# Guide de démarrage pour les tests de sécurité avec OWASP ZAP

Les tests de sécurité sont essentiels pour identifier et corriger les vulnérabilités dans les applications web. OWASP ZAP (Zed Attack Proxy) est un outil open-source qui facilite la détection de ces failles. Ce guide vous accompagnera dans l'installation, la configuration et l'utilisation d'OWASP ZAP pour réaliser des tests de sécurité.

## Qu’est-ce que OWASP ZAP ?

OWASP ZAP est un outil de test de sécurité conçu pour aider les développeurs et les testeurs à trouver des vulnérabilités dans leurs applications web. C'est l'un des projets phares de l'OWASP (Open Web Application Security Project) et il est largement utilisé par les professionnels de la sécurité.

**Principales fonctionnalités :**

* Proxy d'analyse des requêtes HTTP.
* Scans automatisés et manuels.
* Plug-ins pour étendre les fonctionnalités.
* Interface utilisateur conviviale.

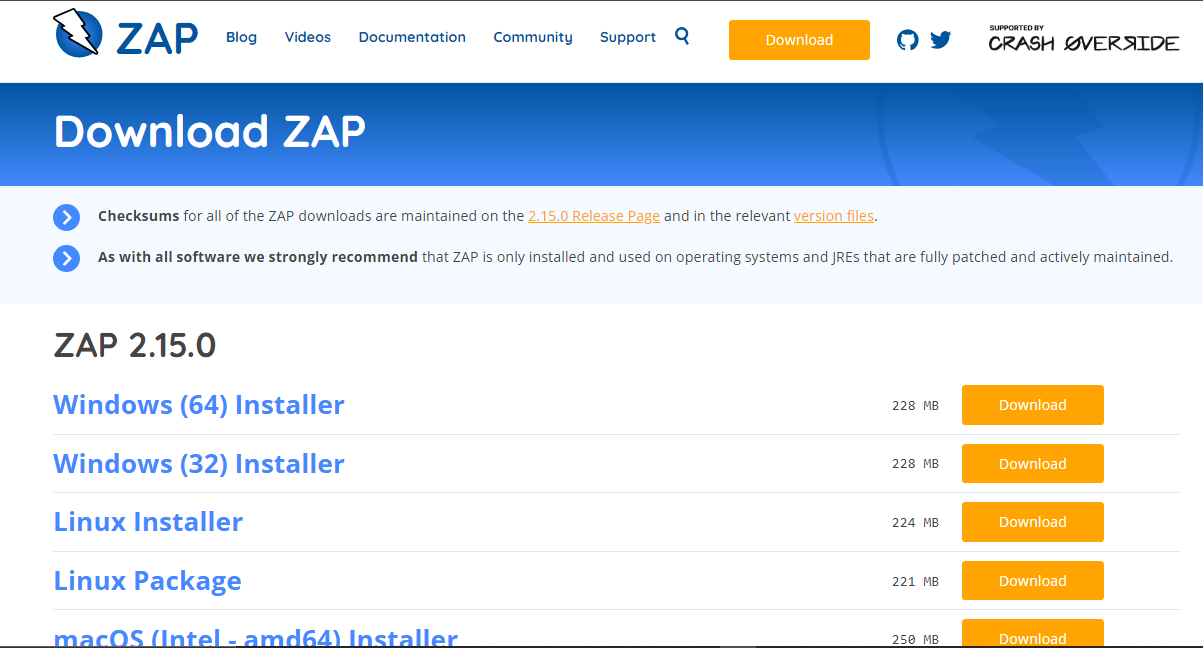
## Installation de OWASP ZAP

Prérequis :

* Java 8 ou plus récent.
* Environ 200 Mo d'espace disque.

Étapes d'installation :

1. **Téléchargement** : Rendez-vous sur le site officiel d'OWASP ZAP et téléchargez la version appropriée pour votre système d'exploitation, après avoir installé au préalable JDK de préférence la dernière version stable.

