

# Diwall — Documentación

*Percepción visual y RPA para agentes LLM*

Versión 1.24.5

2026-09-30

Traducción. La versión inglesa es la que prevalece.

## Acerca de esta compilación

Este documento reúne cuatro textos que también existen por separado, cada uno escrito para un lector distinto. La *ficha de síntesis* abre con las órdenes mismas, para quien necesita una de inmediato. La *presentación* introduce Diwall y lo instala. La *guía del operador* explica qué se delega y por qué la herramienta se comporta así. El *manual de operación* es la referencia: órdenes, acciones, opciones y códigos de salida. Cada uno abre con su propia introducción, porque cada uno también se lee solo.

# Índice

**Diwall — guía rápida** **4**
Tres comandos . . . 4
El bucle . . . 5
Leer la salida en este orden . . . 5
Cada acción . . . 5
Credenciales: la única forma correcta . . . 6
Cuando algo se resiste . . . 6
Códigos de salida . . . 6

**Diwall: Referencia visual compartida entre humanos y modelos de lenguaje grandes (LLM)** **6**
¿Qué es Diwall? . . . 7
Lo que el modelo realmente recibe . . . 7
Arquitectura . . . 8
Capacidades . . . 8
Requisitos . . . 10
Instalación . . . 10
Paquete — la vía sencilla . . . 10
Desde la fuente: para modificar Diwall en sí mismo . . . 11
Desinstalación . . . 11
Uso (por parte del modelo de lenguaje) . . . 12
Captura simple . . . 12
Bucle de ReAct (navegación en múltiples pasos) . . . 12
Escenario RPA . . . 12
Credenciales . . . 12
Seguridad . . . 13
Almacenamiento de capturas . . . 13
Modelos locales versus modelos en la nube . . . 13
Directorio de credenciales . . . 13
Documentación en otros idiomas (v1.23.0) . . . 13
Para los modelos de lenguaje (LLM) que descubren Diwall . . . 14
Créditos . . . 14
Licencia . . . 14

**Diwall — Guía del operador** **14**
¿Por qué Diwall? Lo que realmente delega . . . 14
El problema que resuelve Diwall . . . 14
Lo que realmente se delega . . . 15
Lo que conserva . . . 15
Navegación Respetuosa (v1.15.0) . . . 15
Cuándo Diwall es la herramienta adecuada . . . 15
Casos de uso de demostración . . . 16
Caso 1: Solución de problemas de CSS/JavaScript locales . . . 16
Caso 2: Comparación de componentes de hardware entre diferentes tiendas . . . 16
Caso 3: Exploración y resumen de documentación técnica (aplicaciones de una sola página) 17
Caso 4: Configuración de un panel de control de observabilidad o análisis alojado en su propia infraestructura . . . 17
Caso 5: administración integral de una plataforma de gestión de tickets . . . 17
Caso 6: seguimiento de un calendario de eventos regionales . . . 17
Caso 7: Pruebas de acceso a sitios de comercio electrónico en condiciones reales, utilizando el enfoque ”Respectful Navigation” . . . 18

Requisitos previos antes de comenzar . . . 18
Configuración de credenciales por proyecto . . . 18
Capturar una página y analizarla . . . 19
Automatización de un formulario de inicio de sesión . . . 19
Validación de múltiples páginas en una única ejecución . . . 20
Extracción de un valor de la página . . . 20
Configuración de la monitorización visual . . . 20
Configuración de la monitorización estructural continua (v1.18.0) . . . 21
Errores comunes . . . 21
Desinstalación de Diwall . . . 22
Consultando el historial de operaciones . . . 23

**Diwall — Manual de operación** **23**
Tabla de contenidos . . . 23
1. Verificar la instalación . . . 24
1a. Instalación desde un paquete: el camino más sencillo. . . . 24
1b. Instalación desde el código fuente: para modificar Diwall en sí mismo . . . 25
2. Capturar una página . . . 26
2a. Captura rápida: solo texto, sin imágenes PNG (aproximadamente 2 segundos) . . . 26
2b. Captura visual completa con elementos numerados . . . 26
2c. Lee la brújula primero . . . 27
2d. Leer `etat` para tomar una decisión de "sí/no" (v1.16.0) . . . 27
2e. `mode_conseille` — consejos de configuración previa al vuelo (v1.18.0) . . . 28
3. Navegación Respetuosa (v1.15.0) . . . 28
3a. Modo sigiloso `--stealth` . . . 28
3b. Retrasos por cortesía . . . 28
3c. Métricas de impacto . . . 29
3d. Prueba de rendimiento "Stealth" - cuantitativa (v1.17.1) . . . 29
3e. Señal de detección de WAF (versión 1.16.0, refinada en la versión 1.17.2) . . . 30
3f. Idioma declarado (v1.24.1) . . . 30
4. Directorio cifrado y credenciales . . . 31
4a. Estructura . . . 31
4b. Rellenar un formulario: la regla absoluta . . . 31
4c. Elegir el archivo de credenciales para una ejecución . . . 31
4d. TOTP / Autenticación Multifactorial . . . 32
4e. Suma de comprobación de integridad (opcional, v1.15.0) . . . 32
4f. Directorio cifrado cerrado: ¿qué hacer? . . . 32
4g. Autenticación básica de HTTP — `--http-credentials` (v1.21.0) . . . 32
5. Escribe y ejecuta un escenario de automatización robótica de procesos (RPA) . . . 33
5a. Protocolo de 3 pasos . . . 33
5b. Escenario completo: iniciar sesión y navegar entre páginas . . . 34
5c. Extraer datos del DOM . . . 34
5d. Afirmaciones sobre evaluer (rpa.py solamente) . . . 34
5e. Subescenarios (declencher_scenario) . . . 35
5f. Verificar que la página es la correcta antes de cualquier modificación . . . 35
5g. Reanudar una sesión (cookies persistentes) . . . 35
5h. Estabilidad estructural sin cambios visuales a nivel de píxel — `--replay-verifier` (v1.17.0) . . . 35
5i. Reanudar un escenario largo después de una falla — `--checkpoint` (v1.17.0) . . . 36
5j. Elementos objetivo dentro de un iframe (v1.17.0) . . . 36
5k. Iframes anidados — `iframe_chemin` (v1.18.0) . . . 37
6. Acciones — referencia completa . . . 37

7. Manejar obstáculos comunes . . . 38
7a. Banner de cookies / superposición de bloqueo . . . 38
7b. Elemento fuera de la ventana visible . . . 38
7c. Componentes web: Shadow DOM . . . 39
7d. Aplicaciones web de una sola página (SPA) (React, Vue, Angular) — navegación sin recarga 39
7e. Diálogo de CSS o `showModal()` . . . 39
7f. Operación prolongada (indicador de carga, trabajo por lotes) . . . 39
7g. Límite alcanzado (v1.15.0) . . . 40
7h. `<select>` campo de formulario . . . 40
7i. Identificadores de SoM inválidos en la siguiente ejecución . . . 40
7j. Desfase del ID de SoM en páginas altamente dinámicas: solución híbrida (v1.24.0) . . . 40
7k. Sitio bloqueado por el WAF (error 403 inmediato) . . . 40
7l. La navegación inicial nunca se completa — `--wait-until` (v1.22.0) . . . 41
8. Monitoreo visual — watch.py . . . 41
8a. Guardar una referencia . . . 41
8b. Comparar con la referencia (diferencia de píxeles) . . . 41
8c. Comparación semántica (modelo de lenguaje grande) . . . 42
8d. Ignorar una zona animada . . . 42
8e. Bucle de monitoreo . . . 42
8f. Cron para el monitoreo autónomo . . . 42
8g. Monitoreo estructural continuo — `monitor-verifier.sh` (v1.18.0) . . . 43
9. Registro de operaciones . . . 43
9a. Rotación de registros (G-36, CHANTIER_SANITISATION.md) . . . 44
10. Flags de la línea de comandos: referencia . . . 44
shot.py . . . 44
rpa.py . . . 46
watch.py . . . 46
Códigos de salida . . . 47
Estructura del JSON de salida . . . 47
Error — formato . . . 49
Rutas de referencia . . . 49

# Diwall — guía rápida

Versión 1.24.5 — Septiembre de 2026

Todo en una página. Referencia completa: `docs/MANUEL.md`.

---

## Tres comandos

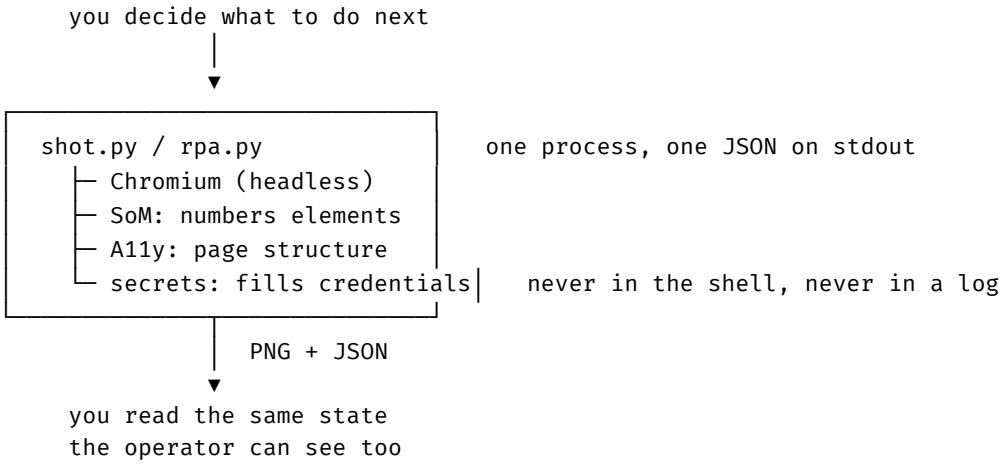
```
# Ver una página: PNG + elementos numerados + árbol de accesibilidad.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url URL --som --a11y

# Leer sin captura (~2 segundos más rápido, sin archivos PNG).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url URL --mode fast

# Ejecutar un escenario.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py --scenario FILE.json
```

¿Instalado desde el `.deb`? Use `diwall-shot` y `diwall-rpa` en lugar de las rutas completas. La primera llamada en una máquina necesita `--guide-version X.Y`, que se lee con `grep notice-version /opt/diwall/docs/GUIDE_LLM.md`.

## El bucle




El estado de la sesión se guarda en un archivo, no en el proceso: una segunda llamada con `--reprendre-session` reutiliza las cookies; nunca el estado del DOM.

## Leer la salida en este orden

| Leer | Qué indica |
|---|---|
| `succes` | si la ejecución llegó a término |
| `boussole.url_courante` | dónde se terminó realmente |
| `boussole.dernier_code_http` | código de la última navegación |
| `etat.pret_a_agir` + `etat.raisons` | fricciones percibidas — un informe, nunca una barrera |
| `capture_som` / `elements_som` | qué pulsar, y con qué número |
| `respect` | la propia huella: páginas, acciones, duración |

Si `boussole` no coincide con lo esperado, deténgase antes de cualquier acción que modifique algo.

## Cada acción

`type` siempre es obligatorio. Las claves siguientes son las adicionales.

| Acción | Requerido | Opcional |
|---|---|---|
| `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` |
| `capturer` | `nom` | `som` |
| `evaluer` | `script` | `attendu` \| `contient` \| `motif` |
| `defiler` | `px` \| `selecteur` | — |

| Acción | Requerido | Opcional |
|---|---|---|
| `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` | — |

---

### Credenciales: la única forma correcta

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

Nunca extraiga un secreto al shell. `lib/repertoire_chiffre.py` lo resuelve dentro del proceso de Playwright; el valor nunca llega a su línea de órdenes, a su historial ni a ningún registro. `depuis_secrets_totp` hace lo mismo con un código TOTP.

---

### Cuando algo se resiste

| Síntoma | Intente |
|---|---|
| El tiempo de espera se agota al hacer clic, el elemento está visualmente oculto | `"force": true`, luego `"repli_js": true` |
| El elemento no está numerado por SoM | `--shadow-dom` (habilitar Shadow Roots) |
| Elemento que está fuera de la pantalla visible | `defiler` primero; verifique `boussole.som_hors_viewport` |
| La página nunca termina de cargar | `--wait-until load` |
| El botón ”Enviar” no hace nada, sin error | validación HTML nativa; envíe el formulario a través de `evaluer` |
| `exit 42` | directorio cifrado no montado: `diwall-monter-secrets` |
| `exit 43` | no hay `diwall.conf` — copie la muestra que está al lado |
| `guide_non_lu` | pase `--guide-version` una vez |
| 403 / 429 | lea `respect.waf_bloquants` — es una señal, no una excepción |

---

### Códigos de salida

`0` éxito · `1` error de Playwright o aserción fallida · `2` viewport no coincidente (`watch.py`) · `3` intérprete incorrecto, use el venv · `42` directorio cifrado cerrado o suma de control incorrecta · `43` falta `diwall.conf`.

## Diwall: Referencia visual compartida entre humanos y modelos de lenguaje grandes (LLM)

> **Para el operador humano**: Diwall le permite delegar la verificación visual a su modelo de lenguaje. Ambos ven la misma captura — ya no tiene que confiar a ciegas en su palabra.



Figura 1: Logotipo de Diwall: una pared en forma de D cuyo recorte interior es un ojo abierto

**Para el modelo de lenguaje**: `docs/GUIDE_LLM.md` es su referencia operativa. Empiece por ahí. Si es un agente de IA que está descubriendo Diwall, omita la página de inicio con formato y obtenga sus instrucciones directamente: **https://diwall.davalan.fr/instructions.md**

---

## ¿Qué es Diwall?

Diwall crea una **referencia visual compartida** entre un operador humano y un modelo de lenguaje. Le da al LLM la capacidad de **ver interfaces web** y le brinda al operador humano una forma de **delegar la verificación visual** sin perder el control.

Sin Diwall, el operador debe confiar a ciegas en la palabra de su modelo de lenguaje o verificar el resultado por sí mismo. Con Diwall, ambas partes ven la misma captura PNG y el mismo árbol de accesibilidad. La duda desaparece en ambos lados.

```
El LLM actúa → Diwall captura → el LLM ve e informa → el operador verifica desde el mismo
↪  estado
```

**Lo que gana el humano**: la delegación del trabajo repetitivo y estresante de verificación visual. En lugar de revisar docenas de páginas después de una implementación, el humano revisa los resultados ya generados por el LLM.

