positive Meldung auslösen kann (Abschnitt 3e) – was bei diesem spezifischen Benchmark zu erwarten ist und kein Zeichen für eine tatsächliche Sperrung darstellt.

3e. Erkennungssignal für Web Application Firewall (WAF) (Version 1.16.0, verfeinert in Version 1.17.2)

Diwall kennzeichnet einen wahrscheinlichen WAF-Block passiv – HTTP 403/429 oder eine Übereinstimmung von Titeln/HTML-Keywords (`Cloudflare`, `CAPTCHA`, `checking your browser`, usw.). Dies ist ein Signal, niemals eine Ausnahme – der Vorgang wird normal abgeschlossen:

```
"respect": {
  "waf_bloquants": 1
}
```

Wenn vorhanden und `> 0`: `etat.niveau_confiance` ist `"faible"` und `etat.pret_a_agir` ist `false`. Entscheiden Sie selbst, ob Sie es mit `--stealth` erneut versuchen, das Ziel ändern oder stoppen möchten — Diwall bricht den Vorgang nicht für Sie ab.

Seit v1.17.2 greifen generische Herstellernamen (`Cloudflare`, `Akamai`) nur noch im Seitentitel — der Abgleich mit dem vollständigen HTML erzeugte zuvor Fehlalarme bei gewöhnlichen Verweisen auf CDN-Ressourcen. Hält ein Fehlalarm an, senkt `--ignorer-waf` den `niveau_confiance`, ohne `pret_a_agir: false` zu erzwingen (`boussole.waf_ignore_actif: true` hält die Übersteuerung fest). Die Erkennung arbeitet mit Schlüsselwörtern und kann Fehlalarme erzeugen: eine Seite, die einen dieser Begriffe legitim erwähnt, wird markiert.

3f. Angegebene Sprache (Version 1.24.1)

Der Browser gibt die Sprache des Operators an. Sie wird aus der Umgebung des Prozesses entnommen, in der Reihenfolge, in der Chromium selbst vorgeht: `LANGUAGE`, dann `LC_ALL`, `LC_MESSAGES`, `LANG` – und `en-US`, wenn keiner von diesen einen brauchbaren Wert liefert (`C`, `POSIX`). Das gleiche Tag wird im `Accept-Language` Header gesendet und von `navigator.language` zurückgegeben, sodass eine Website, die ihre Sprache aushandelt, die Sprache des Operators verwendet.

Um für einen einzelnen Durchlauf eine andere Sprache zu aktivieren, setzen Sie diese in der Umgebung:

```
LANGUAGE=en diwall-shot --url https://example.com --mode fast --guide-version 1.3
```

Vor der Version v1.24.1 wurde überhaupt kein `Accept-Language` Header gesendet, während `navigator.language` die Sprache des Geräts enthielt. Eine Website, die ihre Sprache verhandelt, verwendete ihre Standardeinstellung. Ein Szenario, das einen Text auf einer solchen Website überprüft (`evaluer` mit `contient`, eine Wartezeit auf einem Label), kann daher nach einem Upgrade ein anderes Ergebnis liefern.

---

4. Verschlüsseltes Verzeichnis und Zugangsdaten

4a. Struktur

Die Zugangsdaten befinden sich in einem verschlüsselten Verzeichnis – einem gocryptfs-Volume –, das eine `.json` Datei pro Domain enthält.

```
~/Vaults/__PROJET__/Diwall/
  ├── app.example.com.json          ← credentials for https://app.example.com/
  ├── admin.example.com.json        ← credentials for https://admin.example.com/
  └── operations.jsonl              ← operation log (v1.15.0)
```

Dateiformat für Anmeldeinformationen:

```
{
  "username": "admin@example.com",
  "password": "my-password"
}
```

Der Dateiname ist = `urlparse(url).hostname`. Für `https://app.example.com/login/`, erstellen Sie `app.example.com.json`.

4b. Ein Formular ausfüllen – die unumstößliche Regel

VERBOTEN – zeigt das Passwort im Shell-Fenster und `/proc`:

```
PASS=$(jq -r '.password' ~/Vaults/.../file.json)   # NEVER
curl -d "password=$PASS" https://...                 # NEVER
```

KORREKT – Zugangsdaten werden innerhalb von Playwright aufgelöst:

```
{"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "username"},
{"type": "remplir_som", "id": 3, "valeur": "depuis_secrets", "secret_cle": "password"}
```

Werte gelangen niemals über die Shell, die Bash-Historie, Prozessprotokolle oder irgendeine Datei.

4c. Auswahl der Zugangsdatendatei für einen Durchlauf

```
# Standard-Zugangsdaten-Verzeichnis (definiert in diwall.conf > secrets_dir).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url https://target.local/ --som

# Datei mit expliziten Anmeldeinformationen (--secrets)
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som \
  --secrets /path/to/mounted/directory/creds.json

# Projektbezogenes Zugangsdaten-Verzeichnis über .diwall.conf
export DIWALL_CONF=~/git/MyProject/.diwall.conf
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url https://target.local/ --som
```

Inhalt der Datei `--secrets` — `origines_autorisees` obligatorisch seit dem 05/08/2026 (Breaking Change, keine Übergangsfrist): Eine Datei ohne diesen Schlüssel wird vor jedem Lesevorgang verweigert.

```
{"username": "operator", "password": "secret", "origines_autorisees": ["target.local"]}
```

`origines_autorisees` listet die Hostnamen auf, gegen die diese Datei verwendet werden darf – im gleichen Kleinbuchstabenformat ohne Schema und Port wie `domaine_depuis_url()`. Ein Zugriff auf eine Seite, deren Domain nicht in der Liste enthalten ist, wird verweigert (`SecretsOrigineNonAutoriseeError`).

Inhalt von `~/git/MyProject/.diwall.conf` :

```
{"secrets_dir": "../MyProject-secrets"}
```

Der Pfad wird relativ zum Speicherort von `.diwall.conf` aufgelöst.

4d. TOTP / Multi-Faktor-Authentifizierung

```
{"type": "remplir_som", "id": 6, "valeur": "depuis_secrets_totp"}
```

Liest den Schlüssel `totp_cle` (Base32-Seed) aus der Zugangsdatendatei und generiert den aktuellen TOTP-Code.

Um den Code über ntfy zu erhalten (Workflow ohne menschliches Zutun):

```
{"type": "attendre_mfa_ntfy", "id_som": 6, "timeout": 120}
```

4e. Integritätsprüfsumme (optional, ab Version 1.15.0)

Um eine Datei mit Anmeldeinformationen vor stillem Datenverlust durch FUSE zu schützen, fügen Sie ein `checksum` Feld hinzu:

```
# Generiere den Prüfsummenwert.
/opt/diwall/venv/bin/python3 -c "
import json, hashlib
creds = json.load(open('my_credentials.json'))
fields = {k: creds[k] for k in sorted(['username','password']) if k in creds}
print('sha256:' + hashlib.sha256(json.dumps(fields, sort_keys=True).encode()).hexdigest())
"
```

Fügen Sie den zurückgegebenen Wert der Datei mit den Anmeldeinformationen hinzu:

```
{
  "username": "admin@example.com",
  "password": "my-password",
  "checksum": "sha256:a3f2c1..."
}
```

Stimmt die Prüfsumme nicht, löst `shot.py` `SecretsChecksumError` (Exit 42) mit einer eindeutigen Meldung aus. Ohne den Schlüssel `checksum`: unverändertes Verhalten (striktes Opt-in).