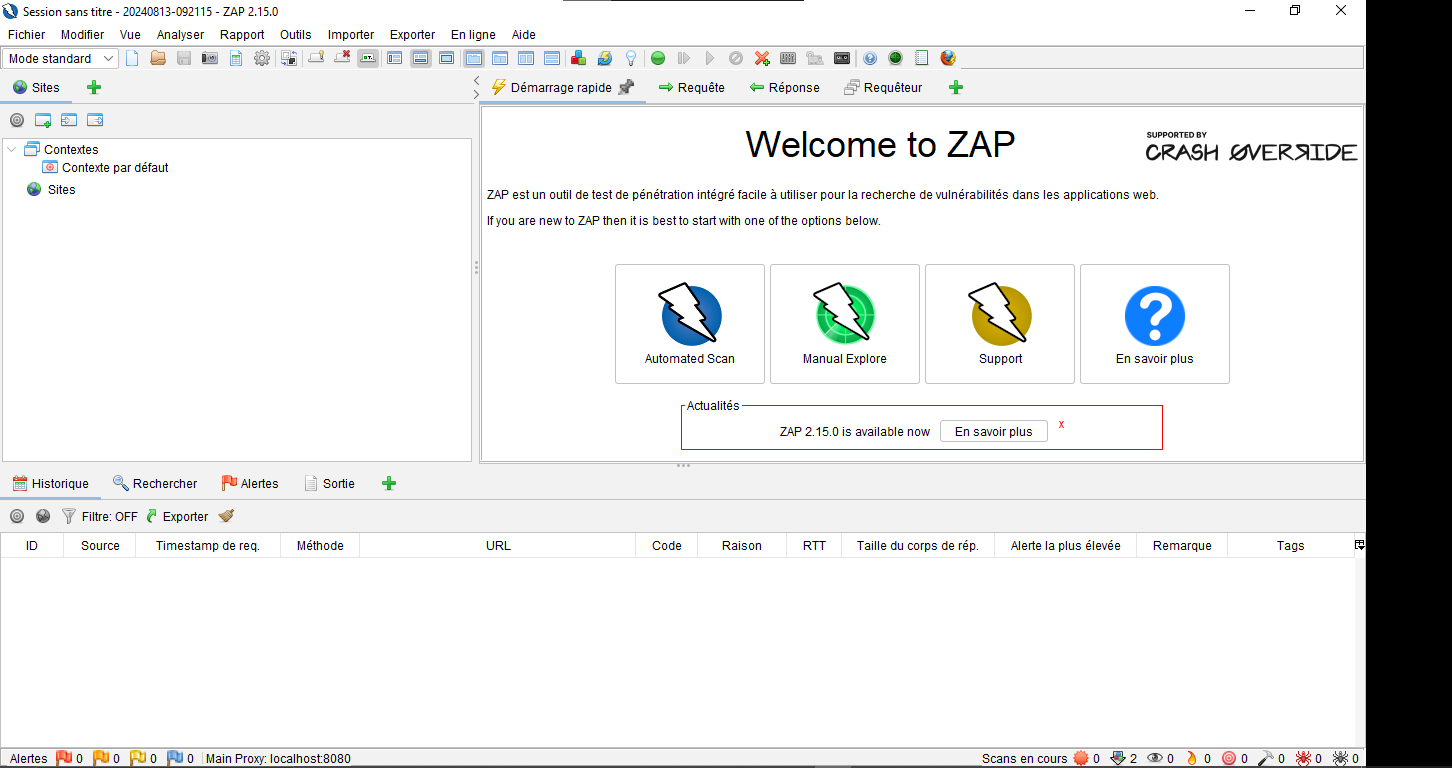


Voici le lien pour le télécharger : <https://www.zaproxy.org/download/>

1. **Installation** :

* Windows : Exécutez le fichier .exe téléchargé et suivez les instructions.
* macOS : Téléchargez le fichier .dmg, puis glissez OWASP ZAP dans le dossier Applications.
* Linux : Extrayez le fichier .tar.gz et exécutez zap.sh.

1. **Lancer OWASP ZAP** : Une fois installé, lancez OWASP ZAP et choisissez de configurer une nouvelle session ou de charger une session existante



## Configuration d'OWASP ZAP

* Configuration initiale :
  + Proxy : Par défaut, OWASP ZAP fonctionne comme un proxy HTTP sur le port 8080. Vous pouvez configurer votre navigateur pour utiliser ce proxy et capturer les requêtes HTTP.
  + Scan Policy : Vous pouvez ajuster les politiques de scan en fonction des types de tests que vous souhaitez réaliser, comme les tests d'injection SQL ou les tests XSS.
* Paramètres recommandés :
  + Active Scan : Sélectionnez les types de tests que vous souhaitez exécuter.
  + Session Management : Configurez la gestion des sessions pour maintenir les connexions actives durant les scans.

## Utilisation d'OWASP ZAP

Démarrer un scan :

* Passive Scan : Lancez un scan passif pour analyser le trafic sans envoyer de requêtes supplémentaires.
* Active Scan : Démarrez un scan actif pour détecter les vulnérabilités en envoyant des requêtes spécifiques à votre application.

Interprétation des résultats :

* Alertes : Les résultats sont classés en alertes selon leur gravité (bas, moyen, élevé).
* Rapports : OWASP ZAP génère des rapports détaillés que vous pouvez exporter en divers formats (HTML, XML, JSON).

