Test automatisés

SIGACS – Documentation du script de démo Selenium

Script de démonstration automatisée de l'interface web SIGACS.
Pilote Firefox en mode graphique, remplit les formulaires caractère par caractère, soumet les données en base, puis marque une pause opérateur après chaque enregistrement.

Prérequis

Outil	Version minimale
Python	3.10+
Firefox	toute version récente
geckodriver	compatible avec Firefox installé
selenium	4.x

Installation rapide :

```
pip install selenium
# geckodriver doit être dans le PATH
```

Lancer le script

```
python demo_sigacs.py
```

Firefox s'ouvre automatiquement, maximisé.
Chaque étape s'affiche dans le terminal.
Après chaque écriture en base, le script se fige jusqu'à ce que l'opérateur appuie sur **Entrée**.

Configuration

Tous les réglages sont regroupés en haut du fichier dans le bloc `CONFIG`.

```
BASE_URL      = "http://192.168.42.131"   # adresse du serveur web

USERNAME      = "demo_user"               # compte existant en base
PASSWORD      = "demo1234"

CHAR_DELAY    = 0.07   # secondes entre chaque frappe clavier
FIELD_PAUSE   = 1.0    # pause après avoir rempli un champ
ACTION_PAUSE  = 1.5    # pause après un clic
SECTION_PAUSE = 2.5    # pause entre deux étapes majeures
PAGE_WAIT     = 8      # délai max d'attente d'un élément (WebDriverWait)
```

Modifier un seul nombre change le rythme de toute la démo.

Scénario couvert

Étape	Fonction	Censé échouer
1	`test_login`	oui
2	`test_admin_login`	non
1	`creation_demo_user`	non
1	`test_login`	non
2	`test_new_serre`	non
3	`test_new_bac`	non
4	`test_new_culture`	non
5	`test_logout`	non

Pause opérateur

Après chaque soumission validée, le terminal affiche :

```
?  New Serre record written – check the DB / UI
   ? Inspect the result, then press [Enter] to continue...
```

Le navigateur reste ouvert et interactif pendant cette pause.
Appuyer sur **Entrée** reprend le scénario.

Gérer un test censé échouer

Utiliser la fonction utilitaire `attempt()` dans `run_demo()` au lieu d'appeler la fonction directement.

```
def attempt(fn, *args, expect_fail=False, **kwargs):
    try:
        fn(*args, **kwargs)
        if expect_fail:
            print(f"    ? {fn.__name__} était censé échouer mais a réussi")
    except Exception as e:
        if expect_fail:
            print(f"    ? {fn.__name__} a échoué comme prévu : {e}")
        else:
            raise  # échec inattendu ? stoppe la démo
```

Exemple d'utilisation dans `run_demo()` :

```
def run_demo() -> None:
    driver = make_driver()
    try:
        attempt(test_login,       driver, expect_fail=True)  # doit échouer
        attempt(test_admin_login, driver)                    # crée le compte
        attempt(test_login,       driver)                    # doit réussir
        ...
```

`expect_fail=True` → l'exception est capturée et affichée comme un succès attendu, le scénario continue.
Sans le flag → toute exception remonte et stoppe la démo normalement.

Fonctions utilitaires

Fonction	Rôle
`slow_type(element, text)`	Tape le texte caractère par caractère avec `CHAR_DELAY`
`pause(duration)`	Attend `duration` secondes (défaut : `ACTION_PAUSE`)
`wait_resume(label)`	Affiche le label et attend `[Entrée]`
`wait_for(driver, by, value)`	Attend qu'un élément soit présent dans le DOM
`wait_clickable(driver, by, value)`	Attend qu'un élément soit cliquable
`wait_invisible(driver, by, value)`	Attend qu'un élément disparaisse
`make_driver()`	Crée et maximise un driver Firefox headful
`attempt(fn, *args, expect_fail)`	Exécute un test en capturant les échecs attendus

Revision #3
Created 2026-05-22 09:03:41 UTC by Clément
Updated 2026-05-22 09:12:50 UTC by Clément