**Lo que gana el modelo**: una percepción real de la interfaz. Sin Diwall, un modelo que desarrolla una aplicación web modifica código pero no puede ver el resultado en un navegador. `lynx` no procesa las interfaces modernas.

### Lo que el modelo realmente recibe

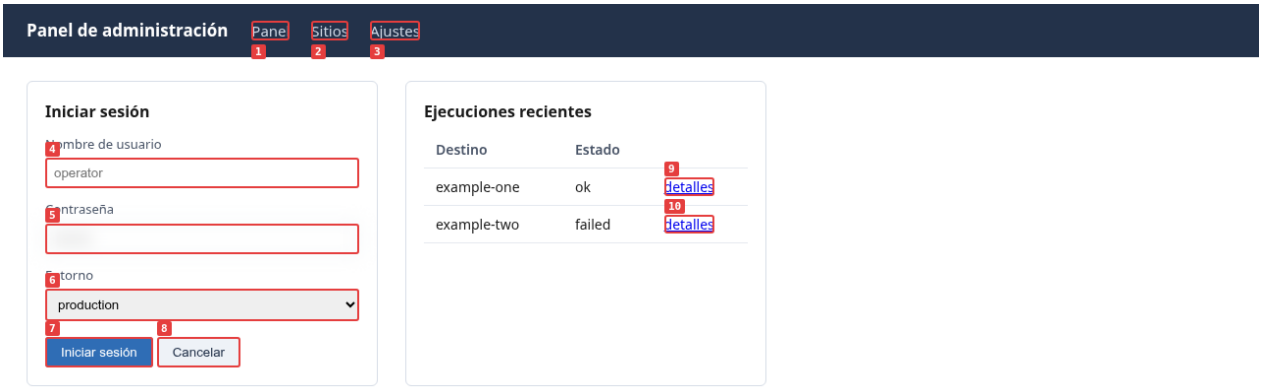



Figura 2: Captura Set-of-Mark: cada elemento interactivo numerado sobre la página renderizada

Se trata de una captura `--som` real, no de una maqueta. Cada elemento interactivo está numerado en la página renderizada, y los mismos números vuelven en el JSON — de modo que `{"type": "cliquer_som", "id": 7}` pulsa *Sign in*, sin selector que adivinar y sin ambigüedad sobre qué botón se pretendía. Reprodúzcalo usted mismo — la página es una fixture versionada en este repositorio, así que obtendrá los mismos números que nosotros:

```
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` regresa con `{"id": 7, "tag": "BUTTON", "texte": "Sign in"}`.

---

## Arquitectura

```
Modelo de lenguaje (el cerebro — bucle ReAct)
        ↓  invoca
  shot.py (las manos — ejecutor Playwright)
        ↓
  Chromium headless → captura PNG
        ↓
  El modelo de lenguaje lee el PNG directamente (multimodal)
```

`shot.py` no tiene inteligencia. Ejecuta instrucciones y devuelve el estado. El modelo de lenguaje decide qué hacer a continuación.

---

## Capacidades

| Característica | Descripción |
|---|---|
| **Captura** | Captura una captura de pantalla de cualquier página web |
| **Acciones** | Rellena formularios, haz clic, navega |
| **Conjunto de marcas (SoM)** | Numera todos los elementos interactivos para clics precisos en el DOM |
| **Instantánea de accesibilidad** | Extrae la estructura semántica de la página (árbol A11y) |
| **Persistencia de sesión** | Mantiene el estado de inicio de sesión a través de bucles ReAct de varios pasos |
| **Escenarios de RPA** | Ejecuta secuencias de acciones desde archivos JSON |
| **Monitoreo visual** | Detecta si una página ha cambiado desde la última referencia |
| **Diferencia de píxeles** | Diferencia cuantitativa y determinista con respecto a una referencia almacenada (v1.2) |
| **Resolución de credenciales** | Inyección segura de credenciales; nunca en texto plano, nunca en la línea de comandos |
| **Directorio cifrado** | Volumen gocryptfs — `SecretsFermesError` (salida 42) si no está montado (v1.5) |
| **Desplazamiento** | Acción `defiler` — desplazamiento relativo de píxeles o por selector CSS `scrollIntoView` (v1.6) |
| **Advertencia fuera de pantalla** | Cuenta `som_hors_viewport` en JSON cuando existen elementos interactivos debajo del pliegue (v1.6) |
| **Memoria procedimental** | Las ejecuciones exitosas se almacenan como habilidades reproducibles a través de `journal.py --exporter-skill` (v1.6) |
| **TOTP 2FA** | Códigos de Google Authenticator / Authy generados en tiempo de ejecución desde una semilla almacenada (v1.6) |
| **MFA asíncrono a través de ntfy** | Códigos 2FA de SMS/correo electrónico recibidos de forma asíncrona a través de la notificación push de ntfy (v1.6) |

| Característica | Descripción |
|---|---|
| **Perfil de operador** | Perfil YAML para eliminar confirmaciones administrativas repetitivas (v1.3) |
| **Trazabilidad del modelo** | Cada ejecución registra qué modelos se llamaron, incluido el resumen de Ollama (v1.3) |
| **Registro de operaciones** | Registro persistente y de solo escritura de todas las ejecuciones: quién hizo qué, dónde, cuándo (v1.4) |
| **Recorrido del DOM sombreado** | Números `--shadow-dom` de elementos interactivos dentro de Shadow Roots abiertos; Angular, Lit, Stencil, FAST (v1.13.0) |
| **Navegación Respetuosa** | `--stealth` (elimina los marcadores automáticos del modo headless), retrasos de cortesía y límites estrictos (`min_action_delay_ms`, `max_pages_par_run`, `max_actions_par_run`), métricas de impacto (`respect`) informadas en cada ejecución (v1.15.0) |
| **Veredicto determinista** | El objeto `etat` (`pret_a_agir`, `niveau_confiance`, `raisons`) sintetiza señales de autenticación, deriva de sesión y fricción en una sola lectura (v1.16.0) |
| **Identidad de ejecución unificada** | `operation_id` aísla los archivos temporales de cada ejecución y los vincula a su entrada del registro de operaciones (v1.16.0) |
| **Señal pasiva de WAF** | `respect.waf_bloquants` marca un posible bloqueo (HTTP 403/429 o palabras clave conocidas) como una señal no fatal, nunca como una excepción (v1.16.0) |
| **No regresión estructural** | `--replay-verifier` compara el estado HTTP, las estadísticas del DOM y los resultados de `evaluer` con una referencia guardada; sin píxeles, sin modelo de visión (v1.17.0) |
| **Puntos de control del escenario** | `--checkpoint` reanuda un escenario largo después de un fallo a la mitad sin reproducir las acciones completadas (v1.17.0) |
| **Identidad SoM híbrida** | `cliquer_som`/`remplir_som` resuelven mediante el marcador `data-dw-som-id` cuando existe, recurren en caso contrario a una reindexación en vivo, e indican la vía seguida y cualquier divergencia entre la vía estable y la bruta en `boussole.respect.som_resolution` / `som_derive_detectee` — nunca en silencio (v1.24.0; `--som-brut` fuerza la reindexación pura) |
| **Iframes entre orígenes** | Los elementos de destino `cliquer_iframe` / `remplir_iframe` dentro de iframes del mismo origen o de diferentes orígenes a través de la API de marco nativa de Playwright (v1.17.0) |
| **Iframes anidados** | Descenso `iframe_chemin` (matriz) iframe-dentro-de-iframe, mutuamente excluyente con `iframe_selecteur` (v1.18.0) |
| **Bloqueo de guía-lectura** | `shot.py`/`rpa.py`/`watch.py` se niega a ejecutarse sin una prueba de que se leyó `docs/GUIDE_LLM.md` — un marcador local lo persiste por máquina/usuario (v1.18.0) |
| **Consejos de configuración** | `mode_conseille` recomienda `--mode`/`--shadow-dom` a partir de ejecuciones de diagnóstico reales anteriores en el mismo host; nunca una conjetura (v1.18.0) |
| **Trazabilidad de escenarios encadenados** | `chainage` registra el árbol de llamadas ordenado de escenarios encadenados a través de `declencher_scenario`, que se muestra en el registro de operaciones (v1.19.0) |
| **Temporización por acción** | `latences_actions` informa la latencia de despacho para cada acción ejecutada, siempre presente (v1.20.0) |

| Característica | Descripción |
|---|---|
| **Vista de registro solo de errores** | `journal.py --erreurs` filtra el registro de operaciones para mostrar solo las ejecuciones fallidas (v1.20.0) |
| **Autenticación HTTP básica** | `--http-credentials` resuelve la autenticación Básica a nivel de red (RFC 7617) desde el archivo de credenciales, con ámbito del origen del objetivo; distinto y adicional a la autenticación basada en formularios (v1.21.0) |
| **Escalado de clics de JavaScript** | `repli_js` en `cliquer` reintenta un clic nativo fallido mediante JS, informado solo en la boussole cuando realmente se ejecutó (v1.22.0) |
| **Objetivos que nunca están inactivos** | `--wait-until load\|domcontentloaded` alcanza páginas que realizan sondeos continuos y nunca permanecen sin actividad de red, donde ningún valor de `--timeout` sería suficiente (v1.22.0) |

---

## Requisitos

| Componente | Versión / Notas |
|---|---|
| **Sistema operativo** | Linux. Paquetes para Debian 13, Ubuntu 24.04 y 26.04, Fedora 44, openSUSE Leap 16.0 y Arch Linux — consulte Instalación para ver lo que se validó en cada uno. No probado en macOS ni Windows |
| **Servidor de visualización** | Wayland (Playwright se ejecuta en este ecosistema) |
| **Python** | 3.11+ en un entorno virtual aislado (PEP 668 — pip del sistema bloqueado en Debian 13) |
| **Playwright** | 1.50+ (instalado en el entorno virtual) |
| **playwright-stealth** | 2.0+ — requerido para `--stealth` (v1.15.0). Incompatible con la API de la versión 1.x |
| **Chromium** | Sin interfaz gráfica, instalado a través de `playwright install chromium` |
| **Ollama** | Modelos de visión locales para `cliquer_visuel` y `watch.py` |
| **GPU** | Recomendado: NVIDIA RTX 3060 de 12 GB de VRAM o equivalente (para los modelos qwen3-vl de Ollama) |

---

## Instalación

Dos canales, **mutuamente excluyentes en una misma máquina**. Elija el paquete a menos que tenga la intención de modificar el código propio de Diwall.

### Paquete — la vía sencilla

Descargue el archivo para su distribución desde la última versión, luego:

```
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
```

Crea el usuario del sistema `diwall`, el entorno virtual y `/opt/diwall/`, instala los seis comandos `diwall-*` en su `PATH`, y proporciona la página de manual:

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

La instalación requiere acceso a la red: instala las dependencias de Python y descarga Chromium.

**Qué significa aquí «validado»**. En cada uno de estos cinco sistemas, el paquete se instaló en un contenedor limpio, `diwall-shot` lanzó Chromium de verdad sobre una página local y después se desinstaló el paquete sin dejar nada que no debiera quedar. Un contenedor no demuestra ni la visualización ni el montaje del directorio cifrado de credenciales (no hay dispositivo FUSE en un contenedor). Debian 12 y Ubuntu 22.04 no están soportados. Linux Mint 22 se basa en Ubuntu 24.04, pero no se ha probado como tal. Playwright solo soporta oficialmente Debian y Ubuntu: en Fedora, openSUSE y Arch, Diwall funciona porque los paquetes declaran las bibliotecas que Chromium necesita, no porque Playwright lo garantice.

La configuración se encuentra en `/etc/diwall/diwall.conf`; una muestra comentada está instalada junto a ella como `diwall-sample.conf`. Referencia completa de comandos: sección 1a, `docs/MANUEL.md`.

La actualización consiste en instalar el archivo más reciente de la misma manera; su configuración se mantiene.

Cuando una actualización del sistema trae una nueva versión de Python —habitual en Arch Linux, en una actualización de versión en los demás—, Diwall deja de funcionar hasta que se vuelve a instalar su paquete, lo que reconstruye su entorno virtual para el nuevo intérprete:

```
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
```

**Desde la fuente: para modificar Diwall en sí mismo**

Si pretende modificar el código de Diwall, instale desde el repositorio: así las fuentes quedan donde `deploy.sh` puede enviar sus cambios a `/opt/diwall/`. El procedimiento de seis pasos está en `docs/MANUEL.md` sección 1b, junto a las órdenes que ejecutará después.

## Desinstalación

Instalado desde el paquete de Debian:

```
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
```

Instalado desde el paquete de Fedora, openSUSE o Arch:

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

Estos sistemas tienen un único comando de desinstalación. Elimina todo lo que se puede reconstruir (entorno virtual, Chromium, bytecode de Python). Conserva lo que usted ha producido y no podría recuperar reinstalando — `/etc/diwall/diwall.conf`, el registro y las evidencias en `/var/log/diwall/`, las capturas de `watch.py` en `/opt/diwall/references/` — junto con la cuenta `diwall` que las protege, e imprime el comando que las borra. Si no ha producido nada, no queda nada.

Instalado desde el código fuente:

```
# Vista previa de lo que se eliminará (sin modificar nada)
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run

# Desinstalación completa con confirmación interactiva.
bash ~/git/Diwall/Diwall/scripts/uninstall.sh
```

```
# No interactivo (pruebas de integración continua, reinstalaciones).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --confirme
```

Elimina: `/opt/diwall/`, `/var/log/diwall/`, usuario del sistema `diwall`, grupo del sistema `diwall`, pertenencia al grupo de operadores, "git pre-push" hook.

El script se niega a ejecutarse cuando Diwall está instalado mediante un paquete (Debian, Fedora, openSUSE o Arch), y muestra el comando de eliminación que debe usarse en su lugar (`sudo apt purge diwall`, `sudo dnf remove diwall`, `sudo zypper remove diwall`, `sudo pacman -R diwall`). `install.sh` y `deploy.sh` también se niegan en el mismo caso.

No se ha modificado: `~/Vaults/` (sus credenciales), el repositorio en sí mismo, la caché del navegador de Playwright.

Si `/var/log/diwall/preuves/` contiene capturas, se conservan por omisión. Añada `--purge-preuves` para eliminarlas.

---

## Uso (por parte del modelo de lenguaje)

### Captura simple

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

### Bucle de ReAct (navegación en múltiples pasos)