## Avantages de OWASP ZAP

Voici les principaux avantages de l'utilisation de OWASP ZAP (Zed Attack Proxy) pour les tests de sécurité des applications web :

1. Open-source et Gratuit

- Accessibilité : OWASP ZAP est un outil open-source, ce qui signifie qu'il est gratuit à utiliser pour les entreprises de toutes tailles, sans frais de licence.

- Communauté Active : L'outil bénéficie d'une grande communauté qui contribue à son développement, à l'ajout de nouvelles fonctionnalités, et au support des utilisateurs.

2. Facilité d'Utilisation

- Interface Conviviale : ZAP est conçu pour être accessible aux débutants tout en offrant des fonctionnalités avancées pour les experts. Son interface utilisateur est intuitive, facilitant la prise en main.

- Documentation Étendue : Il est accompagné d'une documentation complète et de tutoriels, ce qui facilite son adoption même pour les utilisateurs ayant peu d'expérience en sécurité.

3. Flexibilité et Extensibilité

- Plug-ins et Add-ons : ZAP dispose d'une bibliothèque de plug-ins et d'add-ons qui permettent d'étendre ses fonctionnalités selon les besoins spécifiques des utilisateurs.

- Personnalisation : Les utilisateurs peuvent personnaliser ZAP pour adapter les tests aux exigences spécifiques de leur application ou de leur environnement.

4. Scans Automatisés et Manuels

- Scan Actif et Passif : ZAP permet de réaliser à la fois des scans passifs (analyse sans envoyer de requêtes supplémentaires) et des scans actifs (envoi de requêtes pour tester les vulnérabilités), offrant ainsi une couverture complète des tests de sécurité.

- Automatisation : ZAP peut être intégré dans les pipelines CI/CD pour automatiser les tests de sécurité à chaque étape du développement, garantissant que les vulnérabilités sont détectées tôt.

5. Large Gamme de Vulnérabilités Ciblées

- Détection Complète : OWASP ZAP est capable de détecter un large éventail de vulnérabilités courantes, telles que les injections SQL, les failles XSS, les configurations de sécurité faibles, et bien d'autres.

- Mises à Jour Régulières : Grâce à sa communauté active, ZAP est régulièrement mis à jour pour inclure les dernières techniques de détection et les nouveaux types de vulnérabilités.

6. Intégration Facile

- Compatibilité : ZAP peut être facilement intégré avec d'autres outils de sécurité, de développement, et de gestion de projets.

- API REST : ZAP dispose d'une API REST puissante qui permet d'automatiser les tests de sécurité et de les intégrer dans divers processus de développement et de déploiement.

7. Rapports Détaillés

- Rapports Personnalisables : ZAP génère des rapports détaillés sur les vulnérabilités trouvées, classés par gravité, avec des recommandations pour corriger chaque problème.

- Formats Variés : Les rapports peuvent être exportés dans différents formats (HTML, XML, JSON), facilitant leur partage avec les équipes concernées.

Ces avantages font de OWASP ZAP un outil de choix pour les développeurs, les testeurs, et les professionnels de la sécurité cherchant à sécuriser leurs applications web de manière efficace et abordable.

## Inconvénients de OWASP ZAP

Bien que OWASP ZAP soit un outil puissant et populaire pour les tests de sécurité, il présente également quelques inconvénients. Voici certains des défis ou limitations associés à son utilisation :

1. Courbe d'Apprentissage

- Complexité pour les Débutants : Bien que l'interface soit conviviale, OWASP ZAP peut encore être complexe à utiliser pour ceux qui n'ont pas d'expérience préalable en sécurité. Comprendre et interpréter les résultats des tests peut nécessiter un certain niveau de compétence technique.

2. Performances et Scalabilité

- Lenteur sur les Grandes Applications : ZAP peut être lent lors de l'analyse de grandes applications avec un nombre élevé de pages ou de fonctionnalités. Les scans actifs, en particulier, peuvent prendre beaucoup de temps.

- Utilisation Intensive des Ressources : Lors des scans actifs, OWASP ZAP peut consommer beaucoup de ressources système (CPU, RAM), ce qui peut affecter les performances globales du système sur lequel il est exécuté.

3. Détection de Faux Positifs

- Résultats Faux Positifs : Comme beaucoup d'outils automatisés de sécurité, ZAP peut parfois signaler des vulnérabilités qui ne sont pas réellement exploitables (faux positifs). Cela peut nécessiter un examen manuel pour vérifier la validité des alertes.

4. Support et Assistance

- Support Communautaire : Le support technique pour OWASP ZAP repose principalement sur la communauté. Il n'y a pas de support professionnel direct ou de SLA garanti, ce qui peut être un inconvénient pour les entreprises qui ont besoin d'une assistance rapide en cas de problème.