4f. Verschlüsseltes Verzeichnis geschlossen – was tun?

```
SecretsFermesError: Le répertoire chiffré Diwall est initialisé mais non monté.
```

```
# Befestigen Sie das verschlüsselte Verzeichnis.
bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh

# Überprüfen Sie die Montage.
ls ~/Vaults/__PROJET__/Diwall/
# → müssen JSON-Dateien anzeigen.
```

4g. HTTP-Basisauthentifizierung — `--http-credentials` (v1.21.0)

Für Ziele hinter einer HTTP Basic Auth-Authentifizierung auf Netzwerkebene (RFC 7617) – ein Reverse Proxy wie Caddy, nginx oder Traefik stellt diese Herausforderung vor dem Laden jeglicher Seite bereit, was häufig vor selbst gehosteten Admin-Oberflächen zu finden ist. Dies ist ein anderer Mechanismus als die oben beschriebene formbasierte Authentifizierung (4a-4f), die weiterhin vollständig unterstützt wird und davon nicht betroffen ist.

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://internal.example/ \
  --http-credentials --secrets ~/Vaults/__PROJET__/Diwall/internal_example.json
```

Authentifizierungsdatei – das einfache `username` / `password` Paar wird bereits für den üblichen Fall verwendet (ein einziger Satz von Authentifizierungsinformationen für das Ziel):

```
{"username": "admin", "password": "my-password"}
```

Spezielle `http_username`/`http_password` Schlüssel werden zuerst ausprobiert und sind nur erforderlich, wenn das gleiche Ziel sowohl eine Netzwerk-basierte Basic Auth-Authentifizierung als auch einen separaten Anwendungslogin hat (zwei verschiedene Zugangsdaten im selben File). In diesem Fall greift Diwall automatisch auf `username`/`password` zurück, wenn die speziellen Schlüssel fehlen.

Bestätigt in der Produktion gegen ein echtes Caddy-geschütztes Ziel: die sichere Standardeinstellung (`send: "unauthorized"` – Zugangsdaten werden nur nach einem echten 401 gesendet, niemals präventiv) löste die Herausforderung beim ersten Versuch. `boussole.http_credentials_actif: true` bestätigt einen echten Erfolg, nicht nur die Übergabe des Flags; `boussole.http_auth_requise: true` unterscheidet ein nicht gelöstes 401 deutlich von einer WAF-Blockade.

---

5. Schreiben und Ausführen eines RPA-Szenarios

5a. 3-stufiges Protokoll

Schritt 1 – Die Seite erkunden (nur lesen)

```
# Schneller Überblick
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --mode fast

# Vollständige Ansicht mit nummerierten Elementen.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som --a11y

# Web-Komponenten-Anwendung (Angular, Lit, Stencil)
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som --a11y --shadow-dom

# Verbesserter DOM-Inventar (Frameworks, Shadow Roots, stabile Datenattribute).
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/diagnostic_dom.json \
  --url https://target.local/ --mode fast
```

Was festzuhalten ist: - Die SoM-Nummern der Felder und Schaltflächen (`capture_som` lesen) - Stabile Attribute: `name`, `id`, `aria-label`, `data-testid` - Blockierende Überlagerungen (Cookie-Banner, modale Fenster) - SPA oder vollständiges HTTP-Neuladen

Schritt 2 – Schreiben Sie das Szenario.

```
{
  "nom": "login_app",
  "url": "https://app.example.com/login/",
  "intention": "Administrator login with stored credentials",
  "actions": [
    {"type": "nettoyer_overlay", "selecteur": ".cookie-banner"},
    {"type": "remplir_som", "id": 1, "valeur": "depuis_secrets", "secret_cle": "username"},
    {"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "password"},
    {"type": "cliquer_som", "id": 3},
    {"type": "attendre_selecteur_present", "selecteur": ".user-avatar"},
    {"type": "capturer", "nom": "after-login"}
  ]
}
```

Schritt 3 – Ausführen

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/login_app.json --som
```

5b. Vollständiges Szenario: Anmelden und zwischen Seiten navigieren

```
{
  "nom": "audit_pages",
  "url": "https://app.example.com/login/",
  "intention": "Visual audit after deployment",
  "actions": [
    {"type": "remplir_som", "id": 1, "valeur": "depuis_secrets", "secret_cle": "username"},
    {"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "password"},
    {"type": "cliquer_som", "id": 3},
    {"type": "attendre_selecteur_present", "selecteur": ".dashboard-main"},
    {"type": "capturer", "nom": "dashboard"},
```

```
    {"type": "naviguer", "url": "https://app.example.com/settings/"},
    {"type": "attendre_navigation"},
    {"type": "capturer", "nom": "settings"},
    {"type": "naviguer", "url": "https://app.example.com/users/"},
    {"type": "attendre_navigation"},
    {"type": "capturer", "nom": "users"}
  ]
}
```

5c. Daten aus dem DOM extrahieren

```
{
  "nom": "extract_counters",
  "url": "https://app.example.com/dashboard/",
  "actions": [
    {"type": "evaluer", "script": "document.title"},
    {"type": "evaluer", "script": "document.querySelectorAll('.user-row').length"},
    {"type": "evaluer", "script": "window.location.href"}
  ]
}
```

Ergebnis in `evaluations[]` :

```
"evaluations": [
  {"index": 0, "script": "document.title", "valeur": "Dashboard — My App"},
  {"index": 1, "script": "...", "valeur": 42},
  {"index": 2, "script": "...", "valeur": "https://app.example.com/dashboard/"}
]
```

5d. Aussagen zu evaluer (rpa.py nur)

Drei sich gegenseitig ausschließende Schlüssel – jeweils einer pro Aktion:

```
{"type": "evaluer", "script": "document.querySelectorAll('.row').length", "attendu": 3}
{"type": "evaluer", "script": "document.title", "contient": "Dashboard"}
{"type": "evaluer", "script": "window.location.href", "motif": "/dashboard$"}
```

Key	Vergleich	Gültige Typen
`attendu`	strikte Gleichheit `==`	str, int, bool
`contient`	Teilstring `in`	nur str
`motif`	`re.search()` Python	nur str

Schlägt die Zusicherung fehl: rpa.py hält sofort an (Exit 1), vor jeder weiteren verändernden Aktion.

5e. Unter-Szenarien (declencher_scenario)

Definieren Sie eine Anmeldung als wiederverwendbares Teil-Szenario:

```
{
  "nom": "login_app",
  "url": "https://app.example.com/login/",
  "actions": [
    {"type": "remplir_som", "id": 1, "valeur": "depuis_secrets", "secret_cle": "username"},
    {"type": "remplir_som", "id": 2, "valeur": "depuis_secrets", "secret_cle": "password"},
    {"type": "cliquer_som", "id": 3},
    {"type": "attendre_selecteur_present", "selecteur": ".user-avatar"}
  ]
}
```

Rufen Sie dieses Unter-Szenario von einem anderen Szenario aus auf:

```
{
  "nom": "full_audit",
  "url": "https://app.example.com/login/",
  "actions": [
    {"type": "declencher_scenario", "scenario": "login_app"},
    {"type": "naviguer", "url": "https://app.example.com/report/"},
    {"type": "capturer", "nom": "report"}
  ]
}
```

Maximale Verschachtelungstiefe: 5 Ebenen.