```
# Paso 1: navegar y observar.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://your-app.local/ \
  --sauver-session /tmp/diwall/session.json --som

# Paso 2: Actuar según lo observado.
/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
```

### Escenario RPA

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

Referencia completa para los modelos: docs/GUIDE_LLM.md

---

## Credenciales

Las credenciales se almacenan en archivos JSON, uno por dominio, **nunca en el código ni en los archivos de escenarios**:

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

En un escenario o acción: `"valeur": "depuis_secrets", "secret_cle": "password"` — Diwall lee la credencial en tiempo de ejecución desde el directorio de credenciales.

La ruta es configurable a través de `/opt/diwall/diwall.conf` o la variable de entorno `DIWALL_SECRETS_DIR`.

**Recomendación**: proteja `~/Vaults/Diwall/` con `chmod 700` y encripte con `gocryptfs` (consulte `~/git/Diwall/Diwall/scripts/configurer-repertoire-chiffre.sh --gocryptfs`). El directorio cifrado está completamente soportado desde la versión v1.5.0; si se inicializa pero no se monta, Diwall devuelve una estructura `SecretsFermesError` (código de salida 42) en lugar de fallar silenciosamente.

---

## Seguridad

### Almacenamiento de capturas

Por defecto, las capturas se almacenan en `/tmp/diwall/` con permisos `700` (solo el propietario). No cambie `--output-dir` a una ubicación compartida (`/tmp/`, `~/Desktop/`, etc.)—las capturas pueden contener datos de interfaz confidenciales.

### Modelos locales versus modelos en la nube

Cuando Diwall se utiliza con un LLM basado en la nube (API de Claude, OpenAI, etc.), las capturas PNG se transmiten a servidores externos. Esto es responsabilidad del usuario. Para interfaces que contienen datos privados (credenciales, información del cliente, claves privadas), utilice únicamente modelos Ollama locales.

### Directorio de credenciales

El directorio de credenciales, donde sea que hayas apuntado, por ejemplo, `secrets_dir` —como en `~/Vaults/Diwall/`— contiene credenciales en texto plano JSON cuando está desmontado. Protéjelo:

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

El soporte para sistemas de archivos cifrados (`gocryptfs`) ha sido compatible por completo desde la versión 1.5.0; consulte ”Credenciales” arriba y `~/git/Diwall/Diwall/scripts/configurer-repertoire-chiffre.sh`.

---

## Documentación en otros idiomas (v1.23.0)

El inglés es la versión de referencia y no se mueve. Las traducciones de los documentos dirigidos a personas (este README, `docs/GUIDE.md`, `docs/MANUEL.md`, `docs/CHEAT_SHEET.md` y la página de manual) viven en `docs/fr/`, `docs/de/` y `docs/es/` — un directorio por idioma, junto a los originales en inglés.

Las guías del LLM (`docs/GUIDE_LLM.md` y sus tres avisos) están solo en inglés, deliberadamente. Están protegidas por el ”guide-lock”: una traducción cuya número de versión se resincroniza mecánicamente con contenido obsoleto permitiría que un agente supere la protección habiendo leído instrucciones obsoletas; este es exactamente el fallo que la protección existe para evitar. Un modelo lee inglés de forma nativa, por lo tanto, la ventaja es nula y el riesgo es real.

Se crea un único archivo PDF de referencia por idioma a partir de estas fuentes, en un orden declarado una sola vez y compartido por todos los idiomas. Los archivos PDF se publican en el sitio web en lugar de almacenarse aquí; son artefactos generados, y un repositorio no es un canal de distribución para binarios: https://diwall.davalan.fr/en/guides/downloads/

La propia cadena de traducción y de generación de PDF no está en este repositorio. Produce la documentación; no forma parte de Diwall — necesita `pandoc`, un motor LaTeX y una instancia local de Ollama, ninguno de los cuales es una dependencia de Diwall ni aparece en `requirements.txt`. El markdown traducido es el entregable; la máquina que lo produce es herramienta de mantenedor.

---

### Para los modelos de lenguaje (LLM) que descubren Diwall

Si usted es un modelo de lenguaje que está leyendo este archivo README: consulte `docs/GUIDE_LLM.md` para la referencia técnica completa: patrones de invocación, uso de SoM, integración de credenciales, reglas de navegación de SPA y especificaciones del modelo Ollama.

---

### Créditos

Este proyecto fue desarrollado utilizando un **modelo de colaboración humano-LLM asimétrico**. Los roles están documentados formalmente para reflejar el trabajo real realizado.

**Arquitecto y Árbitro**: Ronan Davalan Visión del producto, requisitos de seguridad, dirección del proyecto, validación y pruebas. Todas las decisiones arquitectónicas son validadas por él.

**Ingeniero de Sistemas y Desarrollador Líder**: Claude Code (Anthropic) Implementación del patrón ReAct, scripts en Python/Bash, gestión compleja de estados, inyección de SoM, persistencia de sesiones. Autor principal del código fuente.

**Sintetizador y Asesor Estratégico**: Gemini (Google) Análisis arquitectónico independiente, resolución lógica de conflictos, optimización de flujos de trabajo, validación cruzada de decisiones técnicas.

**Modelos de percepción (Ollama, local)**: - `qwen3-vl:2b` (Alibaba) — localización por clic y comparación semántica, ~9–19 segundos (predeterminado desde la versión 1.3.1) - `qwen3-vl:8b` (Alibaba) — alternativa robusta, ~114 segundos

**Operadores de mantenimiento (a través de OpenCode)**: - Big Pickle: limpieza semántica exhaustiva de la documentación. - MiniMax: verificación y confirmaciones. - DeepSeek V4 Flash: ponerse al día con las confirmaciones omitidas. - Qwen3.6 Plus: pruebas de roles, incluyendo la documentación de una tarea real desde cero como un modelo sin información previa, lo que reveló dos lagunas en la documentación.

---

### Licencia

MIT — consultar el archivo `LICENSE`.

*Desarrollado en Debian 13 Trixie · Wayland · AMD Ryzen 9 3950X · NVIDIA RTX 3060*

## Diwall — Guía del operador

Versión 1.10 — Septiembre de 2026 (v1.24.5) — cuatro casos de uso de demostración adicionales (observabilidad alojada localmente, administración de plataforma de ticketing, seguimiento de eventos locales, acceso a comercio electrónico bajo ”Navegación Respetuosa”).

*También disponible en francés, alemán y español bajo* `docs/fr/`, `docs/de/` *y* `docs/es/`.

---

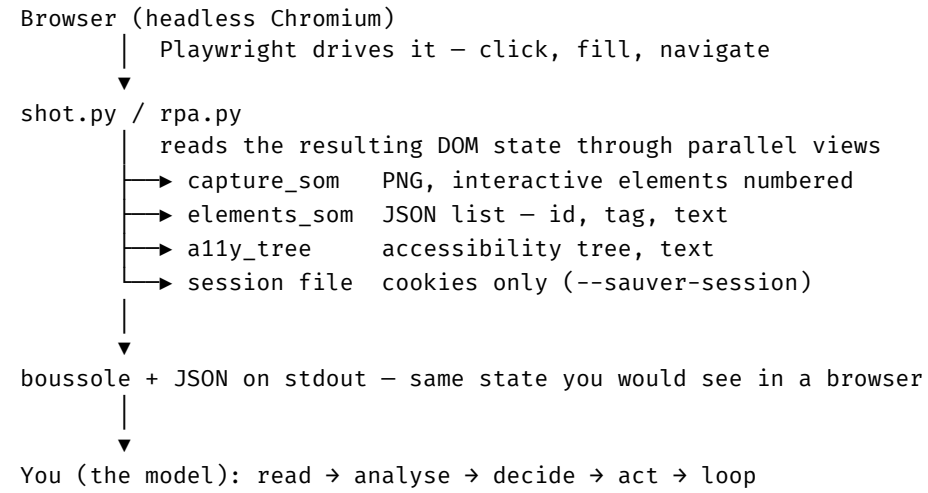
### ¿Por qué Diwall? Lo que realmente delega

#### El problema que resuelve Diwall

Cuando trabaja con un modelo de lenguaje grande (LLM) en una aplicación web, se produce una asimetría perceptual: el modelo lee código, ejecuta comandos y observa la salida textual, pero no ve la interfaz que ven sus usuarios. Usted sí la ve.

Esta asimetría crea una forma específica de ansiedad: usted no sabe si lo que el modelo describe coincide con lo que vería en un navegador. Para estar seguro, debe ya sea confiar en ello sin más, o verificarlo usted mismo.

Diwall resuelve este problema al crear una **referencia visual compartida**: el modelo captura la interfaz con un navegador real (Chromium sin interfaz gráfica), y usted tiene acceso a las mismas capturas PNG y árboles de accesibilidad. Ya no se limita a creer lo que le dice el modelo; usted observa el mismo estado que él.




**Lo que realmente se delega**

Diwall le permite delegar la **verificación visual repetitiva y que genera ansiedad**:

- Verificar que 20 páginas de un sitio se muestran correctamente después de una implementación.
- Confirmar que un formulario de inicio de sesión funciona en la interfaz correcta.
- Asegurarse de que una implementación no haya interrumpido el renderizado de una vista crítica.
- Validar visualmente que una corrección es visible correctamente en la pantalla.

Sin Diwall, estas verificaciones son su responsabilidad. Con Diwall, el modelo las realiza y reporta el resultado, con evidencia visual.

**Lo que conserva**

Usted mantiene la **validación de sentido a nivel superior**: decide si el resultado que presenta el modelo es aceptable, consistente con sus expectativas y acorde con lo que sus usuarios deberían ver. Esa decisión sigue siendo suya.

**Navegación Respetuosa (v1.15.0)**

Diwall no oculta su identidad para evitar la detección por bots. `--stealth` elimina los marcadores técnicos automáticos (`navigator.webdriver`) que bloquean los navegadores sin interfaz gráfica, independientemente de la intención; no cambia la dirección IP del operador, ni su identidad, ni el hecho de que la ejecución esté declarada. A cambio, cada ejecución reporta su propia huella digital (`respect`: páginas visitadas, acciones ejecutadas, duración) y respeta los retrasos configurables y los límites máximos estrictos (`diwall.conf [navigation]`). El derecho a navegar y el deber de navegar de manera medible se consideran inseparables; consulte `docs/RETOUR_EXPERIENCE.md` FR-77/FR-78/FR-79 para el contexto específico que dio forma a esto.

**Destinos locales — el retardo de cortesía no es una doctrina, es un valor por omisión (v1.19.0)**: el `min_action_delay_ms: 800` que se entrega protege una primera ejecución sin configurar frente a la internet pública — carece de sentido contra su propia máquina de desarrollo o producción. Póngalo a `0` en su `diwall.conf` local para depurar en local; véase `docs/MANUEL.md` sección 3b.

**Cuándo Diwall es la herramienta adecuada**

| Caso de uso | ¿Adecuado para Diwall? |
|---|---|
| Validación visual después del despliegue | ✅ Sí |
| Diagnóstico de un error de renderizado | ✅ Sí |
| Navegación y entrada de formularios (máximo ~30 segundos) | ✅ Sí |
| Delegación de comprobaciones repetitivas | ✅ Sí |
| Operación larga del servidor (clonado ~2–5 minutos) | ❌ No — Tiempo de espera de Playwright |
| Eliminación o modificación masiva | ❌ No — Preferir una llamada directa a la API |
| Flujo de trabajo que requiere un "rollback" | ❌ No — Diwall no puede deshacer |

Para casos de desánimo, consulte la sección "Cuándo NO usar Diwall" `docs/GUIDE_LLM.md`. (Se documentan las fricciones FR-59 y FR-60).

---

**Este documento está escrito para la persona que opera Diwall.**

Complementa `GUIDE_LLM.md` (destinado a modelos) con ejemplos concretos, procedimientos paso a paso y recordatorios sobre los puntos más comunes de dificultad.

---

## Casos de uso de demostración

Los casos que se presentan a continuación ilustran cómo puede ser una sesión de "agente-más-Diwall" en la práctica. Están diseñados para que usted los evalúe en relación con su propio contexto, y no como una recomendación para adoptar ninguno específico. Solo el Caso 1 se proporciona como un escenario ejecutable; los demás son narrativos a propósito, y cada uno explica sus motivos bajo su propio encabezado.

### Caso 1: Solución de problemas de CSS/JavaScript locales

Entregado como escenario real y ejecutable: `scenarios/exemples/depannage_local.json`. Diagnostica un desplazamiento visual o una interacción bloqueada en una interfaz servida en local: una sonda rápida (`--mode fast`), la lectura de `erreurs_js`/`erreurs_console`, una captura `--som` si el desplazamiento es puramente visual, y después la validación de la corrección con `watch.py --comparer-pixel` contra una referencia tomada antes de la regresión. Ejecútelo directamente:

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

### Caso 2: Comparación de componentes de hardware entre diferentes tiendas

Un agente que se le pide comparar el precio y la disponibilidad de un componente en varias tiendas online podría usar Diwall con una herramienta separada de descubrimiento de URLs (por ejemplo, una instancia de búsqueda local) para encontrar páginas de tiendas candidatas, luego usar Diwall en modo "sonde" (`--mode fast`, sin PNG) con las acciones `evaluer` para extraer el precio /stock/specifications de cada página, y finalmente comparar los resultados por sí mismo.

**Deliberadamente no entregado como escenario versionado**: nombrar una tienda concreta en un escenario público y versionado es una decisión que le pertenece a usted, no algo que este proyecto deba decidir en su lugar. Conlleva además un riesgo real de fragilidad: un escenario público dirigido a un sitio comercial nombrado puede fallar meses después, cuando cambie la postura anti-bot de ese sitio (el 39 % de los sitios comerciales de la muestra de `docs/RETOUR_EXPERIENCE.md` FR-77 devolvió un bloqueo inmediato), lo que desacredita el ejemplo más de lo que ayuda. Si construye esta composición usted mismo, tenga en cuenta que cualquier herramienta de descubrimiento de URL que combine con Diwall (una instancia de búsqueda local u otra) no es un componente de Diwall: es una pieza aparte que el agente compone por encima.

**Caso 3: Exploración y resumen de documentación técnica (aplicaciones de una sola página)**

Un agente encargado de producir una guía de integración para un sitio de documentación construido como una aplicación de una sola página podría usar `rpa.py` con `attendre_reseau_calme` para permitir que el enrutamiento del lado del cliente se complete, extraer el árbol de accesibilidad en modo rápido para mapear la estructura de la página, luego recorrer los bloques de código recursivamente con `evaluer` para obtener su contenido exacto, y finalmente sintetizar el material recopilado en una guía.

**No se envía como un escenario predefinido, por la misma razón que en el Caso 2**: mencionar un sitio de documentación específico (o, peor aún, un proveedor de pagos específico cuya documentación es el ejemplo funcional) implica un compromiso comercial y de reputación que este proyecto no debería asumir por defecto, y el mismo riesgo de vulnerabilidad del WAF se aplica a un escenario público vinculado a un objetivo real.

**Caso 4: Configuración de un panel de control de observabilidad o análisis alojado en su propia infraestructura**

Un operador que esté configurando un panel de control de monitorización o análisis web alojado localmente detrás de un proxy inverso puede usar Diwall para controlar la propia interfaz: crear un panel, conectar una fuente de datos, configurar una regla de alerta; todo de la misma manera en que se configura cualquier otro panel de administración, en lugar de editar manualmente archivos para tareas que la interfaz de usuario está diseñada para manejar. Esto incluye objetivos que están detrás de un desafío de autenticación HTTP básica a nivel de red (`--http-credentials`, v1.21.0): esto se ha confirmado contra una interfaz de administración protegida por Caddy real, no simplemente una simulación: las credenciales almacenadas respondieron al desafío en el primer intento.

**No se envía como un escenario predefinido** — la disposición del panel y los nombres de las fuentes de datos son específicos de la infraestructura de un operador, e inventar un equivalente sintético duplicaría lo que ya cubre el entorno local en el Caso 1 para la regresión estructural, no para este tipo de trabajo de configuración guiada y con múltiples pasos.

**Caso 5: administración integral de una plataforma de gestión de tickets**

Diwall se utilizó en varias sesiones para configurar y operar una instalación real de ticketing alojada por el usuario: configuración de eventos, categorías de entradas, un dominio personalizado y las herramientas de escaneo/registro del día del evento, todo a través de la misma interfaz web que utilizaría un administrador humano. Se encontraron y resolvieron problemas reales durante el proceso (manejo de sesiones, peculiaridades de los menús desplegables, una solicitud de permiso que bloqueaba un paso automatizado), lo que no resultó en un éxito sin problemas. Esto es parte de lo que hace que sea un ejemplo útil: los obstáculos eran problemas comunes de la automatización web, y no algo específico de Diwall.

**No se envía como un escenario predefinido**— la configuración de los tickets afecta la facturación y detalles específicos del lugar que son únicos para el operador, con la misma lógica que el Caso 2.

**Caso 6: seguimiento de un calendario de eventos regionales**

Un uso sencillo de la función de "semantic probe": pedirle a un agente que revise el calendario de eventos locales para conocer los próximos acontecimientos, sin saber de antemano en qué página se encuentra la respuesta. El modo rápido de Diwall (`--mode fast`, sin captura) combinado con el árbol de accesibilidad permite al agente escanear y reportar resultados en pocas solicitudes; no se necesita un modelo de visión para este tipo de tarea de solo lectura y basada en texto. Una sesión también produjo un ejemplo claro y real del comportamiento documentado de falsos positivos de la señal WAF: una página se cargó normalmente (contenido enriquecido, sin captcha, sin intersticial) mientras que `[respect.waf_bloquants]` aún se activó, debido a un recurso de terceros no relacionado en la página que coincidía con una palabra clave de detección; esto se resolvió en aproximadamente un minuto al leer el árbol de accesibilidad ya presente en la misma respuesta, tal como lo anticipa la regla del manual "señal, nunca un bloqueo".

**No se envía como un escenario predefinido** — un sitio específico de eventos regionales no es un objetivo público estable y reproducible, y designar uno públicamente es decisión del operador, no una configuración por defecto del proyecto.

**Caso 7: Pruebas de acceso a sitios de comercio electrónico en condiciones reales, utilizando el enfoque "Respectful Navigation"**

Una observación honesta y recurrente de sesiones reales: utilizada con respeto (retrasos limitados por tasa, límites de página/acción, `--stealth` activo, sin intento de forzar el acceso más allá de un bloqueo real), Diwall, al ejecutarse contra una variedad de sitios de comercio electrónico, descubre que una gran proporción de las principales plataformas devuelven un bloqueo directo: HTTP 403, o una solicitud que nunca se completa, independientemente de cuán cortés sea el tráfico. Esto no es una deficiencia de Diwall que deba corregirse: la postura anti-bot es la elección propia del sitio, y Diwall no intenta eludirla (ver "Navegación Respetuosa" arriba). Prácticamente: para tareas de comparación de precios contra grandes plataformas comerciales, espere una proporción significativa de resultados nulos, y trate una señal de bloqueo (`respect.waf_bloquants`) como información para evitar, no como un error para intentar nuevamente.

Una distinción que vale la pena tener en cuenta: una pantalla de verificación invisible que nunca se resuelve y no presenta nada con lo cual interactuar (sin casilla de verificación, sin desafío de imagen) es diferente de un CAPTCHA interactivo. Este último es legítimo para responder honestamente; un agente que opera para un humano específico, desde la propia dirección IP de ese humano, no es el "robot" al que está dirigida la pregunta. La primera opción simplemente no ofrece ninguna vía de acceso desde el lado del agente, y forzar el paso (rotación de IP, suplantación de huella digital TLS) está fuera de lo que Diwall hace.

No se envía como un escenario definido, y deliberadamente no se mencionan las plataformas involucradas — consulte la justificación sobre la fragilidad de los WAF en el Caso 2: una tabla de bloqueo/no bloqueo con fechas que está vinculada a sitios comerciales específicos queda obsoleta y socava su propio propósito más rápido de lo que lo ilustra. `docs/RETOUR_EXPERIENCE.md` FR-77 documenta el mismo patrón a escala de panel (tasa de bloqueo inmediato del 39%).

---

## Requisitos previos antes de comenzar