5. Personnalisation Avancée

- Limites dans les Scénarios Complexes : Bien que ZAP soit flexible, la personnalisation avancée des tests pour des scénarios très spécifiques ou complexes peut être difficile et peut nécessiter des connaissances approfondies du fonctionnement de l'outil.

- Scripts Complexes : L'écriture de scripts personnalisés pour automatiser des tests complexes peut être difficile pour ceux qui ne sont pas familiers avec les langages de script supportés par ZAP.

6. Couverture des Tests

- Vulnérabilités Non Couvertes : Bien que ZAP détecte une large gamme de vulnérabilités, il ne peut pas couvrir tous les types de failles possibles, en particulier celles qui sont spécifiques à certaines configurations ou technologies très récentes.

- Limites des Tests Automatisés : Comme tout outil de test automatisé, ZAP peut ne pas détecter certaines vulnérabilités complexes ou logiques qui nécessitent une analyse humaine.

7. Mises à Jour et Maintenance

- Fréquence des Mises à Jour : Les mises à jour de ZAP dépendent de la communauté open-source, ce qui peut entraîner des périodes où les nouvelles vulnérabilités ou les changements technologiques ne sont pas immédiatement pris en charge.

- Compatibilité : Lors des mises à jour, certaines fonctionnalités ou add-ons peuvent devenir incompatibles, nécessitant des ajustements ou des attentes pour la mise à jour des add-ons concernés.

8. Intégration avec d'Autres Outils

- Compatibilité avec les Écosystèmes Propriétaires : ZAP est plus difficile à intégrer avec certains environnements propriétaires ou très spécifiques par rapport à d'autres solutions commerciales qui offrent des intégrations prêtes à l'emploi.

Ces inconvénients doivent être pris en compte lors de la sélection d'OWASP ZAP comme outil principal pour les tests de sécurité, surtout dans des environnements critiques ou à grande échelle où des solutions alternatives ou complémentaires peuvent être nécessaires.

# Guide de démarrage pour les tests end to end avec Playwright

Les tests end-to-end (E2E) permettent de vérifier le bon fonctionnement d'une application du point de vue de l'utilisateur. Playwright est un outil moderne, créé par Microsoft, qui permet d'automatiser les interactions utilisateur pour tester les applications web. Ce guide vous accompagnera dans l'installation, la configuration et l'utilisation de Playwright pour réaliser des tests E2E.

## Qu’est-ce que Playwright ?

Playwright est un framework open-source qui permet d'automatiser les interactions avec les navigateurs web. Il prend en charge plusieurs navigateurs (Chromium, Firefox, WebKit) et permet d'écrire des tests fiables et rapides.

**Principales fonctionnalités :**

* Prise en charge de plusieurs navigateurs.
* Exécution des tests en parallèle.
* Tests cross-platform (Windows, macOS, Linux).
* Capture de vidéos et de captures d'écran pendant les tests.

## Installation de Playwright

**Prérequis :**

* Node.js version 12 ou supérieure.
* Environ 100 Mo d'espace disque pour l'installation des navigateurs.

### Étapes d'installation via commande

1. **Initialisation du projet** : Créez un nouveau projet Node.js ou naviguez dans un projet existant.

* npm init -y

1. **Installation de Playwright** : Utilisez npm pour installer Playwright.

* npm install --save-dev @playwright/test

1. **Installation des navigateurs** : Playwright téléchargera automatiquement les navigateurs lors de l'installation. Vous pouvez les installer manuellement si besoin avec :