5f. Überprüfen Sie, ob Sie sich auf der richtigen Seite befinden, bevor Sie Änderungen vornehmen

Fügen Sie immer eine Sicherheitsmaßnahme als erste Aktion in Szenarien hinzu, die Daten löschen oder ändern:

```
{"type": "evaluer", "script": "window.location.href", "contient": "/dashboard"},
{"type": "evaluer", "script": "document.querySelector('.alert-danger')?.textContent ??
 ↪  null", "attendu": null}
```

Wenn die Sicherheitsprüfung fehlschlägt: rpa.py stoppt, bevor die Löschung ausgeführt wird.

5g. Eine Sitzung fortsetzen (mit persistenten Cookies)

```
# Erster Aufruf – Authentifizierung und Speicherung der Sitzung.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://app.example.com/login/ \
  --actions /tmp/login.json \
  --sauver-session /tmp/diwall/session.json \
  --som

# Nachfolgende Aufrufe – Wiederverwendung der Sitzung (kein erneutes Anmelden).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://app.example.com/dashboard/ \
  --reprendre-session /tmp/diwall/session.json \
  --som
```

Signal für Session-Ablauf: Wenn die Sitzung abgelaufen ist, steht dort `boussole.session_derive: true` im JSON. In diesem Fall: Starte den vollständigen Login neu, ohne `--reprendre-session`.

5h. Strukturelle Nicht-Regression ohne Pixel – `--replay-verifier` (v1.17.0)

```
# Erster Durchlauf – Speichern Sie die strukturelle Referenz.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/dashboard.json \
  --sauver-verifier-reference /tmp/dashboard.ref.json

# Nachfolgende Durchläufe – vergleichen.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/dashboard.json \
  --replay-verifier /tmp/dashboard.ref.json
```

Vergleicht die Ergebnisse von `http_status`, `dom_stats`, `evaluer` und die Anzahl der Elemente des SoM (nicht den Inhalt) mit der gespeicherten Referenz. Urteil in stderr:

```
{"type_comparaison": "replay_verifier", "verdict": "stable", "diffs": []}
```

Exit 1 bei `verdict: "regression"`, wobei `diffs` jedes abweichende Feld auflistet (`reference` gegen `obtenu`). Die beiden Optionen schliessen sich gegenseitig aus.

Eine vor v1.24.3 aufgezeichnete Referenz enthält eine von `evaluer` zurückgegebene Zahl oder einen booleschen Wert als Text; mit v1.24.3 oder neuer wiederholt, wird sie als Regression gemeldet (`"2"` gegenüber

2). Zeichnen Sie sie mit `--sauver-verifier-reference` neu auf.

5i. Ein langes Szenario nach einem Fehler fortsetzen — `--checkpoint` (v1.17.0)

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/long_audit.json \
  --checkpoint /tmp/long_audit.checkpoint.json
```

Schlägt das Szenario unterwegs fehl, wird `/tmp/long_audit.checkpoint.json` geschrieben, mit der Anzahl der abgeschlossenen Aktionen und einer Sitzungsdatei. **Starten Sie genau denselben Befehl erneut**, um fortzusetzen: bereits abgeschlossene Aktionen werden übersprungen. Bei vollem Erfolg wird die Checkpoint-Datei automatisch gelöscht.

Ein abgebrochener Durchlauf aufgrund einer Navigationsbeschränkung (`max_actions_par_run`/`max_pages_par_run`) wird seit Version v1.17.2 auf die gleiche Weise behandelt wie ein teilweiser Fehler – der Checkpoint wird mit dem tatsächlichen Fortschritt aktualisiert und nicht gelöscht. Vor Version v1.17.2 wurde er in diesem Fall ebenfalls gelöscht (er gibt das gleiche `succes: true` Signal wie ein vollständig abgeschlossener Abschnitt) und verliert dabei stillschweigend alle verbleibenden Fortschritte bei langen Szenarien.

Der DOM-Zustand (geöffnete Dialogfenster, halb ausgefüllte Formulare) wird nie über ein Neustart gespeichert – nur Cookies/`localStorage` und die Position in der Aktionsliste werden erhalten. Verlassen Sie sich nicht auf `--checkpoint`, um eine mehrstufige Formularausfüllung fortzusetzen; es setzt die Ausführung nur an den Grenzen von Aktionen fort.

5j. Elemente innerhalb eines Iframes ansteuern (Version 1.17.0)

Es findet keine Nummerierung mit ”Set-of-Mark” innerhalb eines `<iframe>` statt (weder bei derselben Herkunft als auch bei unterschiedlicher Herkunft) – greifen Sie stattdessen direkt über einen CSS-Selektor darauf zu:

```
{"type": "cliquer_iframe", "iframe_selecteur": "iframe#paiement", "selecteur":
 ↪  "button.valider"},
{"type": "remplir_iframe", "iframe_selecteur": "iframe#paiement", "selecteur":
 ↪  "input[name=cvv]", "valeur": "depuis_secrets", "secret_cle": "cvv"}
```

`remplir_iframe` unterstützt `valeur: "depuis_secrets"` genau wie `remplir`. (Abschnitt 4b) – es gibt niemals eine Klartext-Anmeldeinformation in diesem Szenario. Wenn das Ziel- Element die Interaktion verweigert (z. B. ein `contenteditable` Bereich in einem schreibgeschützten Zustand), fügen Sie `"force": true` zu `cliquer_iframe` hinzu – dieselbe Semantik wie `cliquer`. (Abschnitt 7e).

Um den internen Selektor zu finden: Verwenden Sie `evaluer` im Inhalt des Iframes, wenn er die gleiche Herkunft hat (`document.querySelector('iframe').contentDocument...`), oder konsultieren Sie die eigene Markup-/Dokumentation der Zielanwendung, wenn sie eine andere Herkunft hat.

5k. Verschachtelte Iframes — `iframe_chemin` (v1.18.0)

Ein Iframe innerhalb eines anderen Iframes: Ersetzen Sie `iframe_selecteur` durch `iframe_chemin`, ein geordnetes Array – ein CSS-Selektor pro Verschachtelungsebene, von äußerster bis innerster.