```
# Verificar que Diwall responda.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://example.com --som --a11y
# → debe retornar {"éxito": verdadero, ...}

# 2. Verificar que el directorio cifrado esté montado (si se utiliza gocryptfs).
ls ~/Vaults/Diwall/
# → debe mostrar archivos .json, no contenido cifrado.

# 3. Verificar las credenciales para un dominio.
/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')
"
```

---

## Configuración de credenciales por proyecto

Cada proyecto puede tener su propio directorio de credenciales. Dos métodos:

**Método 1: Variable de entorno directa (única ejecución):**

```
DIWALL_SECRETS_DIR=~/Vaults/MyProject \
```

```
  /opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url …
```

Método 2: Archivo de proyecto `.diwall.conf` (recomendado para proyectos recurrentes):

```
# Cree el archivo en la raíz del proyecto.
echo '{"secrets_dir": "../MyProject-secrets"}' > ~/git/MyProject/.diwall.conf

# Mantén el formato de Markdown exactamente como está. Responde solo con la traducción, sin
 ↪  preámbulos.  Luego, antepón cada invocación (o exporta al inicio de la sesión del
 ↪  shell).
export DIWALL_CONF=~/git/MyProject/.diwall.conf
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url …
```

El `secrets_dir` en `.diwall.conf` puede ser una ruta relativa; se resuelve relativamente a la ubicación del archivo `.diwall.conf`.

---

### Capturar una página y analizarla

```
# Verificación rápida (sin archivos PNG, ~2 segundos, solo lectura).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --mode fast
# → devuelve url_courante, titre_page, a11y_tree en el formato JSON.

# Captura completa con elementos numerados.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ \
  --som --a11y
# La captura en formato PNG se encuentra en /tmp/diwall/capture_<ts>.png.
```

**Lo que obtiene**: - `boussole.url_courante` + `boussole.titre_page`: URL y título efectivos tras la navegación - `capture`: ruta del PNG de la página tal como se renderizó - `capture_som`: PNG anotado con los números de los elementos - `a11y_tree`: estructura de la página en texto (títulos, campos, botones)

---

### Automatización de un formulario de inicio de sesión

**Paso 1** — Preparar el archivo de credenciales.

El archivo de credenciales se llama `<hostname>.json`, donde `hostname` es el resultado de `urlparse(url).hostname`. Para `https://app.example.com/`, el archivo es `app.example.com.json`.

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

**Paso 2** — Explora la página de inicio de sesión.

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

Abra la imagen PNG con anotaciones (`capture_som`) para identificar los ID de campo.

**Paso 3** — Escriba el escenario.

```
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
```

**Paso 4** — Ejecutar.

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

---

### Validación de múltiples páginas en una única ejecución

Para revisar N páginas de un sitio autenticado sin tener que volver a ingresar cada vez:

```
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
```

---

### Extracción de un valor de la página

Para leer una cadena de texto, un contador o cualquier valor de DOM:

```
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
# → result in evaluaciones[0].valor
```

**Importante**: escriba siempre los scripts JS en un archivo `--actions`, nunca en línea con `--action` (el shell corrompe las comillas anidadas).

---

### Configuración de la monitorización visual

```
# Guarde la referencia visual.
/opt/diwall/venv/bin/python3 /opt/diwall/watch.py \
  --url https://target.local/ --sauver-reference --nom home

# 2. Comparar posteriormente (diferencia de píxeles).
/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
# → veredicto: estable / deriva / regresión (código de salida 0 o 1)

# 3. En una página autenticada: primero captura con rpa.py, luego guarda.
/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
```

---

### Configuración de la monitorización estructural continua (v1.18.0)

Complementa la monitorización visual anterior: esto verifica la *estructura* de la página (código de estado, número de elementos DOM, resultados de la evaluación de JavaScript) en lugar de su *apariencia*. Es más económico y detecta un tipo diferente de regresión (por ejemplo, un campo de formulario que ha desaparecido pero con el diseño sin cambios).

```
# Guarde una referencia estructural, solo una vez.
/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. Una verificación y alerta.
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
```

Silencioso mientras todo es estable, un único aviso `ntfy` cuando se detecta una regresión. Prográmelo usted mismo con cron: el script hace una pasada y termina, no entra en bucle. `scripts/*.sh` nunca se despliega en `/opt/diwall/`, así que la entrada de cron se ejecuta desde el código fuente git, con su propio usuario (no con la cuenta de servicio `diwall`, que no puede acceder a `~/git/Diwall/Diwall/`):

```
# crontab -e (su propio archivo 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
```

---

### Errores comunes

| Situación | ¿Qué hacer |
|---|---|
| `FileNotFoundError` en el archivo de credenciales | Verifique que el archivo JSON tenga el nombre completo del FQDN (`urlparse(url).hostname`) |
| `SecretsFermesError` (salida 42) | Monte el directorio encriptado: `bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh` |
| JSON inválido en la salida | Use `2>/dev/null \| tail -1` para extraer solo la línea JSON |
| Los ID de SoM difieren entre sesiones | Esperado: los ID de SoM se recalculan en cada captura. No los reutilice entre sesiones. |
| Inicio de sesión seguido de una redirección de Django al panel | No use `naviguer` en una sesión de Django reanudada; pase la URL a través de `--url` |

| Situación | ¿Qué hacer |
|---|---|
| El campo `<select>` no está lleno | Use `remplir_som` (no `remplir`) con el ID de SoM del `<select>` |
| El clic no tiene efecto en un botón fuera de la vista | Agregue `{"type":"defiler","selecteur":"#the-button"}` antes del clic |
| `auth_status: "active"` incluso en la página de inicio de sesión | El selector positivo es ambiguo (encabezado persistente): agregue `--auth-indicator-negative .btn-login` |
| Los elementos de Web Components no están numerados por SoM | Agregue `--shadow-dom` (Angular, Lit, Stencil) |
| `respect.waf_bloquants` aparece en una página que en realidad no está bloqueada | La detección se basa en palabras clave (v1.16.0, refinada en v1.17.2): considérelo como una señal, no como un veredicto. Si persiste en una página que ha confirmado que no está bloqueada, agregue `--ignorer-waf` |
| `cliquer_som` hace clic en el elemento incorrecto en una página que mutó entre la captura y el clic | Desde la versión 1.24.0, la resolución es híbrida de forma predeterminada: utiliza el marcador `data-dw-som-id` cuando el mismo escenario capturó el SoM primero, e informa cualquier divergencia estable/cruda como `boussole.respect.som_derive_detectee`. Si aparece esa bandera, vuelva a capturar el SoM antes de actuar. `--som-brut` fuerza la reindexación pura anterior a la versión 1.24.0. |
| Un escenario largo de RPA falla a mitad de camino y no desea reproducir los pasos completados | Agregue `--checkpoint FILE` (v1.17.0): reinicie el mismo comando para reanudar; el estado del DOM no se conserva, solo la sesión y la posición de la acción |
| Los elementos interactivos dentro de un iframe son invisibles para Diwall | SoM no puede numerar el contenido del iframe (mismo origen o diferente origen): use `cliquer_iframe`/`remplir_iframe` (v1.17.0) con un selector CSS explícito, o `iframe_chemin` (v1.18.0) para un iframe anidado dentro de otro |
| Su modelo informa de `"erreur": "guide_non_lu"` / salida 1 en su primera llamada a Diwall | Esperado la primera vez que un modelo utiliza Diwall en esta máquina como este usuario del sistema operativo (v1.18.0): debe leer `docs/GUIDE_LLM.md` y pasar `--guide-version` una vez. Esto es intencional, no un error: indique al modelo que lea la guía en lugar de intentar solucionar el error |

---

## Desinstalación de Diwall

El script `~/git/Diwall/Diwall/scripts/uninstall.sh` elimina la instalación de forma limpia, en orden inverso a `install.sh`.