* npx playwright install

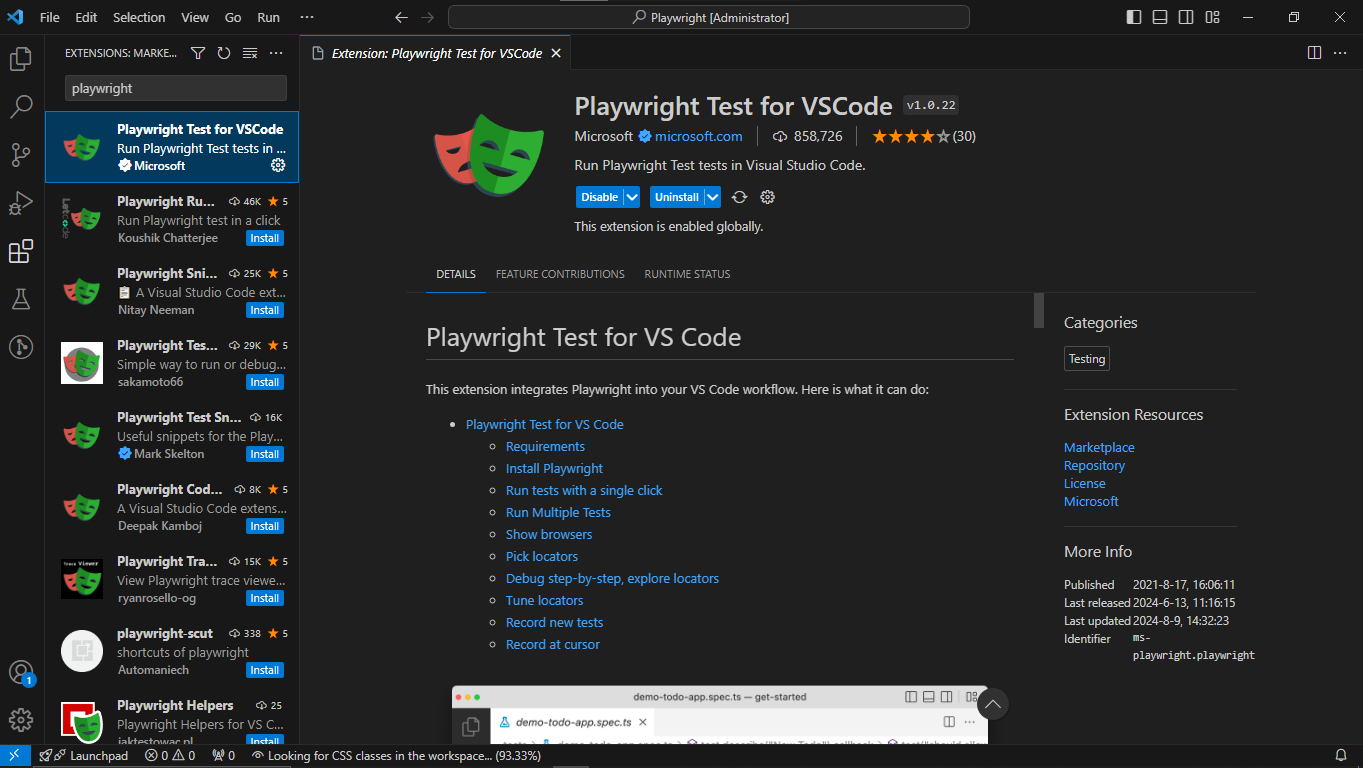
### Étapes d'installation via Vscode

1. Installer Vscode : Il faut tout d’abord télécharger, puis installer Visual Studio code.



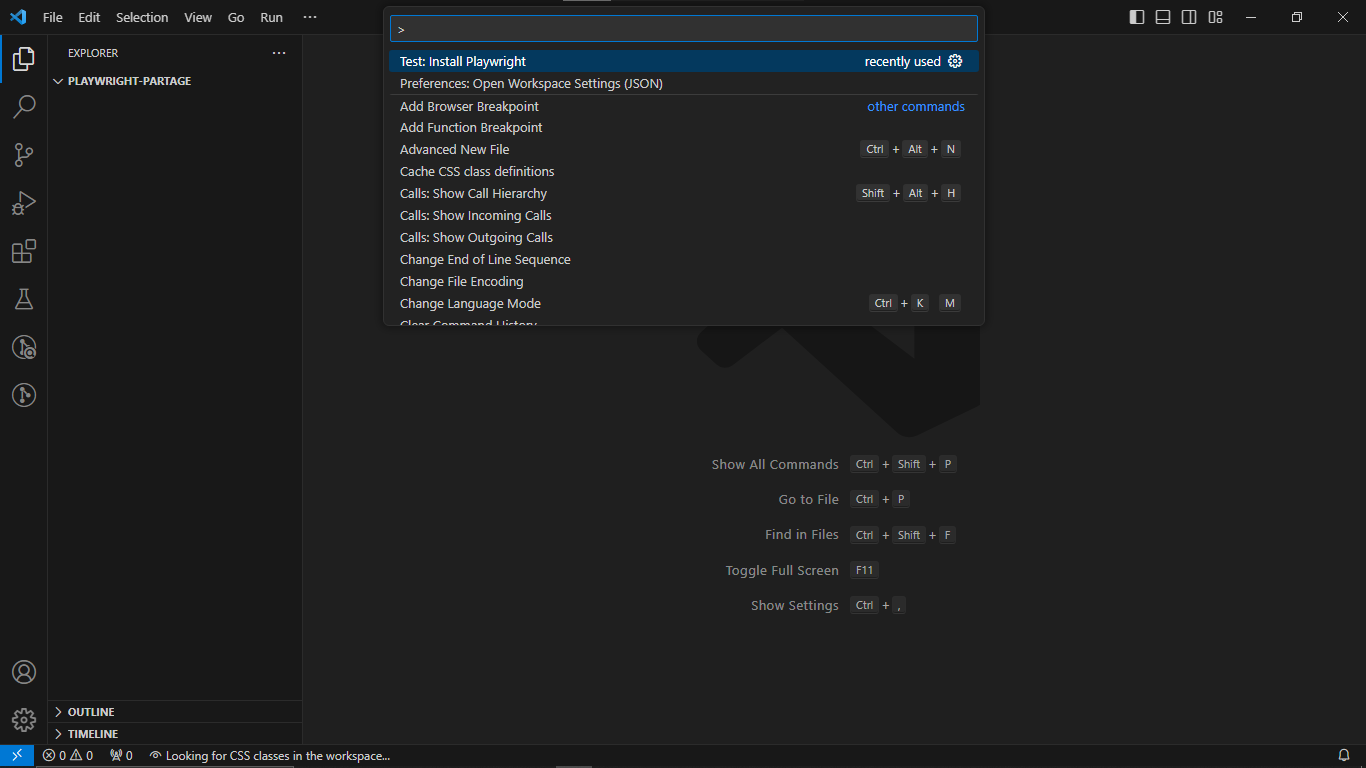
Voici le lien de téléchagement de VScode : <https://code.visualstudio.com/download>