```
{"type": "cliquer_iframe", "iframe_chemin": ["iframe#wrapper", "iframe#paiement"],
 ↪  "selecteur": "button.valider"},
{"type": "remplir_iframe", "iframe_chemin": ["iframe#wrapper", "iframe#paiement"],
 ↪  "selecteur": "input[name=cvv]", "valeur": "depuis_secrets", "secret_cle": "cvv"}
```

`iframe_selecteur` (einzelner Frame) und `iframe_chemin` (verschachtelte Einbettung) sind gegenseitig ausschließend – genau eines ist pro Aktion erforderlich. Für einen iframe der obersten Ebene, verwenden Sie weiterhin `iframe_selecteur` (Abschnitt 5j).

---

6. Aktionen – vollständige Referenz

Typ	Erforderliche Parameter	Optionale Parameter	Hinweise
`naviguer`	`url`	—	Vollständiges HTTP-Neuladen. Wird in `respect.pages_visitees` gezählt
`cliquer`	`selecteur`	`force` (bool), `repli_js` (bool)	`force: true` umgeht CSS-versteckte Elemente oder zeigt ein Modal an. `repli_js: true` wiederholt den Klick über JavaScript, falls der native Klick fehlschlägt (v1.22.0) – benötigt `--no-evaluer` deaktiviert
`cliquer_som`	`id`	—	Klickt in die Mitte des Elements. Kein `force` erforderlich
`cliquer_visuel`	`description`	—	LLM-Vision (~32 s). Letzte Möglichkeit für Canvas oder Elemente ohne Attribute
`remplir`	`selecteur`, `valeur`	`secret_cle`	`valeur: "depuis_secrets"` löst die gespeicherten Zugangsdaten
`remplir_som`	`id`, `valeur`	`secret_cle`	Löscht das Feld vor dem Tippen. `valeur: "depuis_secrets_totp"` für TOTP
`capturer`	`nom`	`som` (bool)	Benanntes Zwischen-PNG. `som: true` für einen annotierten Screenshot
`evaluer`	`script`	`attendu`, `contient`, `motif`	JavaScript wird im Browser ausgeführt. Assertions nur für rpa.py
`defiler`	`px` oder `selecteur`	—	Vertikales Scrollen in Pixeln (`px`) oder Scrollen zu einem Element (`selecteur`)
`pause`	`ms`	`interval_capture`	Feste Verzögerung in ms. Bevorzugt `attendre_selecteur_present` für DOM-Signale
`attendre`	`selecteur`	`interval_capture`	Wartet, bis ein CSS-Selektor vorhanden ist
`attendre_navigation`	—	—	Wartet auf `networkidle` (Ende der Netzwerkaktivität)
`attendre_url`	`motif`	`attendre_changement` (bool)	Die URL enthält ein Muster (teilweise Übereinstimmung). `attendre_changement: true`, wenn die aktuelle URL bereits das Muster enthält

Typ	Erforderliche Parameter	Optionale Parameter	Hinweise
`attendre_selecteur_present`	`selecteur`	—	Wartet, bis ein Element sichtbar ist (Zustand=sichtbar)
`attendre_absence`	`selecteur`	`delai_initial_ms`	Wartet, bis ein Element aus dem DOM entfernt wird (Zustand=abgetrennt)
`attendre_reseau_calme`	—	`timeout_ms`	500 ms Netzwerkaktivität. `timeout_ms`: maximale Dauer, bevor abgebrochen wird
`attendre_mfa_ntfy`	`id_som`	`timeout`	Wartet auf einen TOTP-Code über ntfy und füllt ihn in das SoM-Feld ein
`nettoyer_overlay`	`selecteur`	—	Versteckt blockierende Overlays (Cookie-Banner, Modal). Vor der Verwendung von SoM verwenden
`declencher_scenario`	`scenario`	—	Fügt Aktionen eines Unter-Szenarios inline ein. Maximale Tiefe: 5
`cliquer_iframe`	`iframe_selecteur` | `iframe_chemin`, `selecteur`	`force` (bool)	Klickt innerhalb eines Iframes (v1.17.0). `iframe_chemin` für verschachtelte Iframes (v1.18.0, Abschnitt 5k). Kein SoM innerhalb von Frames
`remplir_iframe`	`iframe_selecteur` | `iframe_chemin`, `selecteur`, `valeur`	`secret_cle`	Füllt innerhalb eines Iframes aus (v1.17.0). `iframe_chemin` für verschachtelte Iframes (v1.18.0). `valeur: "depuis_secrets"` unterstützt

---

7. Umgang mit häufigen Hindernissen

7a. Cookie-Banner / Sperrbanner

```
{"type": "nettoyer_overlay", "selecteur": ".cookie-consent-banner, #gdpr-overlay"}
```

Platzieren Sie dies **vor** jeder anderen Aktion und vor dem SoM. Die Überlagerung maskiert Elemente, die mit SoM-Nummern versehen sind. Verwenden Sie dies nicht in `watch.py` Szenarien (die Überlagerung ist Teil der visuellen Referenz).

7b. Element außerhalb des sichtbaren Bereichs

SoM warnt, wenn ein interaktives Element außerhalb des Bildschirms liegt:

```
"som_hors_viewport": 3,
"avertissement_scroll": "3 interactive element(s) off-viewport — use defiler before
↪  cliquer_som"
```

```
{"type": "defiler", "selecteur": "#the-button"},
{"type": "remplir_som", "id": 7, "valeur": "depuis_secrets", "secret_cle": "username"}
```

7c. Web-Komponenten – Shadow DOM

Wenn sichtbare, interaktive Elemente keine SoM-Nummer erhalten:

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som --shadow-dom
```

Oder im Szenario: `"shadow_dom": true` am Anfang.

Wann verwenden: Angular, Lit, Stencil, FAST. Nicht in Projekten ohne Web-Komponenten aktivieren.

Um auf ein Element innerhalb eines Shadow Roots zuzugreifen, ohne `--shadow-dom` zu verwenden:

```
{"type": "evaluer", "script":
 ↪  "document.querySelector('my-component').shadowRoot.querySelector('button').click()"}
```

7d. Single-Page Applications (SPA) (React, Vue, Angular) – Navigation ohne Neuladen

Nach einem Klick, der die Ansicht in einer Single-Page-Anwendung (SPA) verändert, weiß Playwright nicht, wann die Navigation abgeschlossen ist.

```
{"type": "cliquer_som", "id": 5},
{"type": "attendre_url", "motif": "/dashboard"},
{"type": "evaluer", "script": "document.title", "contient": "Dashboard"}
```

Oder warten Sie auf ein Element, das spezifisch für die neue Ansicht ist:

```
{"type": "cliquer_som", "id": 5},
{"type": "attendre_selecteur_present", "selecteur": "[data-testid='dashboard-main']"}
```

Verwenden Sie niemals eine Klick-Aktion als alleiniges Indiz dafür, dass die Navigation abgeschlossen ist, ohne ein entsprechendes Signal vom Document Object Model (DOM).

7e. CSS-Dialog oder showModal()

`TimeoutError` auf `cliquer` wenn das Element im DOM sichtbar ist = CSS-verstecktes Element oder innerhalb eines Dialogfensters.

```
{"type": "cliquer", "selecteur": "#dialog-confirm button[type=submit]", "force": true}
```

Wenn `force: true` unzureichend ist (Element fehlt im DOM):

```
{"type": "evaluer", "script": "document.querySelector('#dialog-confirm
 ↪  button[type=submit]').click()"}
```

Verwenden Sie nicht "`force`" auf "`cliquer_som`". Das ist unnötig, da "`cliquer_som`" Koordinaten verwendet und native Prüfungen umgeht.