```
# Observa qué se eliminará, sin hacer nada.
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run

# Desinstalación completa (confirmación interactiva).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh

# Sin confirmación (pruebas sin confirmar, reinstalaciones repetidas).
bash ~/git/Diwall/Diwall/scripts/uninstall.sh --confirme && bash
 ↪  ~/git/Diwall/Diwall/scripts/install.sh
```

¿Qué se elimina:

| Item | Detail |
|---|---|
| `/opt/diwall/` | Código, entorno virtual de Python, configuración |
| `/var/log/diwall/` | Registros de operación |
| `diwall` usuario del sistema | Creado exclusivamente para Diwall |
| `diwall` grupo del sistema | Lo mismo |
| Pertenencia a grupos | Su cuenta se elimina del grupo `diwall` |
| Hook pre-push de git | `core.hooksPath` deshabilitado en el repositorio fuente |

¿Qué nunca debe ser modificado: - `~/Vaults/` — sus credenciales - `~/git/Diwall/` — fuentes de Git - La caché del navegador de Playwright (`~/.cache/ms-playwright/`)

**Capturas de prueba (`/var/log/diwall/preuves/`)**: si el directorio contiene capturas, se conserva por omisión con un aviso. Para eliminarlo:

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

---

### Consultando el historial de operaciones

```
# Todas las operaciones en un objetivo.
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local

# Operaciones de mutación únicamente (clics, entrada de formularios).
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local --mutatif

# Desde una fecha.
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py --cible target.local \
  --depuis 2026-06-01
```

## Diwall — Manual de operación

**Versión 1.24.5 — Septiembre de 2026**

*También disponible en francés, alemán y español bajo* `docs/fr/`, `docs/de/` *y* `docs/es/`.

Este documento responde a una pregunta: **cómo realizar la tarea X con Diwall**.

Si usted es un usuario: no se necesitan comandos. Indique a su modelo qué desea visitar, observar o lograr en un sitio web, una aplicación web o una interfaz de administración. El modelo lee este manual y traduce su intención en las acciones correctas.

Si usted es un modelo de lenguaje: estos son sus comandos. Ejecútelos directamente.

No hay descripciones arquitectónicas. Comandos que funcionan.

---

### Tabla de contenidos

1. Verificar la instalación
2. Capturar una página
3. Navegación segura (v1.15.0)
4. Directorio cifrado y credenciales
5. Escribir y ejecutar un escenario de automatización robótica de procesos (RPA)
6. Acciones: referencia completa
7. Manejar obstáculos comunes
8. Monitoreo visual — watch.py

9. Registro de operaciones
10. Opciones de línea de comandos: referencia
11. Códigos de salida y resultados

---

## 1. Verificar la instalación

```
# Verificación más económica posible: sin Playwright, sin URL, salida 0 inmediata
↪ (v1.18.0+).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --version
# → {"outil": "shot.py", "version": "1.23.0"}

# Prueba completa con un solo comando (aproximadamente 3 segundos).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://example.com --mode fast --guide-version 1.3
```

Resultado esperado: JSON en stdout con `"succes": true`.

**`--guide-version` (v1.18.0+)**: `shot.py`, `rpa.py`, y `watch.py` se niegan a ejecutarse sin él, a menos que ya exista un indicador local de una llamada aceptada anterior (`~/.config/diwall/guide_state.json`). El valor es el `<!-- notice-version: X.Y -->` en la línea 3 de `docs/GUIDE_LLM.md` — no el número de versión de Diwall. Lea el valor actual en lugar de confiar en cualquier valor mencionado aquí: `grep notice-version /opt/diwall/docs/GUIDE_LLM.md`. Consulte la sección "Verificación previa obligatoria" de `docs/GUIDE_LLM.md` para conocer el mecanismo completo y el formato del error si lo omite.

Una vez que el marcador existe, `--guide-version` vuelve a ser opcional — cada otro ejemplo de comando en este manual lo omite deliberadamente, ya que un marcador de cualquier llamada anterior exitosa ya los cubre, siempre y cuando `docs/GUIDE_LLM.md`'s `notice-version` no haya cambiado desde entonces.

```
# Verifique la versión instalada.
grep "__version__" /opt/diwall/shot.py
# → __version__ = "1.23.0"

# Verificar que playwright-stealth esté disponible (v1.15.0).
/opt/diwall/venv/bin/python3 -c "import playwright_stealth; print('stealth OK')"

# Verifique que el directorio cifrado esté montado.
ls ~/Vaults/__PROJET__/Diwall/
# → debe mostrar archivos .json, no una lista vacía.
```

Si `ls ~/Vaults/...` devuelve una lista vacía o un error: → móntalo: `bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh`

### 1a. Instalación desde un paquete: el camino más sencillo.

Los paquetes son recursos de lanzamiento en GitHub. Este es el canal recomendado, a menos que tenga la intención de modificar el código propio de Diwall, en cuyo caso, consulte 1b. Los dos canales son mutuamente excluyentes en una sola máquina; ambos apuntan a `/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
```

Cada paquete fue validado en un contenedor limpio de su sistema: instalación, un lanzamiento real de Chromium por `diwall-shot`, eliminación. Ni la visualización ni el montaje FUSE del directorio encriptado

se pueden probar en un contenedor. Debian 12 y Ubuntu 22.04 no son compatibles; Linux Mint 22 se basa en Ubuntu 24.04, pero no se probó como tal. Playwright oficialmente solo es compatible con Debian y Ubuntu: en Fedora, openSUSE y Arch, Chromium se ejecuta porque el paquete declara las bibliotecas que necesita.

La instalación requiere acceso a la red (la instalación de dependencias y la descarga de Chromium ocurren en la fase de instalación posterior, como root, en `/opt/diwall/.cache/ms-playwright/`). Seis comandos están disponibles, cada uno es una capa delgada: no hay diferencia funcional con las propias invocaciones del canal "git-clone":

| Comando | Envuelve |
|---|---|
| `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` |

La configuración se encuentra en una ruta diferente en este canal: `/etc/diwall/diwall.conf` (no `/opt/diwall/diwall.conf`) — se deja un modelo en `/etc/diwall/diwall-sample.conf`, nunca se activa automáticamente:

```
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` conserva `/var/log/diwall/` (registro de operaciones, evidencias), `/etc/diwall/`, las capturas de referencia de `watch.py` (`/opt/diwall/references/`) y la cuenta `diwall` que las protege (1.24.4: antes la cuenta se eliminaba, lo que dejaba los archivos conservados con un número de grupo huérfano) — `apt purge diwall` lo elimina todo, además de `/opt/diwall/` por completo. Guarde antes `/opt/diwall/references/` si desea conservar sus capturas de referencia.

En Fedora, openSUSE y Arch (`dnf remove`, `zypper remove`, `pacman -R`) hay un único comando de desinstalación. Elimina lo que se puede reconstruir (entorno virtual, Chromium, bytecode de Python) y conserva lo que usted ha producido — `/etc/diwall/diwall.conf`, los archivos de `/var/log/diwall/`, las capturas de `/opt/diwall/references/` — junto con la cuenta `diwall`, y luego imprime el comando exacto que los borra. Si no se ha producido nada, no queda nada.

`~/Vaults/` nunca se toca, en ningún canal.

**Página del manual (v1.22.0):** `man diwall` documenta los seis comandos en una sola página. Los otros cinco nombres de comandos (`man diwall-rpa`, y así sucesivamente) se resuelven a la misma página. Se genera a partir de `debian/diwall.1.md` durante la compilación, por lo que no puede volverse obsoleta silenciosamente; sin embargo, para la lista exhaustiva de opciones de cualquier comando, `--help` sigue siendo la fuente autorizada sobre la página del manual.

### 1b. Instalación desde el código fuente: para modificar Diwall en sí mismo

Use este canal solo si pretende modificar el código de Diwall: coloca el repositorio donde `deploy.sh` puede enviar sus cambios a `/opt/diwall/`. Para un uso corriente, el `.deb` anterior es una sola orden y hace lo mismo.

```
# 1. Crear usuario del sistema y directorio.
sudo useradd --system --no-create-home --shell /bin/false diwall
sudo mkdir -p /opt/diwall
sudo chown root:diwall /opt/diwall

# 2. Clona el repositorio.
```

```
git clone https://github.com/ronandavalan/diwall.git ~/git/Diwall/Diwall
cd ~/git/Diwall/Diwall

# 3. Crear un entorno virtual de Python.
sudo /usr/bin/python3 -m venv /opt/diwall/venv
sudo /opt/diwall/venv/bin/pip install -r requirements.txt

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

# 5. Implementar / Desplegar
bash ~/git/Diwall/Diwall/scripts/deploy.sh

# 6. Crea tu directorio de credenciales cifradas.
mkdir -p ~/Vaults/<your-project>/Diwall
# Cree el archivo `~/Vaults/<su-proyecto>/Diwall/<nombre_de_host>.json` con sus
 ↪  credenciales.
```

Si Diwall ya está instalado mediante un paquete, `install.sh` y `deploy.sh` se niegan a ejecutarse (v1.24.2 para el `.deb`, v1.24.4 para los paquetes de Fedora, openSUSE y Arch): sobrescribirían archivos que gestiona el gestor de paquetes, y la siguiente actualización o eliminación del paquete los desharía sin avisar. Para cambiar de canal, elimine primero el paquete; los scripts muestran el comando exacto (`sudo apt purge diwall`, `sudo dnf remove diwall`, `sudo zypper remove diwall`, `sudo pacman -R diwall`).

En este canal, la configuración es `/opt/diwall/diwall.conf`, no `/etc/diwall/diwall.conf`. Desinstale con `bash ~/git/Diwall/Diwall/scripts/uninstall.sh --dry-run` primero, luego sin la opción. `uninstall.sh` también se niega cuando hay un paquete instalado (v1.24.3 para el `.deb`, v1.24.4 para los demás): eliminaría `/opt/diwall/` y la cuenta `diwall` que gestiona el gestor de paquetes. Muestra el comando de eliminación que debe usarse en su lugar.

---

## 2. Capturar una página

### 2a. Captura rápida: solo texto, sin imágenes PNG (aproximadamente 2 segundos)

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

Devuelve: `a11y_tree` (estructura de texto de la página), `boussole` (URL efectiva, título). Úsalo cuando quieras leer el título, verificar la URL o extraer texto sin capturar una imagen PNG.

### 2b. Captura visual completa con elementos numerados

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

Devuelve: - `capture`: ruta a la imagen PNG de la página. - `capture_som`: imagen PNG con números en los elementos interactivos (SoM). - `elements_som`: lista JSON de elementos (id, etiqueta, texto). - `a11y_tree`: árbol de accesibilidad.

*Lo que `--som` produce. Los números en la imagen son los valores de `id` en `elements_som`, por lo tanto, hacer clic se vuelve `{"type": "cliquer_som", "id": 7}` — no hay ningún selector para adivinar. Generado a partir de una versión de un componente almacenada en este repositorio (`scenarios/interoperabilite/fixture/`); la misma figura existe en francés, alemán y español junto con esta.*

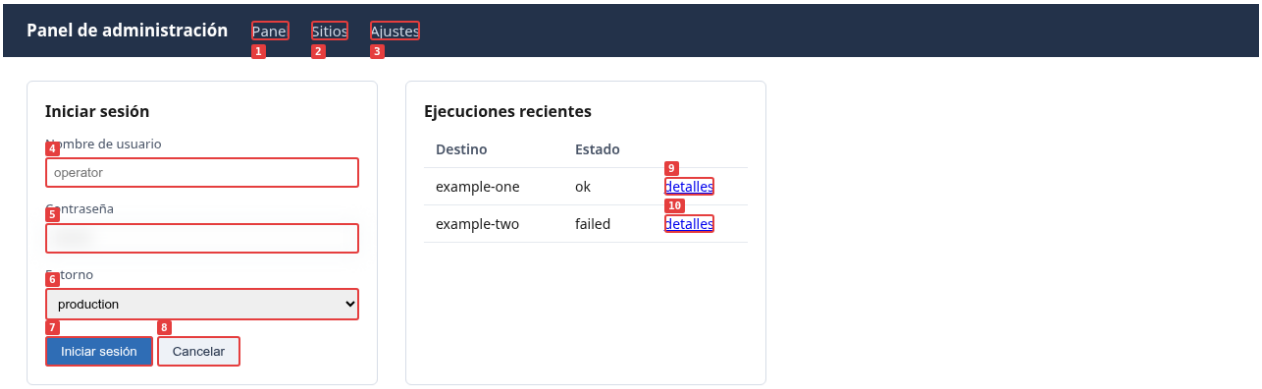



Figura 3: Superposición Set-of-Mark: cada elemento interactivo rodeado y numerado

### 2c. Lee la brújula primero

Cada salida contiene un objeto `boussole`. Léelo antes de cualquier otra cosa:

```
"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
  }
}
```

Si `boussole.url_courante` no coincide con lo que espera: deténgase e investigue antes de cualquier acción que modifique algo.

`boussole.secrets_dir_effectif` y `boussole.secrets_dir_source` (v1.23.1) informan qué directorio de secretos encriptados se resolvió realmente para esta ejecución y de dónde proviene (`env:DIWALL_SECRETS_DIR`, `env:DIWALL_CONF`, `conf:<path>`, o `non_configure`). Una máquina con múltiples proyectos puede, silenciosamente, retroceder al directorio `diwall.conf` a nivel de máquina cuando el que llama olvida exportar `DIWALL_SECRETS_DIR` — estos dos campos hacen que esto sea visible en cada ejecución en lugar de después.

### 2d. Leer `etat` para tomar una decisión de "sí/no" (v1.16.0)

Cada ejecución exitosa incluye un objeto `etat` en la raíz del JSON; léalo antes de cualquier acción que lo modifique, en lugar de verificar manualmente `auth_status`, `respect.plafond_atteint`, `erreurs_js` y `erreurs_console` usted mismo:

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

Si `pret_a_agir` es `false`: consulte `raisons` para la causa (autenticación inactiva, desviación de sesión, límite de navegación alcanzado o un bloqueo de WAF detectado) antes de continuar.

`etat` no verifica si la URL o el contenido de la página coinciden con las expectativas de su negocio; utilice `evaluer` con `attendu`/`contient`/`motif` (sección 5d) para eso.

### 2e. `mode_conseille` — consejos de configuración previa al vuelo (v1.18.0)

Si Diwall tiene datos reales anteriores sobre el host al que llama — procedentes de una ejecución previa de `diagnostic_dom.json` contra él —, `etat` incluye una recomendación para su **siguiente** llamada, nunca aplicada automáticamente:

```
"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"]
  }
}
```

Obtenga estos datos para un host ejecutando el diagnóstico una sola vez:

```
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/diagnostic_dom.json \
  --url https://target.local/ --mode fast
```

No hay un diagnóstico previo para este host → `mode_conseille` está ausente, nunca se hace una suposición. Detalles completos en `GUIDE_LLM_MONITORING.md`.

El campo `som_rafraichir` se mantiene para la estabilidad de la forma de salida, pero es un vestigio desde la versión v1.24.0 (la resolución híbrida SoM con opción de respaldo es la opción predeterminada; no es necesario habilitar nada; consulte la sección 7j).

---

## 3. Navegación Respetuosa (v1.15.0)

### 3a. Modo sigiloso `--stealth`

Algunos sitios bloquean los navegadores sin interfaz gráfica en `navigator.webdriver=true` sin examinar la intención. `--stealth` elimina este marcador técnico automático.

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

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

Cuando está activo: `boussole.stealth_actif = true` en la salida JSON.

**Lo que cambia `--stealth`**: se elimina `navigator.webdriver`, se normalizan plugins/languages/platform.
**Lo que `--stealth` no cambia**: la IP del operador, su identidad ni su intención de navegación.

### 3b. Retrasos por cortesía

Configurado en `/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`: tiempo de retardo mínimo (ms) entre cada acción. Enviado. valor predeterminado: 800 ms.

Desarrollo local: establezca el valor en `0` (v1.19.0): los 800 ms por defecto protegen a un operador distraído durante su *primera ejecución, sin configuración*, contra la conexión a Internet pública; esto no tiene ningún propósito de protección con respecto a su propia máquina de desarrollo/producción, donde nada se espera que se comporte de cierta manera. Establezca la clave explícitamente en su archivo local `diwall.conf`:

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

Mantenga el valor predeterminado de 800 ms (o auméntelo) para cualquier objetivo alcanzado a través de Internet pública. El valor siempre es una elección consciente asociada al objetivo, no una propiedad fija de la herramienta; consulte la guía sobre WAF y técnicas de ocultamiento en `docs/GUIDE_LLM.md` para ver el mismo principio aplicado al comportamiento de bloqueo.

Los límites `max_pages_par_run` y `max_actions_par_run` detienen limpiamente la ejecución si se exceden. No hay excepción: el JSON de salida contendrá:

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

### 3c. Métricas de impacto

Cada ejecución devuelve `respect` (en la raíz del JSON y dentro de boussole):

| Clave | Significado |
|---|---|
| `pages_visitees` | Número de navegaciones `type: naviguer` ejecutadas |
| `actions_executees` | Número total de acciones del escenario ejecutadas |
| `duree_totale_ms` | Duración total de la ejecución |
| `plafond_atteint` | `"max_pages_par_run"` o `"max_actions_par_run"` si hubo parada anticipada |

### 3d. Prueba de rendimiento ”Stealth” - cuantitativa (v1.17.1)

Prefiera contar señales concretas de la huella digital en lugar de comparar capturas de pantalla a simple vista; este es el método utilizado para verificar la compatibilidad de la API v1.17.0 `playwright-stealth`. Corrección de compatibilidad de la API (`docs/RETOUR_EXPERIENCE.md` FR-79):

```
# Sin sigilo.
/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"}]'
```

```
# Con sigilo.
/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"}]'
```

Lea los tres valores de `evaluations[].valeur`: `navigator.webdriver` debe pasar de `true` a `false`, y `td.failed` debe tender a `0`. Medición de referencia (corrección de la v1.17.0, sesión 47): 12 failed → 0 failed. Un número o un booleano devuelto por `evaluer` conserva su tipo JSON en la salida y en el diario (v1.24.3); antes, `shot.py` lo devolvía como texto (`"false"`, `"0"`). Solo el texto se filtra en busca de formas de secretos.

Para una segunda opinión cualitativa, el escenario proporcionado aún genera capturas de pantalla para su inspección:

```
/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` y `capture_intoli_*.png` se guardan en ese directorio. Nota: ambas páginas de destino discuten la detección de bots en su propio contenido, lo que puede activar `respect.waf_bloquants` como un falso positivo (sección 3e) — esto es esperado en esta prueba específica, no una señal de un bloqueo real.

### 3e. Señal de detección de WAF (versión 1.16.0, refinada en la versión 1.17.2)

Diwall detecta una posible obstrucción de un WAF de forma pasiva: HTTP 403/429, o una coincidencia de título/palabra clave en HTML (`Cloudflare`, `CAPTCHA`, `checking your browser`, etc.). Esto es una señal, nunca una excepción; la ejecución se completa normalmente:

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

Cuando estén presentes y `> 0`: `etat.niveau_confiance` es `"faible"` y `etat.pret_a_agir` es `false`. Usted decide si intenta de nuevo con `--stealth`, cambia el objetivo o se detiene; Diwall no interrumpe la ejecución por usted.

Desde la versión v1.17.2, los nombres genéricos de proveedores (`Cloudflare`, `Akamai`) solo coinciden con el título de la página; anteriormente, coincidencias incorrectas se producían en referencias ordinarias a recursos de CDN. Si persiste una coincidencia incorrecta, `--ignorer-waf` degrada `niveau_confiance` sin forzar `pret_a_agir: false` (`boussole.waf_ignore_actif: true` registra la anulación). La detección se basa en palabras clave y puede producir coincidencias incorrectas en páginas que legítimamente discuten el bloqueo/detección (por ejemplo, una página de referencia para la detección de bots); considérela como una señal rápida, no como un veredicto definitivo.

### 3f. Idioma declarado (v1.24.1)

El navegador declara el idioma del operador. Se toma del entorno del proceso, en el orden que Chromium sigue: `LANGUAGE`, luego `LC_ALL`, `LC_MESSAGES`, `LANG` — y `en-US` cuando ninguno de ellos proporciona un valor utilizable (`C`, `POSIX`). La misma etiqueta se envía en el encabezado `Accept-Language` y se devuelve mediante `navigator.language`, por lo que un sitio que negocia su idioma sirve el idioma del operador.

Para declarar otro idioma para una ejecución, configúrelo en el entorno:

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

Antes de la versión v1.24.1, no se enviaba ningún encabezado `Accept-Language` y `navigator.language`

contenía el idioma de la máquina: un sitio que negocia su idioma, utilizaba su valor predeterminado. Un escenario que verifica un texto en un sitio de este tipo (`evaluer` con `contient`, una espera en una etiqueta) puede, por lo tanto, dar un resultado diferente después de la actualización.

---

## 4. Directorio cifrado y credenciales

### 4a. Estructura

Las credenciales se encuentran en un directorio cifrado: un volumen gocryptfs, que contiene un archivo `.json` por dominio.

```
~/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)
```

Formato del archivo de credenciales:

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

El nombre del archivo = `urlparse(url).hostname`. Para `https://app.example.com/login/`, crear `app.example.com.json`.

### 4b. Rellenar un formulario: la regla absoluta

**PELIGRO — expone la contraseña en el shell y `/proc`:**

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

**CORRECTO: las credenciales se resuelven dentro de Playwright.**

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

Los valores nunca pasan por la línea de comandos, el historial de Bash, los registros de procesos ni ningún archivo.

### 4c. Elegir el archivo de credenciales para una ejecución

```
# Directorio de credenciales predeterminadas (definido en diwall.conf > secrets_dir).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url https://target.local/ --som

# Archivo de credenciales explícito (--secrets).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som \
  --secrets /path/to/mounted/directory/creds.json

# Directorio de credenciales específico para cada proyecto a través de .diwall.conf.
export DIWALL_CONF=~/git/MyProject/.diwall.conf
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py --url https://target.local/ --som
```

**Contenido del archivo `--secrets` — `origines_autorisees` obligatorio desde el 05/08/2026** (cambio disruptivo, sin período de compatibilidad): un archivo sin esta clave se rechaza antes de cualquier lectura.

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

`origines_autorisees` enumera los nombres de host contra los cuales este archivo puede ser utilizado. El formato es el mismo que en `domaine_depuis_url()`: minúsculas, sin esquema y sin puerto. Una lectura

contra una página cuyo dominio no está en la lista será rechazada (`SecretsOrigineNonAutoriseeError`).

Contenido de `~/git/MyProject/.diwall.conf`:

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

La ruta se resuelve en relación con la ubicación de `.diwall.conf`.

#### 4d. TOTP / Autenticación Multifactorial

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

Lee la clave `totp_cle` (semilla base32) del archivo de credenciales y genera el código TOTP actual.

Para recibir el código a través de ntfy (flujo de trabajo sin intervención humana):

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

#### 4e. Suma de comprobación de integridad (opcional, v1.15.0)

Para proteger un archivo de credenciales contra la corrupción silenciosa de FUSE, agregue un campo `checksum`:

```
# Genera el valor de suma de verificación.
/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())
"
```

Agregue el valor devuelto al archivo de credenciales:

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

Si la suma de control no coincide, `shot.py` lanza `SecretsChecksumError` (exit 42) con un mensaje explícito. Sin la clave `checksum`: comportamiento sin cambios (opt-in estricto).

#### 4f. Directorio cifrado cerrado: ¿qué hacer?

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

```
# Monte el directorio cifrado.
bash ~/git/Diwall/Diwall/scripts/monter-repertoire-chiffre.sh

# Verifique el montaje.
ls ~/Vaults/__PROJET__/Diwall/
# → debe mostrar archivos JSON
```

#### 4g. Autenticación básica de HTTP — `--http-credentials` (v1.21.0)

Para los objetivos que se encuentran detrás de un desafío de autenticación HTTP Basic (RFC 7617) a nivel de red: un "firewall" que un proxy inverso como Caddy, nginx o Traefik presenta antes de que se renderice cualquier página, común frente a interfaces de administración alojadas internamente. Este es un mecanismo diferente al de la autenticación basada en formularios descrito anteriormente (4a-4f), el cual sigue estando totalmente soportado y no se ve afectado.

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

Archivo de credenciales: el par simple `username`/`password` ya utilizado para el caso común (un único conjunto de credenciales para el destino):

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

Las claves dedicadas `http_username`/`http_password` se prueban primero y solo son necesarias cuando el mismo destino tiene tanto una barrera de autenticación básica a nivel de red como su propio inicio de sesión de aplicación separado (dos pares de credenciales diferentes en el mismo archivo). Diwall recurre automáticamente a `username`/`password` cuando las claves dedicadas no están presentes.

Confirmado en producción contra un objetivo real protegido por Caddy: la configuración predeterminada (`send: "unauthorized"` — las credenciales se envían solo después de un 401 genuino, nunca preventivamente) resolvió el desafío en el primer intento. `boussole.http_credentials_actif: true` confirma un éxito real, no solo que se haya pasado la bandera; `boussole.http_auth_requise: true` distingue claramente entre un 401 sin resolver y un bloqueo de WAF.

---

## 5. Escribe y ejecuta un escenario de automatización robótica de procesos (RPA)

### 5a. Protocolo de 3 pasos

**Paso 1: Explorar la página (solo lectura)**

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

# Vista completa con elementos numerados.
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://target.local/ --som --a11y

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

# Inventario enriquecido del DOM (frameworks, shadow roots, atributos de datos estables).
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/diagnostic_dom.json \
  --url https://target.local/ --mode fast
```

**Qué anotar**: - Los identificadores SoM de los campos y los botones (leer `capture_som`) - Los atributos estables: `name`, `id`, `aria-label`, `data-testid` - Las superposiciones bloqueantes (avisos de cookies, ventanas modales) - SPA o recarga HTTP completa

**Paso 2: Escriba el escenario.**

```
{
  "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"}
  ]
}
```

**Paso 3: Ejecutar**

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

### 5b. Escenario completo: iniciar sesión y navegar entre páginas

```
{
  "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. Extraer datos del DOM

```
{
  "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"}
  ]
}
```

Resultado en `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. Afirmaciones sobre evaluer (rpa.py solamente)

Tres claves mutuamente excluyentes, una por cada acción:

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

| Clave | Comparación | Tipos válidos |
|---|---|---|
| `attendu` | igualdad estricta `==` | str, int, bool |
| `contient` | subcadena `in` | solo str |
| `motif` | `re.search()` de Python | solo str |

Si la aserción falla: rpa.py se detiene de inmediato (exit 1) antes de cualquier acción posterior que modifique algo.

### 5e. Subescenarios (declencher_scenario)

Define un inicio de sesión como un subescenario reutilizable:

```
{
  "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"}
  ]
}
```

Llama a este subescenario desde otro escenario:

```
{
  "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"}
  ]
}
```

Profundidad máxima: 5 niveles de anidamiento.

### 5f. Verificar que la página es la correcta antes de cualquier modificación

Siempre agrega una protección como la primera acción en los escenarios que eliminan o modifican:

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

Si la protección falla: rpa.py se detiene antes de que se ejecute la eliminación.

### 5g. Reanudar una sesión (cookies persistentes)

```
# Primera invocación: autenticar y guardar la sesión.
/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

# Invocaciones posteriores: reutilizar la sesión (sin volver a iniciar sesión).
/opt/diwall/venv/bin/python3 /opt/diwall/shot.py \
  --url https://app.example.com/dashboard/ \
  --reprendre-session /tmp/diwall/session.json \
  --som
```

**Señal de desviación de sesión**: si la sesión ha expirado, poner `boussole.session_derive: true` en el JSON. En ese caso: reiniciar el inicio de sesión completo sin `--reprendre-session`.

### 5h. Estabilidad estructural sin cambios visuales a nivel de píxel — `--replay-verifier` (v1.17.0)

```
# Primera ejecución: guardar la referencia estructural.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/dashboard.json \
  --sauver-verifier-reference /tmp/dashboard.ref.json
```

```
# Ejecuciones posteriores — comparar.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario /opt/diwall/scenarios/dashboard.json \
  --replay-verifier /tmp/dashboard.ref.json
```

Compara los resultados de `http_status`, `dom_stats`, `evaluer`, y el número de elementos de SoM (no contenido) con la referencia guardada. Verificación en stderr:

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

Exit 1 con `verdict: "regression"`, donde `diffs` enumera cada campo divergente (`reference` frente a `obtenu`). Las dos opciones son mutuamente excluyentes.