1. Ouvrir Vscode et installer l’extension Playwright : installer l’extension Playwright test qui est validé par Microsoft

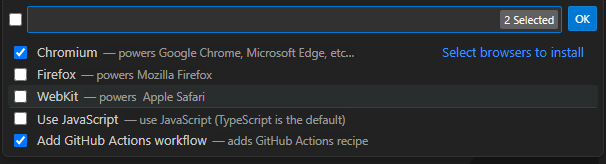


1. Ouvrir un nouveau dossier où installer le projet : ouvrir un nouveau dossier puis initier un projet playwright dans le dossier

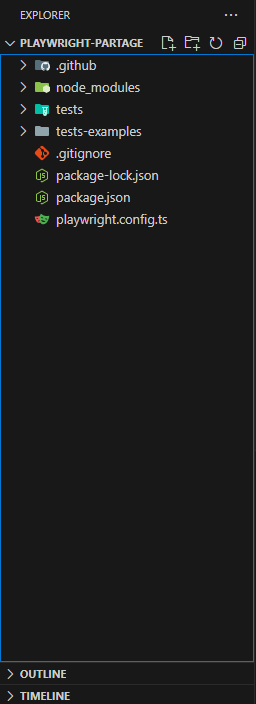
* Cliquer sur le raccourci : CTRL + SHIFT + P ; pour ouvrir la panelle de commande



* Choisir le langage et le navigateur : choisir le langage à utiliser soit Typescript soit Javascript, ensuite choisir le navigateur entre Chromium, Webkit et Mozila. Puis cliquer sur OK.

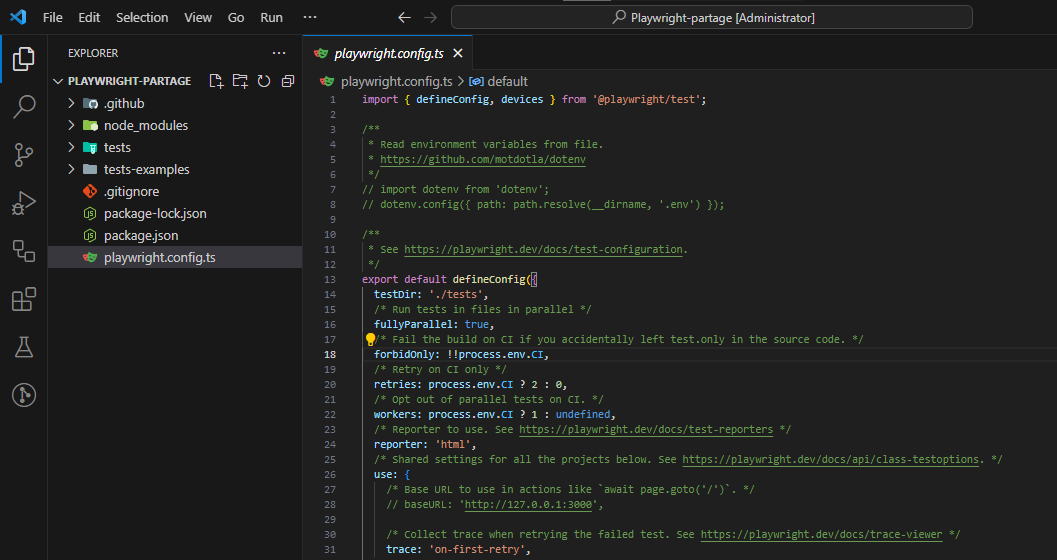


* Playwright installé : vous pouvez voir que playwright est installé dans le dossier vide créé initialement.