7f. Lange Operation (Spinner, Batch-Job)

Verwenden Sie `pause` nicht, um eine feste Dauer abzuwarten. Warten Sie auf das DOM-Signal:

```
{"type": "cliquer_som", "id": 7},
{"type": "attendre_absence", "selecteur": ".spinner", "delai_initial_ms": 500},
{"type": "attendre_selecteur_present", "selecteur": ".result-container"},
{"type": "capturer", "nom": "result"}
```

Wenn die Operation kein DOM-Signal liefert, verwenden Sie `interval_capture` zur Beobachtung des Zustands:

```
{"type": "pause", "ms": 30000, "interval_capture": 5}
```

Zwischenaufnahmen erscheinen in `stream_captures[]`.

7g. Kapazitätsgrenze erreicht (Version 1.15.0)

Ist `respect.plafond_atteint` in der Ausgabe vorhanden, wurde der Lauf vor dem Ende des Szenarios angehalten. Die verbleibenden Aktionen wurden nicht ausgeführt.

Erhöhen Sie `max_pages_par_run` oder `max_actions_par_run` in `diwall.conf`. Teilen Sie das Szenario in mehrere Durchläufe auf. Überschreiben Sie die Grenzwerte im Szenario-JSON (dazu wird in *CADRE* dokumentiert).

7h. `<select>` Formularfeld

`remplir` funktioniert nicht auf `<select>`. Verwenden Sie `remplir_som` mit der SoM-ID von `<select>`.

7i. Ungültige SoM-IDs beim nächsten Durchlauf

SoM-IDs werden bei jeder Aufnahme neu berechnet. Sie bleiben nicht zwischen den Aufrufen erhalten. Führen Sie immer `shot.py --som` erneut aus, um die IDs der aktuellen Ausführung zu erhalten. Nach einem `defiler` oder beim Öffnen eines Modals: Führen Sie `shot.py --som` erneut aus.

7j. SoM ID-Drift bei hochdynamischen Seiten – hybride Lösung (v1.24.0)

`cliquer_som`/`remplir_som` lösen `id: N` durch erneutes Indizieren des aktiven DOM zum Zeitpunkt des Klicks – wenn ein interaktives Element erscheint oder verschwindet **vor** Ihrem Ziel in der DOM-Reihenfolge zwischen dem `--som` Capture und dem Klick (z. B. ein Cookie-Banner, das geschlossen wird, ein Modal, das geöffnet wird), kann ein rohes `id: N` stillschweigend auf ein **anderes** Element aufgelöst werden, als das Element, das im Screenshot mit der Nummer N angezeigt wird.

Seit Version v1.24.0 ist der Standard-Resolver ein Hybrid-Resolver. Bezeichnen Sie ihn auf einer Seite als:

- sucht den `data-dw-som-id="N"` Marker (**stabilen** Pfad, festgelegt durch eine `{"type":"capturer","som":true}` Aktion oder die finale Erfassung);
- berechnet das N-te Element durch direkte Neuindizierung (**roher** Pfad);
- gibt den stabilen Pfad zurück, wenn der Marker existiert, den rohen Pfad andernfalls (**Fallback** – identisch zum Verhalten vor v1.24.0);
- meldet `boussole.respect.som_resolution` (`stable` | `brut` | `brut_sans_reference`) und, wenn die beiden Pfade voneinander abweichen, `boussole.respect.som_derive_detectee` mit der SoM-ID.

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som \
  --actions '[{"type":"capturer","nom":"avant","som":true},{"type":"cliquer_som","id":5}]'
```

Der stabile Pfad hilft nur innerhalb eines Szenarios – der Marker überlebt nicht über zwei `shot.py` Aufrufe (jeder lädt die Seite neu). Für ein anfälliges Ziel für Mutationen, erfassen Sie den Start-of-Mutation (SoM) im selben Szenario, bevor der erste `cliquer_som` ausgeführt wird. Wenn `som_resolution` gleich `brut_sans_reference` ist, kann kein Drift erkannt werden – erfassen Sie den SoM erneut, wenn sich das DOM geändert hat. `--som-brut` erzwingt eine reine, unveränderte Neuindizierung (keine Markerprüfung, keine Divergenzprüfung); `boussole.som_brut_actif: true` danach. `--som-rafraichir` (v1.17.0) ist ein Alias, der aus Gründen der Abwärtskompatibilität beibehalten wird und keine Funktion ausführt.

Seit Version v1.17.2 löscht der Injector Markierungen, die von einer vorherigen `--som` Aufnahme auf derselben Seite hinterlassen wurden, bevor er die Nummerierung neu beginnt. Ohne dies könnte ein Element, das zwischen zwei Aufnahmen ausgeblendet oder aus dem sichtbaren Bereich geschoben wurde, eine veraltete `data-dw-som-id` beibehalten, was zu einer Kollision mit einem neu nummerierten Element führen könnte und dazu, dass das falsche Element ausgewählt wird.

7k. Website durch WAF blockiert (sofortige 403-Fehlermeldung)

```
# Versuche es heimlich.
```

```
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --mode fast --stealth
```

Wenn der Fehler 403 trotz `--stealth` weiterhin auftritt: Die Website verwendet TLS-Fingerprinting (JA3/JA4) oder eine erweiterte Verhaltensanalyse (Cloudflare Enterprise). `playwright-stealth` umgeht diese Schutzmaßnahmen nicht. Siehe `docs/RETOUR_EXPERIENCE.md` FR-77/FR-78/FR-79 für den Kontext.

Diwall markiert außerdem wahrscheinlich einen Block passiv, ohne dass Sie den HTTP-Status selbst überprüfen müssen – siehe Abschnitt 3e (`respect.waf_bloquants`).

7l. Die anfängliche Navigation wird nie abgeschlossen — `--wait-until` (v1.22.0)

Symptom: `TimeoutError` beim ersten Navigieren und das Auslösen von `--timeout` ändert nichts (45 Sekunden schlagen genau wie 10 Sekunden fehl). Ursache: Standardmäßig wartet Diwall auf `networkidle` – 500 ms Netzwerk-Stille. Eine Seite, die kontinuierlich abfragt (Live-Statistiken, automatisch aktualisierte Zähler, Router-Admin-Panels), erzeugt diese Stille nie, sodass kein Timeout-Wert jemals groß genug sein kann.