Una referencia grabada antes de la v1.24.3 guarda como texto un número o un booleano devuelto por `evaluer`; al reproducirla con la v1.24.3 o posterior, se señala como una regresión (`"2"` frente a 2). Vuelva a grabarla con `--sauver-verifier-reference`.

#### 5i. Reanudar un escenario largo después de una falla — `--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
```

Si el escenario falla a mitad de camino, se escribe `/tmp/long_audit.checkpoint.json` con el número de acciones completadas y un archivo de sesión. **Vuelva a lanzar exactamente la misma orden** para retomarlo: las acciones ya completadas se omiten. Si todo termina con éxito, el archivo de checkpoint se elimina automáticamente.

Una ejecución interrumpida por un límite de navegación (`max_actions_par_run`/`max_pages_par_run`) se trata de la misma manera que una falla parcial desde la versión v1.17.2: el punto de control se actualiza con el progreso real, no se elimina. Antes de la versión v1.17.2, también se eliminaba en este caso (devuelve la misma señal `succes: true` que un tramo completado por completo), perdiendo silenciosamente todo el progreso restante en escenarios largos.

El estado del DOM (modales abiertos, formularios parcialmente completados) nunca se guarda al reanudar una sesión; solo se guardan las cookies/`localStorage` y la posición en la lista de acciones. No confíe en `--checkpoint` para continuar un formulario de varios pasos a mitad de proceso; este solo permite la continuación en los límites de acción.

#### 5j. Elementos objetivo dentro de un iframe (v1.17.0)

No se aplica numeración de "Set-of-Mark" dentro de un `<iframe>` (ya sea del mismo origen o de otro origen). En su lugar, selecciónelo directamente mediante un selector CSS:

```
{"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` soporta `valeur: "depuis_secrets"` exactamente como `remplir`. (sección 4b) — nunca se utiliza una credencial en texto plano en este escenario. Si el elemento de destino rechaza la interacción (por ejemplo, un área `contenteditable` en estado de solo lectura), agregue `"force": true` a `cliquer_iframe` — con la misma semántica que `cliquer`. (sección 7e).

Para encontrar el selector interno: utilice `evaluer` en el contenido del iframe si es del mismo origen (`document.querySelector('iframe').contentDocument...`), o consulte la propia estructura de marcado/documentación de la aplicación destino si es de un origen diferente.

### 5k. Iframes anidados — `iframe_chemin` (v1.18.0)

Un iframe dentro de otro iframe: reemplazar `iframe_selecteur` con `iframe_chemin`, un arreglo ordenado: un selector CSS por cada nivel de anidamiento, desde el más externo hasta el más interno.

```
{"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` (marco único) y `iframe_chemin` (descenso anidado) son mutuamente excluyentes; se requiere exactamente uno por acción. Para un iframe de un solo nivel, continúe utilizando `iframe_selecteur` (sección 5j).

---

## 6. Acciones — referencia completa

| Type | Required params | Optional params | Notes |
|---|---|---|---|
| `naviguer` | `url` | — | Recarga completa de HTTP. Contabilizado en `respect.pages_visitees` |
| `cliquer` | `selecteur` | `force` (bool), `repli_js` (bool) | `force: true` omite elementos ocultos por CSS o muestra un modal. `repli_js: true` reintenta a través de JS si el clic nativo aún falla (v1.22.0) — requiere que `--no-evaluer` esté desactivado |
| `cliquer_som` | `id` | — | Clic en las coordenadas centrales del elemento. No se necesita `force` |
| `cliquer_visuel` | `description` | — | Visión de LLM (~32 s). Último recurso para canvas o elementos sin atributos |
| `remplir` | `selecteur`, `valeur` | `secret_cle` | `valeur: "depuis_secrets"` resuelve la credencial almacenada |
| `remplir_som` | `id`, `valeur` | `secret_cle` | Limpia el campo antes de escribir. `valeur: "depuis_secrets_totp"` para TOTP |
| `capturer` | `nom` | `som` (bool) | PNG intermedio con nombre. `som: true` para una captura anotada |
| `evaluer` | `script` | `attendu`, `contient`, `motif` | JS ejecutado en el navegador. Asertos solo para rpa.py |
| `defiler` | `px` or `selecteur` | — | Desplazamiento vertical en píxeles (`px`) o desplazamiento al elemento (`selecteur`) |
| `pause` | `ms` | `interval_capture` | Retraso fijo en ms. Prefiera `attendre_selecteur_present` para señales de DOM |

| Type | Required params | Optional params | Notes |
|---|---|---|---|
| `attendre` | `selecteur` | `interval_capture` | Espera a que un selector CSS esté presente |
| `attendre_navigation` | — | — | Espera a que finalicen las solicitudes de red (`networkidle`) |
| `attendre_url` | `motif` | `attendre_changement` (bool) | La URL contiene un patrón (coincidencia parcial). `attendre_changement: true` si la URL actual ya contiene el patrón |
| `attendre_selecteur_present` | `selecteur` | — | Espera a que el elemento sea visible (state=visible) |
| `attendre_absence` | `selecteur` | `delai_initial_ms` | Espera a que se elimine el elemento del DOM (state=detached) |
| `attendre_reseau_calme` | — | `timeout_ms` | 500 ms de silencio de red. `timeout_ms`: duración máxima antes de desistir |
| `attendre_mfa_ntfy` | `id_som` | `timeout` | Espera un código TOTP a través de ntfy, lo completa en el campo SoM |
| `nettoyer_overlay` | `selecteur` | — | Oculta las superposiciones que bloquean (banner de cookies, modal). Usar antes de SoM |
| `declencher_scenario` | `scenario` | — | Incluye las acciones de un sub-escenario. Profundidad máxima: 5 |
| `cliquer_iframe` | `iframe_selecteur` \| `iframe_chemin`, `selecteur` | `force` (bool) | Clic dentro de un iframe (v1.17.0). `iframe_chemin` para iframes anidados (v1.18.0, sección 5k). No se permite SoM dentro de frames |
| `remplir_iframe` | `iframe_selecteur` \| `iframe_chemin`, `selecteur`, `valeur` | `secret_cle` | Rellena dentro de un iframe (v1.17.0). `iframe_chemin` para iframes anidados (v1.18.0). `valeur: "depuis_secrets"` soportado |

---

## 7. Manejar obstáculos comunes

### 7a. Banner de cookies / superposición de bloqueo

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

Coloque **antes** de cualquier otra acción y antes de los números de SoM. La superposición oculta los elementos que tienen números de SoM. No la utilice en escenarios `watch.py`. (La superposición forma parte de la referencia visual).

### 7b. Elemento fuera de la ventana visible

SoM advierte cuando un elemento interactivo está fuera de la pantalla:

```
"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. Componentes web: Shadow DOM

Si los elementos interactivos visibles no reciben un número de SoM:

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

O en el escenario: `"shadow_dom": true` en la raíz.

Cuándo usar: Angular, Lit, Stencil, FAST. No activar en proyectos que no utilicen Web Components.

Para acceder a un elemento dentro de un Shadow Root sin `--shadow-dom`:

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

### 7d. Aplicaciones web de una sola página (SPA) (React, Vue, Angular) — navegación sin recarga

Después de un clic que cambia la vista en una aplicación SPA, Playwright no sabe cuándo se ha completado la navegación.

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

O espere a que aparezca un elemento específico de la nueva vista:

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

Nunca asumas que un clic ha completado la navegación sin una señal del DOM.

### 7e. Diálogo de CSS o `showModal()`

`TimeoutError` en `cliquer` cuando el elemento es visible en el DOM = elemento oculto con CSS o dentro de un diálogo.

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

Si `force: true` es insuficiente (el elemento está ausente del DOM):

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

No utilices `force` en `cliquer_som`. Es innecesario, `cliquer_som` utiliza coordenadas y evita las comprobaciones de forma nativa.

### 7f. Operación prolongada (indicador de carga, trabajo por lotes)

No use `pause` para esperar una duración fija. Espere la señal del DOM:

```
{"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"}
```

Si la operación no proporciona ninguna señal de DOM, utilice `interval_capture` para observar el estado:

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

Las capturas intermedias aparecen en `stream_captures[]`.

### 7g. Límite alcanzado (v1.15.0)

Si `respect.plafond_atteint` aparece en la salida, la ejecución se detuvo antes de terminar el escenario. Las acciones restantes no se ejecutaron.

Opciones: 1. Aumentar `max_pages_par_run` o `max_actions_par_run` en `diwall.conf` 2. Dividir el escenario en múltiples ejecuciones 3. Anular los límites máximos en el archivo JSON del escenario (esto se documentará en _CADRE).

### 7h. `<select>` campo de formulario

`remplir` no funciona en `<select>`. Utilice `remplir_som` con el ID de SoM de la `<select>`.

### 7i. Identificadores de SoM inválidos en la siguiente ejecución

Los ID de SoM se recalculan en cada captura. No persisten entre ejecuciones. Siempre vuelva a ejecutar `shot.py --som` para obtener los ID de la ejecución actual. Después de un `defiler` o al abrir una ventana emergente: vuelva a ejecutar `shot.py --som`.

### 7j. Desfase del ID de SoM en páginas altamente dinámicas: solución híbrida (v1.24.0)

`cliquer_som`/`remplir_som` resuelven `id: N` reindexando el DOM activo en el momento del clic; si un elemento interactivo aparece o desaparece **antes** de su objetivo en el orden del DOM entre el evento de captura `--som` y el clic (por ejemplo, un banner de cookies que se cierra, un modal que se abre), un identificador de elemento sin formato `id: N` puede resolver silenciosamente a un elemento **diferente** al que se muestra con el número N en la captura de pantalla.

Desde la versión v1.24.0, el resolvedor predeterminado es híbrido. En una página, llámalo:

- busca el marcador `data-dw-som-id="N"` (**estable**, definido por una acción `{"type":"capturer","som":true}` o la captura final);
- calcula el elemento N-ésimo mediante reindexación directa (**directa**);
- devuelve la ruta estable si el marcador existe, y la ruta directa en caso contrario (**alternativa**; idéntico al comportamiento previo a la versión 1.24.0);
- informa de `boussole.respect.som_resolution`, (`stable` | `brut` | `brut_sans_reference`) y, cuando las dos rutas no coinciden, `boussole.respect.som_derive_detectee` con el ID de SoM.

```
/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}]'
```

El camino estable solo ayuda dentro de un escenario: el marcador no persiste a través de dos llamadas a `shot.py` (cada una recarga la página). Para un objetivo propenso a mutaciones, capture el SoM (State of Mind) en el mismo escenario antes de la primera llamada a `cliquer_som`. Cuando `som_resolution` es `brut_sans_reference`, no se puede detectar la deriva; vuelva a capturar el SoM si el DOM cambió. `--som-brut` fuerza una reindexación pura y directa (sin búsqueda de marcador, sin detección de divergencia); luego, use `boussole.som_brut_actif: true`. `--som-rafraichir` (v1.17.0) es un alias inactivo que se mantiene para la compatibilidad hacia atrás.

Desde la versión v1.17.2, el inyector elimina los marcadores dejados por una captura anterior `--som` en la misma página antes de volver a numerar; sin esto, un elemento oculto o desplazado entre dos capturas podría conservar un identificador obsoleto `data-dw-som-id`, lo que provocaría una colisión con un elemento recién numerado y daría como resultado el identificador incorrecto.

### 7k. Sitio bloqueado por el WAF (error 403 inmediato)

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

Si el error 403 persiste con `--stealth`: el sitio utiliza huellas digitales TLS (JA3/JA4) o análisis de comportamiento avanzado (Cloudflare Enterprise). `playwright-stealth` no evita estas protecciones. Consulte `docs/RETOUR_EXPERIENCE.md` FR-77/FR-78/FR-79 para obtener más información.

Diwall también indica de forma pasiva una posible obstrucción sin que usted tenga que verificar el estado HTTP directamente; consulte la sección 3e (`respect.waf_bloquants`).

#### 7l. La navegación inicial nunca se completa — `--wait-until` (v1.22.0)

Síntoma: `TimeoutError` en la navegación inicial, y aumentar `--timeout` no cambia nada (45 s falla exactamente como 10 s). Causa: por defecto Diwall espera a `networkidle` — 500 ms de silencio de red. Una página que realiza consultas continuamente (estadísticas en tiempo real, contadores de actualización automática, paneles de administración del router) nunca produce ese silencio, así que ningún valor de tiempo de espera puede ser lo suficientemente grande.

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

# rpa.py — se propagó a shot.py, por lo que los escenarios alcanzan los mismos objetivos.
/opt/diwall/venv/bin/python3 /opt/diwall/rpa.py \
  --scenario ./admin_login.json --wait-until load --guide-version 1.3
```

Un escenario puede incluirlo como una propiedad raíz, permaneciendo así autocontenido:

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

La opción de línea de comandos tiene prioridad sobre la propiedad del escenario.

| Valor | Espera por | Úselo cuando |
|---|---|---|
| `networkidle` | 500 ms de silencio en la red | predeterminado; manténgalo a menos que falle |
| `load` | evento `load` (página y subrecursos) | sondeo continuo / estadísticas en tiempo real |
| `domcontentloaded` | HTML analizado, los subrecursos aún están pendientes | página muy pesada, solo necesita el DOM |

Se aplica solo a la navegación inicial; la acción `naviguer` no se ve afectada. `boussole.wait_until` informa el valor solo cuando es diferente del valor predeterminado.

---

## 8. Monitoreo visual — watch.py

### 8a. Guardar una referencia

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

La referencia se guarda en `/opt/diwall/references/`.

### 8b. Comparar con la referencia (diferencia de píxeles)

```
/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
```

Veredictos:

| taux_diff | Veredicto | Código de salida |
|---|---|---|
| < 0.2% | stable | 0 |
| 0.2% – 5% | drift | 0 |
| ≥ 5% | regression | 1 |
| Dimensiones diferentes | viewport_mismatch | 2 |

### 8c. Comparación semántica (modelo de lenguaje grande)

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

Combina el análisis de diferencias de píxeles con el análisis de modelos de lenguaje grandes (LLM):

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

`--llm claude` (v1.24.2) envía las dos capturas — la de referencia y la actual — a la API de Anthropic en lugar del modelo local; se aplica tanto a `--comparer` como a `--llm-en-complement`. Las capturas salen de la máquina, reducidas para que su lado más largo no supere los 1568 px, el tamaño con el que trabaja la API. Necesita el módulo `anthropic`, ausente de una instalación estándar, y una clave en el entorno del proceso:

```
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
```

Un módulo ausente, una clave rechazada o un rechazo del modelo se devuelven como un mensaje en la salida (`erreur`, o `analyse_llm` en modo píxel), nunca como un traceback.

### 8d. Ignorar una zona animada

```
/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. Bucle de monitoreo

```
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 para el monitoreo autónomo

```
# /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. Monitoreo estructural continuo — `monitor-verifier.sh` (v1.18.0)

Complementos 8a–8f: `watch.py` monitorea la *apariencia* (píxeles/semántica). `scripts/monitor-verifier.sh` monitorea la *estructura* (`http_status`, `dom_stats`, `evaluations`, conteo de SoM) — imagen nula, llamada LLM nula, construido sobre `--no-capture` + `--replay-verifier` (sección 5h).

```
# Primera ejecución: crear la referencia estructural.
/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

# Una verificación y alerta que se ejecuta periódicamente (no es un demonio), ejecútela
 ↪  repetidamente mediante cron.
# Los archivos `scripts/*.sh` nunca se despliegan a /opt/diwall/, por lo que se ejecutan
 ↪  desde el repositorio de Git.
# fuente, como si fuera su propio usuario.
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 (su propio archivo 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
```

Estable ⟶ silencio. Regresión ⟶ una notificación `ntfy` con las diferencias. Cada ejecución es un proceso aislado: sin demonio, sin riesgo de fuga de memoria, y la función ”Navegación Respetuosa” restablece los límites limpiamente en cada ejecución.