## Configuration de Playwright

La configuration par défaut est suffisante pour une approche débutante, consultez le fichier paywright.config.ts ou playwright.config.js.



Voici quelques paramètres recommandés pour une approche plus avancée :

* Headless Mode : Exécuter les tests en mode headless pour des tests plus rapides.
* Screenshots/Videos : Activer la capture d'écran et l'enregistrement vidéo pour faciliter le débogage.
* Parallel Execution : Configurer l'exécution en parallèle pour accélérer les tests.

## Utilisation de Playwright

Ecriture d’un premier test :

1. Créer un fichier example.spec.js.



1. Exécution du test : CRTL + J pour ouvrir le terminal, ensuite exécuter la commande suivante :

* npx playwright test example.spec.js

1. Génération de rapports : Playwright génère des rapports HTML par défaut. Vous pouvez les trouver dans le dossier playwright-report. Ou vous pouvez utiliser la commande suivante

* npx playwright show-report

## Avantages de Playwright

Voici les principaux avantages de Playwright, un outil de test automatisé développé par Microsoft pour tester les applications web :

1. Prise en Charge Multinavigateur

- Compatibilité Étendue : Playwright permet de tester les applications web sur tous les principaux navigateurs modernes, y compris Chromium (Chrome, Edge), WebKit (Safari), et Firefox. Cela garantit que l'application fonctionne correctement sur différents navigateurs sans avoir besoin de multiples outils.

- Headless et Headed : Il prend en charge à la fois le mode "headless" (sans interface graphique, pour des tests plus rapides) et "headed" (avec interface graphique, pour le débogage visuel).

2. Prise en Charge Multiplateforme

- Tests sur Divers Systèmes d'Exploitation : Playwright fonctionne sur Windows, macOS, et Linux, offrant une grande flexibilité pour tester les applications sur différents environnements.

3. API Puissante et Moderne

- Contrôle Complet : Playwright offre une API riche qui permet un contrôle granulaire sur les éléments du navigateur, comme les pages, les cadres, les requêtes réseau, et les réponses. Cela facilite la gestion de scénarios de test complexes.

- Écriture de Tests Intuitive : L'API de Playwright est conçue pour être intuitive et facile à utiliser, ce qui rend l'écriture de tests plus rapide et moins sujette aux erreurs.

4. Tests Automatisés Fiables

- Tests Stables : Playwright inclut des fonctionnalités qui améliorent la stabilité des tests, comme la gestion automatique des attentes (waits) pour les éléments, réduisant ainsi les risques de tests intermittents (flaky tests).

- Isolation des Contexte : Chaque test est exécuté dans un nouveau contexte de navigateur, garantissant qu'il n'y a pas de fuite d'état ou de dépendances entre les tests, ce qui améliore la fiabilité des résultats.

5. Prise en Charge des Tests End-to-End

- Scénarios Complexes : Playwright permet de simuler des interactions utilisateur complexes, comme les clics, les saisies de texte, les déplacements de souris, et bien plus, couvrant ainsi l'ensemble des interactions possibles dans une application web.

- Test des Fonctionnalités Riches : Il supporte des fonctionnalités avancées comme les tests de téléchargements et d'uploads de fichiers, les notifications push, les permissions de navigateur, etc.

6. Tests Parallèles et Distribution

- Exécution en Parallèle : Playwright permet d'exécuter des tests en parallèle, réduisant ainsi le temps total nécessaire pour exécuter une suite complète de tests, ce qui est crucial pour l'intégration continue (CI/CD).

- Tests Distribués : Il est possible de distribuer les tests sur plusieurs machines, ce qui est particulièrement utile pour les grandes suites de tests.

7. Support Multilingue

- Support de Différents Langages : Playwright offre des bindings pour plusieurs langages de programmation, y compris JavaScript/TypeScript, Python, C#, et Java. Cela permet aux équipes de développement d'utiliser Playwright dans le langage avec lequel elles sont les plus à l'aise.

8. Capture de Vidéos et de Screenshots

- Debugging Amélioré : Playwright peut capturer des vidéos et des captures d'écran pendant les tests, facilitant ainsi le débogage et l'identification des problèmes lorsque des tests échouent.

9. Support des Tests Mobile et Responsive

- Émulation d'Appareils : Playwright permet d'émuler des appareils mobiles pour tester les versions responsives des sites web, vérifiant ainsi que l'application fonctionne correctement sur différents types d'appareils.

- Test de Performances : Il permet également de simuler des conditions réseau différentes, comme des connexions lentes, pour tester la performance de l'application sous différentes conditions.