```
# shot.py – direkte Aufklärung
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url http://target.local/ --wait-until load --som --a11y --guide-version 1.3

# rpa.py – verbreitet zu shot.py, sodass Szenarien die gleichen Ziele erreichen.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario ./admin_login.json --wait-until load --guide-version 1.3
```

Ein Szenario kann dies stattdessen als eine Stammeigenschaft enthalten und so für sich allein stehen:

```
{"url": "http://target.local/", "wait_until": "load", "actions": [...]}
```

Die Kommandozeilenoption hat Vorrang vor der Szenarioeigenschaft.

Wert	Wartet auf	Verwendung bei
`networkidle`	500 ms Netzwerkruhe	Standard – beibehalten, es sei denn, es schlägt fehl
`load`	`load` Ereignis (Seite und Unterressourcen)	kontinuierliches Abfragen / Live-Statistiken
`domcontentloaded`	HTML geparst, Unterressourcen noch ausstehend	sehr umfangreiche Seite, das DOM ist alles, was Sie benötigen

Gilt nur für die anfängliche Navigation – die Aktion `naviguer` ist davon nicht betroffen. `boussole.wait_until` meldet den Wert nur, wenn er sich von dem Standardwert unterscheidet.

---

8. Visuelle Überwachung — watch.py

8a. Eine Referenz speichern

```
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/status \
  --sauver-reference \
  --nom home
```

Die Referenz wird in `/opt/diwall/references/` gespeichert.

8b. Vergleich mit der Referenz (Pixeldifferenz)

```
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/status \
```

```
  --comparer-pixel /opt/diwall/references/target.local_home/reference.png \
  --nom home
```

Urteile:

taux_diff	Urteil	Exit-Code
< 0,2 %	stable	0
0,2 % – 5 %	drift	0
≥ 5 %	regression	1
Unterschiedliche Dimensionen	viewport_mismatch	2

8c. Semantische Vergleichung (LLM)

```
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/status \
  --comparer \
  --llm local
```

Kombinieren Sie Pixel-Differenzanalyse und LLM-Analyse:

```
--llm-en-complement   # LLM only if pixel verdict is drift or regression
```

`--llm claude` (v1.24.2) sendet sowohl die Referenzaufnahme als auch die aktuelle Aufnahme an die Anthropic API anstelle des lokalen Modells; dies gilt sowohl für `--comparer` als auch für `--llm-en-complement`. Die Aufnahmen verlassen den Computer und werden so reduziert, dass ihre längste Seite 1568 px nicht überschreitet, was die Größe ist, mit der die API arbeitet. Es benötigt das Modul `anthropic`, das in einer Standardinstallation nicht vorhanden ist, und einen Schlüssel in der Umgebung des Prozesses:

```
sudo /opt/diwall/venv/bin/pip install anthropic
ANTHROPIC_API_KEY=... diwall-watch --url https://target.local/status --comparer --llm
 ↪  claude --guide-version 1.3
```

Ein fehlendes Modul, ein verweigerter Schlüssel oder eine Ablehnung des Modells wird als eine einfache Nachricht in der Ausgabe angezeigt (`erreur`, oder `analyse_llm` im Pixelmodus), niemals als Traceback.

8d. Eine animierte Zone ignorieren

```
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/status \
  --comparer-pixel reference.png \
  --exclure-zone 100,200,300,50    # X,Y,Width,Height in pixels
```

8e. Überwachungs-Schleife

```
while true; do
  /opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
    --url https://target.local/status \
    --comparer-pixel /opt/diwall/references/status-ok.png \
    --ntfy-url https://ntfy.sh/my-alerts
  sleep 60
done
```

8f. Cron für autonome Überwachung

```
# /etc/cron.d/diwall-monitor
*/30 * * * * diwall /opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/status \
  --comparer-pixel /opt/diwall/references/status-ok.png \
  --ntfy-url https://ntfy.sh/my-alerts \
  >> /var/log/diwall/cron.jsonl 2>&1
```

8g. Kontinuierliche strukturelle Überwachung — `monitor-verifier.sh` (v1.18.0)

Komplemente 8a–8f: `watch.py` überwacht das *Aussehen* (Pixel/Semantik). `scripts/monitor-verifier.sh` überwacht die *Struktur* (`http_status`, `dom_stats`, `evaluations`, SoM-Anzahl) — kein Bild, kein LLM-Aufruf, basiert auf `--no-capture` + `--replay-verifier` (Abschnitt 5h).

```
# Erster Durchlauf – Erstellung der strukturellen Referenz.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/sillage_login.json \
  --sauver-verifier-reference /opt/diwall/references/sillage_login.ref.json

# Ein einmaliger Check und Alarm – kein Hintergrunddienst, sondern ein Skript, das
↪  wiederholt über Cron ausgeführt wird.
# `scripts/*.sh` wird niemals auf /opt/diwall/ bereitgestellt, daher läuft es von der
↪  Git-Repository.
# Quelle, wie von einem eigenen Benutzer.
bash ~/git/Diwall/Diwall/scripts/monitor-verifier.sh \
  --scenario /opt/diwall/scenarios/sillage_login.json \
  --reference /opt/diwall/references/sillage_login.ref.json \
  --ntfy-topic diwall-monitoring
```

```
# crontab -e (Ihre eigene Crontab)
*/15 * * * * bash ~/git/Diwall/Diwall/scripts/monitor-verifier.sh \
  --scenario /opt/diwall/scenarios/sillage_login.json \
  --reference /opt/diwall/references/sillage_login.ref.json \
  --ntfy-topic diwall-monitoring \
  >> /var/log/diwall/cron-structural.jsonl 2>&1
```

Stabilität ⟶ Stille. Regression ⟶ eine `ntfy` Benachrichtigung mit den Änderungen. Jede Aufrufung ist ein isolierter Prozess – kein Hintergrunddienst, keine Gefahr von Speicherlecks, und Die Einstellungen für die Respektvolle Navigation werden bei jedem Durchlauf sauber zurückgesetzt.

---

9. Betriebsprotokoll

Das Protokoll ist in `diwall.conf` (Version 1.15.0) konfigurierbar:

```
"journal": {
  "chemin": "~/Vaults/__PROJET__/Diwall/operations.jsonl"
}
```

Wenn das Verzeichnis nicht vorhanden ist oder nicht gemountet wurde, Fallback: Umgebungsvariable `DIWALL_JOURNAL`, dann `/var/log/diwall/operations.jsonl`.

```
# Lesen Sie die letzten 10 Einträge.
tail -n 10 ~/Vaults/__PROJET__/Diwall/operations.jsonl | python3 -m json.tool

# Filtern nach Ziel (journal.py Tool).
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com

# Filtern Sie nur Operationen, die den Zustand verändern.
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com --mutatif

# Von einem Datum
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com --depuis 2026-07-01

# Fehlgeschlagene Läufe nur (Version 1.20.0) – resultat != "Erfolg"
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
```

```
  --cible app.example.com --erreurs
```

Felder in jedem Eintrag:

Feld	Bedeutung
`ts`	ISO 8601-Zeitstempel
`version`	Diwall-Version
`outil`	`shot.py` oder `rpa.py`
`cible_url`	Ziel-URL
`scenario`	Pfad zur Szenariodatei (RPA-Modus)
`source_scenario`	Nur Dateiname der Szenariodatei, ohne Pfad (v1.18.0) — steuert `mode_conseille` (Abschnitt 2e)
`resultat`	`"succes"` oder `"echec"`
`mutatif`	`true`, wenn mindestens eine Schreibaktion vorhanden ist
`duree_ms`	Dauer in ms
`intention`	Label, das über `--intention` oder das Feld ”Szenario” `intention` übergeben wird

9a. Protokolldrehung (G-36, CHANTIER_SANITISATION.md)

Diwall liefert keine Logrotate-Konfiguration — `/var/log/diwall/operations.jsonl` wächst unbegrenzt, bis der Administrator eine solche installiert. `lib/journal.py` öffnet und schließt die Datei bei jedem Schreibvorgang (kein persistenter Dateideskriptor über mehrere Ausführungen), speziell damit das **Standard**verhalten von Logrotate (die aktuelle Datei umbenennen und eine neue erstellen) korrekt funktioniert, ohne dass spezielle Optionen erforderlich sind: der nächste Schreibvorgang öffnet den Pfad erneut und findet den neuen Inode.

Fügen Sie keinen Eintrag `copytruncate` zu einer Diwall-Logrotate-Konfiguration hinzu – er ist hier unnötig (im Gegensatz zu Tools, die einen Dateideskriptor während ihrer gesamten Lebensdauer geöffnet halten) und führt ein Fenster für Datenverluste wieder ein, das dieses Design vermeiden soll. Beispiel: `/etc/logrotate.d/diwall`:

```
/var/log/diwall/operations.jsonl {
    weekly
    rotate 8
    compress
    delaycompress
    missingok
    notifempty
    create 0640 diwall diwall
}
```

`journal.py` (der Leser) verfolgt bereits transparente rotierte Dateien automatisch. (`operations.jsonl`, `.1`, `.2.gz`, ...) – kein zusätzlicher Schritt erforderlich nach der Rotation.

---

10. Befehlszeilenparameter – Referenz

shot.py

Flag	Standardwert	Beschreibung
`--version`	—	Gibt die installierte Version aus und beendet sofort – keine Playwright- oder andere Argumente erforderlich (v1.18.0)

Flag	Standardwert	Beschreibung
`--guide-version X.Y`	—	Nachweis des Lesens von `docs/GUIDE_LLM.md` – erforderlich, es sei denn, ein gültiger lokaler Marker existiert bereits (v1.18.0, Abschnitt 1)
`--url URL`	erforderlich	URL zur Erfassung
`--actions FILE`	—	JSON-Datei mit sequenziellen Aktionen
`--output-dir DIR`	`/tmp/diwall`	PNG-Ausgabeverzeichnis
`--timeout MS`	10000	Playwright-Timeout pro Aktion (ms)
`--screenshot-timeout MS`	120000	Timeout für `page.screenshot()` (ms). Unterscheidet sich von `--timeout`
`--largeur PX`	1280	Viewport-Breite
`--hauteur PX`	720	Viewport-Höhe
`--som`	off	Aktiviert Set-of-Mark (Elementnummerierung)
`--a11y`	off	Inkludiert den Accessibility-Baum in der JSON-Datei
`--shadow-dom`	off	Durchläuft Shadow Roots für SoM (Angular, Lit, Stencil)
`--stealth`	off	playwright-stealth Stealth-Modus (v1.15.0)
`--mode fast|full`	—	`fast` = `--no-capture --a11y`. `full` = Standardverhalten
`--no-capture`	off	Überspringt die PNG-Erfassung und SoM
`--llm local|claude`	`local`	LLM-Engine für `cliquer_visuel`
`--secrets FILE`	—	Expliziter Pfad zu einer Datei mit Anmeldeinformationen
`--auth-indicator SEL`	—	CSS-Selektor, der nur in einer authentifizierten Sitzung vorhanden ist
`--auth-indicator-negative SEL`	—	CSS-Selektor, der nur außerhalb einer authentifizierten Sitzung vorhanden ist
`--intention TEXT`	—	Geschäftlicher Label, der im Protokoll aufgezeichnet wird
`--sauver-session FILE`	—	Speichert Cookies nach den Aktionen
`--reprendre-session FILE`	—	Setzt eine gespeicherte Sitzung fort
`--interval-capture N`	0	Periodische Erfassungen alle N Sekunden während `attendre`, `pause`
`--som-brut`	off	Erzwingt eine reine, rohe SoM-Neuindizierung; deaktiviert den stabilen Pfad und die Divergenz-Erkennung des Hybrid-Resolvers (v1.24.0, Abschnitt 7j)
`--som-rafraichir`	off	No-op-Alias seit v1.24.0 – Hybrid-Auflösung ist die Standardeinstellung (v1.17.0, Abschnitt 7j)

Flag	Standardwert	Beschreibung
`--ignorer-waf`	off	Ein erkannten WAF-Block verschlechtert `niveau_confiance` erzwingt aber nicht mehr automatisch `pret_a_agir: false` (v1.17.2, Abschnitt 3e)
`--http-credentials`	off	Löst HTTP-Basic-Auth-Anmeldeinformationen aus der Datei mit Anmeldeinformationen auf, beschränkt auf den Ursprung des Ziels (v1.21.0, Abschnitt 4g)
`--no-evaluer`	off	Verweigert die Aktion **evaluer** für den gesamten Lauf – empfohlen in der Produktion für Ziele mit sensiblen Formularen (v1.15.1)
`--no-filtre-evaluer`	off	Deaktiviert die stdout-Neutralisierung der Rückgabewerte, URLs und Fehlermeldungen von **evaluer** – nur für explizite Debug-Läufe. Die Neutralisierung ist standardmäßig aktiviert; wenn sie deaktiviert ist, wird `boussole.filtre_evaluer_actif: false` in der Ausgabe gesetzt, sodass der Bediener sie aus der JSON-Datei selbst überprüfen kann (v1.23.0)

rpa.py

Überträgt alle relevanten shot.py Flags, sowie:

Flag	Beschreibung
`--version`	Gibt die installierte Version aus und beendet sofort (v1.18.0)
`--guide-version X.Y`	Nachweis des Lesens von `docs/GUIDE_LLM.md` – unabhängig überprüft, gleiche Regel wie shot.py (v1.18.0)
`--scenario FILE`	Pfad zu einem JSON- oder YAML-Szenario (erforderlich)
`--url URL`	Überschreibt die Szenario-URL, ohne die Datei zu ändern
`--stealth`	Wird an shot.py weitergegeben
`--mode fast|full`	Wird an shot.py weitergegeben
`--som-brut`	Wird an shot.py weitergegeben. Kann auch als Szenario-Root-Eigenschaft `"som_brut": true` festgelegt werden (v1.24.0, Abschnitt 7j)
`--som-rafraichir`	Wird an shot.py weitergegeben – seit v1.24.0 ein Alias ohne Funktion (v1.17.0, Abschnitt 7j)
`--ignorer-waf`	Wird an shot.py weitergegeben (v1.17.2, Abschnitt 3e)
`--http-credentials`	Wird an shot.py weitergegeben. Kann auch als Szenario-Root-Eigenschaft `"http_credentials": true` festgelegt werden (v1.21.0, Abschnitt 4g)
`--sauver-verifier-reference FILE`	Speichert eine strukturelle Referenz für `--replay-verifier` (v1.17.0, Abschnitt 5h)
`--replay-verifier FILE`	Vergleicht den Lauf mit einer strukturellen Referenz, beendet mit Fehlercode 1 bei Regression (v1.17.0, Abschnitt 5h)

Flag	Beschreibung
`--checkpoint FILE`	Setzt ein langes Szenario nach einem Fehler während des Laufs fort (v1.17.0, Abschnitt 5i)

watch.py

Flag	Beschreibung
`--version`	Gibt die installierte Version aus und beendet sofort (v1.18.0)
`--guide-version X.Y`	Nachweis des Lesens von `docs/GUIDE_LLM.md` – unabhängig überprüft, gleiche Regel wie shot.py (v1.18.0)
`--url URL`	URL zur Überwachung
`--sauver-reference`	Erfassen und als Referenz speichern
`--comparer-pixel REF`	Pixel-Differenz im Vergleich zur PNG-Datei REF
`--comparer`	Semantische LLM-Differenz
`--llm local|claude`	Engine für `--comparer` und `--llm-en-complement` (Standard: `local`; `claude` sendet die Erfassungen an die Anthropic API, v1.24.2)
`--nom NAME`	Name der Ansicht (mehrere Ansichten pro URL)
`--seuil-stable F`	`stable` Schwellenwert (Standard: 0.002 = 0.2%)
`--seuil-regression F`	`regression` Schwellenwert (Standard: 0.05 = 5%)
`--exclure-zone X,Y,W,H`	Zone, die ignoriert werden soll (wiederholbar)
`--heatmap`	Erzeugt ein PNG der geänderten Zonen
`--ntfy-url URL`	Sendet eine ntfy-Benachrichtigung bei Regression
`--llm-en-complement`	Fügt eine LLM-Differenz hinzu, wenn die Pixel-Differenz eine Abweichung oder Regression darstellt

---

11. Rückgabecodes und Ausgabe

Rückgabecodes

Code	Ursache	Was zu tun ist
0	Erfolg	—
1	Playwright-Fehler, fehlgeschlagene Aktion, rpa.py-Zusicherung	`erreur` im JSON lesen. Siehe `GUIDE_LLM_INTERACTIONS.md`
1	`guide_non_lu` — `--guide-version` fehlt oder ist falsch, kein gültiger Marker (v1.18.0)	Greift, bevor Playwright startet. `docs/GUIDE_LLM.md` lesen, mit `--guide-version X.Y` erneut starten (Abschnitt 1)
2	`viewport_mismatch` (watch.py)	Referenz im gleichen Viewport neu aufnehmen
3	Modul `playwright` nicht gefunden	Über `/opt/diwall/venv/bin/python3` aufrufen
42	`SecretsFermesError` — verschlüsseltes Verzeichnis nicht gemountet oder Prüfsumme ungültig	Es mounten oder die Zugangsdatendatei prüfen
43	`SecretsNonConfigureError` — `diwall.conf` fehlt	`sudo cp /opt/diwall/diwall-sample.conf /opt/diwall/diwall.conf && sudo nano /opt/diwall/diwall.conf`

Struktur des Ausgabe-JSON