---

## 9. Registro de operaciones

El registro es configurable en `diwall.conf` (v1.15.0):

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

Si está ausente o el directorio cifrado no está montado, alternativa: variable de entorno `DIWALL_JOURNAL`, luego `/var/log/diwall/operations.jsonl`.

```
# Lee las últimas 10 entradas.
tail -n 10 ~/Vaults/__PROJET__/Diwall/operations.jsonl | python3 -m json.tool

# Filtra por objetivo (herramienta journal.py).
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com

# Filtrar solo las operaciones que modifican los datos.
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com --mutatif

# Desde una fecha.
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
  --cible app.example.com --depuis 2026-07-01

# Ejecuciones fallidas únicamente (v1.20.0) — resultado != "éxito"
/opt/diwall/venv/bin/python3 /opt/diwall/journal.py \
```

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

Campos en cada entrada:

| Campo | Significado |
|---|---|
| `ts` | Marca de tiempo ISO 8601 |
| `version` | Versión de Diwall |
| `outil` | `shot.py` o `rpa.py` |
| `cible_url` | URL de destino |
| `scenario` | Ruta del archivo de escenario (modo RPA) |
| `source_scenario` | Solo el nombre del archivo de escenario, sin ruta (v1.18.0) — activa `mode_conseille` (sección 2e) |
| `resultat` | `"succes"` o `"echec"` |
| `mutatif` | `true` si hay al menos una acción de escritura |
| `duree_ms` | Duración en ms |
| `intention` | Etiqueta pasada a través de `--intention` o el campo de escenario `intention` |

### 9a. Rotación de registros (G-36, CHANTIER_SANITISATION.md)

Diwall no incluye una configuración de logrotate — `/var/log/diwall/operations.jsonl` crece sin límite hasta que el administrador instala una. `lib/journal.py` abre y cierra el archivo en cada escritura (sin un descriptor de archivo persistente entre ejecuciones), específicamente para que el comportamiento **predeterminado** de logrotate (renombrar el archivo actual, crear uno nuevo) funcione correctamente sin ninguna opción especial: la siguiente escritura vuelve a abrir la ruta y encuentra el nuevo inode.

No agregue `copytruncate` a la configuración de logrotate de Diwall; es innecesario aquí (a diferencia de las herramientas que mantienen un descriptor de archivo abierto durante toda su vida útil) y reintroduce una ventana de pérdida de escritura que este diseño se diseñó para evitar. Ejemplo: `/etc/logrotate.d/diwall`

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

`journal.py` (el lector) ya sigue los archivos rotados de forma transparente. (`operations.jsonl`, `.1`, `.2.gz`, ...) — no se necesita ningún paso adicional después de la rotación.

---

## 10. Flags de la línea de comandos: referencia

### shot.py

| Flag | Default | Descripción |
|---|---|---|
| `--version` | — | Imprime la versión instalada y sale inmediatamente — no se requiere Playwright ni ningún otro argumento (v1.18.0) |

| Flag | Default | Descripción |
|---|---|---|
| `--guide-version X.Y` | — | Prueba de lectura de `docs/GUIDE_LLM.md` — requerido a menos que ya exista un marcador local válido (v1.18.0, sección 1) |
| `--url URL` | requerido | URL a capturar |
| `--actions FILE` | — | Archivo JSON de acciones secuenciales |
| `--output-dir DIR` | `/tmp/diwall` | Directorio de salida PNG |
| `--timeout MS` | 10000 | Tiempo de espera de Playwright por acción (ms) |
| `--screenshot-timeout MS` | 120000 | Tiempo de espera para `page.screenshot()` (ms). Distinto de `--timeout` |
| `--largeur PX` | 1280 | Ancho del viewport |
| `--hauteur PX` | 720 | Altura del viewport |
| `--som` | off | Activa el "Set-of-Mark" (numeración de elementos) |
| `--a11y` | off | Incluye el árbol de accesibilidad en el JSON |
| `--shadow-dom` | off | Recorre Shadow Roots para SoM (Angular, Lit, Stencil) |
| `--stealth` | off | Modo sigiloso de playwright-stealth (v1.15.0) |
| `--mode fast\|full` | — | `fast` = `--no-capture --a11y`. `full` = comportamiento predeterminado |
| `--no-capture` | off | Omite la captura de PNG y el SoM |
| `--llm local\|claude` | `local` | Motor LLM para `cliquer_visuel` |
| `--secrets FILE` | — | Ruta explícita a un archivo de credenciales |
| `--auth-indicator SEL` | — | Selector CSS presente solo en la sesión autenticada |
| `--auth-indicator-negative SEL` | — | Selector CSS presente solo fuera de la sesión autenticada |
| `--intention TEXT` | — | Etiqueta comercial registrada en el registro |
| `--sauver-session FILE` | — | Guarda las cookies después de las acciones |
| `--reprendre-session FILE` | — | Reanuda una sesión guardada |
| `--interval-capture N` | 0 | Capturas periódicas cada N segundos durante `attendre`, `pause` |
| `--som-brut` | off | Fuerza la reindexación pura de SoM; desactiva la ruta estable y la detección de divergencia del resolvedor híbrido (v1.24.0, sección 7j) |
| `--som-rafraichir` | off | Alias de no-op desde v1.24.0 — la resolución híbrida es el valor predeterminado (v1.17.0, sección 7j) |
| `--ignorer-waf` | off | Un bloqueo de WAF detectado degrada `niveau_confiance` pero ya no fuerza `pret_a_agir: false` por sí solo (v1.17.2, sección 3e) |
| `--http-credentials` | off | Resuelve las credenciales de HTTP Basic Auth del archivo de credenciales, con alcance al origen del objetivo (v1.21.0, sección 4g) |

| Flag | Default | Descripción |
|---|---|---|
| `--no-evaluer` | off | Rechaza la acción **evaluer** para toda la ejecución — recomendado en producción para objetivos con formularios confidenciales (v1.15.1) |
| `--no-filtre-evaluer` | off | Deshabilita la neutralización de stdout de los valores de retorno, las URL y los mensajes de error de **evaluer** — solo ejecuciones de depuración explícitas. La neutralización está activada de forma predeterminada; cuando está deshabilitada, `boussole.filtre_evaluer_actif: false` se establece en la salida para que el operador pueda auditarla desde el propio JSON (v1.23.0) |

**rpa.py**

Propaga todas las banderas relevantes de shot.py, además de:

| Flag | Description |
|---|---|
| `--version` | Imprime la versión instalada y sale inmediatamente (v1.18.0) |
| `--guide-version X.Y` | Prueba de lectura de `docs/GUIDE_LLM.md` — verificada de forma independiente, misma regla que shot.py (v1.18.0) |
| `--scenario FILE` | Ruta al escenario JSON o YAML (requerido) |
| `--url URL` | Sobreescribe la URL del escenario sin modificar el archivo |
| `--stealth` | Se propaga a shot.py |
| `--mode fast\|full` | Se propaga a shot.py |
| `--som-brut` | Se propaga a shot.py. También se puede establecer como propiedad raíz del escenario `"som_brut": true` (v1.24.0, sección 7j) |
| `--som-rafraichir` | Se propaga a shot.py — alias sin efecto desde v1.24.0 (v1.17.0, sección 7j) |
| `--ignorer-waf` | Se propaga a shot.py (v1.17.2, sección 3e) |
| `--http-credentials` | Se propaga a shot.py. También se puede establecer como propiedad raíz del escenario `"http_credentials": true` (v1.21.0, sección 4g) |
| `--sauver-verifier-reference FILE` | Guarda la referencia estructural para `--replay-verifier` (v1.17.0, sección 5h) |
| `--replay-verifier FILE` | Compara la ejecución con una referencia estructural, sale con código 1 en caso de regresión (v1.17.0, sección 5h) |
| `--checkpoint FILE` | Reanuda un escenario largo después de un fallo durante la ejecución (v1.17.0, sección 5i) |

**watch.py**

| Flag | Description |
|---|---|
| `--version` | Imprime la versión instalada y sale inmediatamente (v1.18.0) |

| Flag | Description |
|---|---|
| `--guide-version X.Y` | Prueba de lectura de `docs/GUIDE_LLM.md` — verificada de forma independiente, misma regla que shot.py (v1.18.0) |
| `--url URL` | URL a monitorizar |
| `--sauver-reference` | Captura y guarda como referencia |
| `--comparer-pixel REF` | Diferencia de píxeles con respecto al archivo PNG REF |
| `--comparer` | Diferencia semántica del LLM |
| `--llm local\|claude` | Motor para `--comparer` y `--llm-en-complement` (por defecto `local`; `claude` envía las capturas a la API de Anthropic, v1.24.2) |
| `--nom NAME` | Nombre de la vista (múltiples vistas por URL) |
| `--seuil-stable F` | Umbral de `stable` (por defecto: 0.002 = 0.2%) |
| `--seuil-regression F` | Umbral de `regression` (por defecto: 0.05 = 5%) |
| `--exclure-zone X,Y,W,H` | Zona a ignorar (repetible) |
| `--heatmap` | Produce una imagen PNG de las zonas modificadas |
| `--ntfy-url URL` | Envía una alerta ntfy en caso de regresión |
| `--llm-en-complement` | Añade la diferencia del LLM cuando el píxel es una desviación o una regresión |

---

## 11. Códigos de salida y resultados

### Códigos de salida

| Código | Causa | Qué hacer |
|---|---|---|
| 0 | Éxito | — |
| 1 | Error de Playwright, acción fallida, aserción de rpa.py | Leer `erreur` en el JSON. Ver `GUIDE_LLM_INTERACTIONS.md` |
| 1 | `guide_non_lu` — `--guide-version` ausente o incorrecto, sin marcador válido (v1.18.0) | Se dispara antes de arrancar Playwright. Leer `docs/GUIDE_LLM.md` y relanzar con `--guide-version X.Y` (sección 1) |
| 2 | `viewport_mismatch` (watch.py) | Volver a capturar la referencia con el mismo viewport |
| 3 | Módulo `playwright` no encontrado | Invocar mediante `/opt/diwall/venv/bin/python3` |
| 42 | `SecretsFermesError` — directorio cifrado no montado, o suma de control inválida | Montarlo, o verificar el archivo de credenciales |
| 43 | `SecretsNonConfigureError` — falta `diwall.conf` | `sudo cp /opt/diwall/diwall-sample.conf /opt/diwall/diwall.conf && sudo nano /opt/diwall/diwall.conf` |

### Estructura del JSON de salida

```
{
  "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) siempre está presente e identifica esta ejecución de forma única: indica el directorio de aislamiento que se encuentra en `/tmp/diwall/<operation_id>/` y coincide con el campo `operation_id` de la entrada de esta ejecución en el registro de operaciones (sección 9). `etat` (v1.16.0) solo está presente en la ruta de éxito. `latences_actions` (v1.20.0) siempre está presente (lista vacía si no hay acciones), una entrada por cada acción que se ejecutó realmente; consulta `GUIDE_LLM_MONITORING.md` para ver cómo complementa a `respect.duree_totale_ms`.

Claves condicionales (ausentes cuando está inactivo): `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` (presente siempre que se haya resuelto una acción de SoM), `respect.som_derive_detectee` (presente solo en una divergencia estable/cruda), `respect.indice_agressivite` (presente siempre que se haya ejecutado al menos una acción), `actions_executees_avant_echec`, `pages_visitees_avant_echec` (solo en el JSON de error, v1.17.0), `etat.mode_conseille` (presente solo con datos previos reales `diagnostic_dom.json` para este host, v1.18.0, sección 2e).

### Error — formato

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

---

## Rutas de referencia

| Path | Role |
|---|---|
| `/opt/diwall/` | Instalación de producción |
| `/opt/diwall/venv/bin/python3` | Python a utilizar en cada ejecución |
| `/opt/diwall/diwall.conf` | Configuración de la máquina (credenciales, navegación, registro) |
| `/opt/diwall/diwall-sample.conf` | Plantilla de configuración |
| `/opt/diwall/scenarios/` | Escenarios de RPA |
| `/opt/diwall/docs/` | Documentación |
| `/opt/diwall/references/` | Referencias visuales watch.py |
| `/tmp/diwall/<operation_id>/` | Capturas temporales para una ejecución, aisladas por `operation_id` (v1.16.0, se borran al reiniciar) |
| `~/Vaults/__PROJET__/Diwall/` | Credenciales + registro (volumen gocryptfs) |
| `~/git/Diwall/Diwall/` | Fuentes de Git (modificar aquí, luego `deploy.sh`) |

Desplegar después de modificar las fuentes:

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