```
{
  "succes": true,
  "http_status": 200,
  "url_finale": "https://target.local/dashboard",
  "erreurs_js": [],
  "erreurs_console": [],
  "duree_ms": 2400,
  "horodatage": "2026-07-01T12:00:00+02:00",
  "capture": "/tmp/diwall/a1b2c3d4e5f6/capture_1234567890123456789.png",
  "capture_som": "/tmp/diwall/a1b2c3d4e5f6/capture_som_1234567890123456789.png",
  "elements_som": [...],
  "a11y_tree": "...",
  "evaluations": [...],
  "latences_actions": [
    {"index": 0, "type": "naviguer", "latence_ms": 842},
    {"index": 1, "type": "cliquer_som", "latence_ms": 63}
  ],
  "respect": {
    "pages_visitees": 0,
    "actions_executees": 3,
    "duree_totale_ms": 2400,
    "indice_agressivite": 0.33
  },
  "etat": {
    "pret_a_agir": true,
    "niveau_confiance": "eleve",
    "raisons": ["aucun signal de friction détecté"]
  },
  "boussole": {
    "utilisateur": "operator",
    "ip_locale": "__IP_LAN__",
    "repertoire": "/opt/diwall",
    "operation_id": "a1b2c3d4e5f6",
    "url_courante": "https://target.local/dashboard",
    "titre_page": "Dashboard — My App",
    "stealth_actif": true,
    "shadow_dom_actif": true,
    "auth_status": "active",
    "som_hors_viewport": 0,
    "respect": { "pages_visitees": 0, "actions_executees": 3, "duree_totale_ms": 2400,
    ↪  "indice_agressivite": 0.33, "som_resolution": "stable" }
  },
  "diwall_meta": {
    "version_shot": "1.23.0",
    "profil": "operator",
    "modeles_appeles": []
  }
}
```


`operation_id` (v1.16.0) ist immer vorhanden und identifiziert diesen Durchlauf eindeutig – es benennt das Isolationsverzeichnis unter `/tmp/diwall/<operation_id>/` und entspricht dem Feld `operation_id` des Eintrags dieses Durchlaufs im Operations-Log (Abschnitt 9). `etat` (v1.16.0) ist nur auf dem erfolgreichen Pfad vorhanden. `latences_actions` (v1.20.0) ist immer vorhanden (leere Liste, wenn keine Aktionen ausgeführt wurden), ein Eintrag pro Aktion, die tatsächlich ausgelöst wurde – siehe `GUIDE_LLM_MONITORING.md` zur Ergänzung von `respect.duree_totale_ms`.

Bedingte Schlüssel (fehlen, wenn inaktiv): `capture`, `capture_som`, `elements_som`, `a11y_tree`, `evaluations`, `auth_status`, `stealth_actif`, `shadow_dom_actif`, `som_brut_actif`, `som_hors_viewport`,

`session_derive`, `respect.plafond_atteint`, `respect.waf_bloquants`, `respect.som_resolution` (vorhanden, wenn eine SoM-Aktion abgeschlossen wurde), `respect.som_derive_detectee` (nur bei einer stabilen/rohen Divergenz vorhanden), `respect.indice_agressivite` (vorhanden, wenn mindestens eine Aktion ausgeführt wurde), `actions_executees_avant_echec`, `pages_visitees_avant_echec` (nur im Fehler-JSON, v1.17.0), `etat.mode_conseille` (nur mit realen vorherigen `diagnostic_dom.json` Daten für diesen Host, v1.18.0, Abschnitt 2e).

Fehler — Formatierung

```
{
  "succes": false,
  "erreur": "secrets_fermes",
  "message": "Le répertoire chiffré Diwall est initialisé mais non monté.",
  "code_sortie_recommande": 42,
  "boussole": { "url_courante": "", "titre_page": "" }
}
```

Referenzpfade

Pfad	Rolle
`/opt/diwall/`	Produktionsinstallation
`/opt/diwall/venv/bin/python3`	Python-Version für jeden Aufruf
`/opt/diwall/diwall.conf`	Maschinenkonfiguration (Zugangsdaten, Navigation, Protokollierung)
`/opt/diwall/diwall-sample.conf`	Konfigurationsvorlage
`/opt/diwall/scenarios/`	RPA-Szenarien
`/opt/diwall/docs/`	Dokumentation
`/opt/diwall/references/`	Visuelle Referenzen watch.py
`/tmp/diwall/<operation_id>/`	Temporäre Daten für einen Lauf, isoliert durch `operation_id` (Version 1.16.0, wird beim Neustart gelöscht)
`~/Vaults/__PROJET__/Diwall/`	Zugangsdaten + Protokoll (gocryptfs-Volume)
`~/git/Diwall/Diwall/`	Git-Quellen (hier ändern, dann `deploy.sh`)

Implementieren Sie die Änderungen nach der Modifikation der Quelldateien:

```
bash ~/git/Diwall/Diwall/scripts/deploy.sh
```