10. Documentation Complète et Support Actif

- Documentation Détaillée : Playwright est accompagné d'une documentation complète, avec de nombreux exemples et guides pratiques pour aider les utilisateurs à tirer le meilleur parti de l'outil.

- Communauté Croissante : Bien que relativement nouveau, Playwright dispose d'une communauté active et bénéficie du support de Microsoft, garantissant des mises à jour régulières et une amélioration continue de l'outil.

Ces avantages font de Playwright un outil très puissant pour les tests end-to-end, offrant flexibilité, fiabilité, et performance dans un seul package.

## Inconvénients de Playwright

Bien que Playwright soit un outil puissant pour les tests end-to-end, il présente également certains inconvénients qui peuvent poser des défis selon les cas d'utilisation. Voici quelques-uns des inconvénients associés à Playwright :

1. Complexité pour les Débutants

- Courbe d'Apprentissage : Bien que l'API soit moderne et bien conçue, Playwright peut être intimidant pour les débutants, en particulier pour ceux qui ne sont pas familiers avec les tests automatisés ou les concepts sous-jacents comme la gestion des contextes de navigateur et des sessions.

- Documentation Technique : La documentation, bien que complète, peut parfois être technique et difficile à comprendre pour ceux qui débutent, nécessitant un investissement de temps pour maîtriser l'outil.

2. Maturité Relativement Récente

- Nouveauté Relative : Comparé à d'autres outils de test plus anciens comme Selenium, Playwright est encore relativement nouveau. Cela signifie qu'il peut ne pas être aussi bien supporté dans certaines configurations d'entreprise ou par certains fournisseurs d'intégration continue (CI/CD).

- Moins de Support de la Communauté : Bien que Playwright ait une communauté active, elle est encore en croissance, ce qui peut signifier moins de ressources communautaires, de forums ou de bibliothèques tierces par rapport à des outils plus établis.

3. Intégration Limitée avec Certains Outils

- Compatibilité : Playwright peut ne pas s'intégrer aussi facilement avec certains outils ou frameworks existants, surtout dans des environnements où des solutions plus anciennes sont prédominantes. Certaines configurations spécifiques peuvent nécessiter des ajustements supplémentaires.

- Dépendance à Node.js : Playwright repose sur Node.js, ce qui peut être une contrainte pour les équipes ou les projets qui ne sont pas déjà basés sur cet écosystème.

4. Gestion des Tests et Maintenance

- Tests Flaky : Comme pour tout outil de test automatisé, il est possible que des tests deviennent "flaky" (c'est-à-dire qu'ils échouent de manière intermittente sans raison apparente). Bien que Playwright ait des fonctionnalités pour réduire cela, des tests complexes ou mal écrits peuvent encore souffrir de ce problème.

- Complexité des Scénarios Avancés : Les tests très complexes, impliquant plusieurs interactions simultanées ou des scénarios réseau complexes, peuvent nécessiter une gestion minutieuse pour s'assurer qu'ils restent fiables et maintenables.

5. Performances

- Consommation des Ressources : L'exécution de tests sur plusieurs navigateurs en parallèle, ou l'utilisation intensive de fonctionnalités avancées (comme les vidéos ou captures d'écran), peut nécessiter une machine avec des ressources matérielles importantes, ce qui peut être une contrainte dans des environnements de CI/CD avec des ressources limitées.

- Temps d'Exécution : Bien que Playwright soit rapide, certains tests, en particulier ceux qui impliquent des simulations complexes ou des tests de performance, peuvent prendre plus de temps à exécuter, ce qui peut rallonger le temps de feedback.

6. Manque de Fonctionnalités Prêtes à l'Emploi

- Moins de Plugins Tiers : Contrairement à des outils comme Selenium, qui bénéficie d'une vaste écosystème de plugins et d'intégrations tiers, Playwright a moins de plugins disponibles, ce qui peut limiter certaines fonctionnalités ou nécessiter du développement personnalisé pour combler les lacunes.

- Personnalisation et Configuration : Certaines personnalisations avancées peuvent nécessiter une configuration manuelle ou l'écriture de code supplémentaire, ce qui peut être un inconvénient pour les équipes cherchant une solution clé en main.

7. Support et Dépendances

- Dépendance à Microsoft : Étant un projet soutenu par Microsoft, certains utilisateurs peuvent être préoccupés par la direction future du projet et par la dépendance à l'égard des décisions stratégiques de Microsoft.

- Problèmes de Compatibilité : Avec l'évolution rapide des navigateurs, il peut y avoir des périodes où certaines fonctionnalités de Playwright ne sont pas immédiatement compatibles avec les dernières versions des navigateurs, nécessitant des mises à jour ou des correctifs.

8. Tests Mobiles Limités

- Émulation vs. Tests Réels : Bien que Playwright permette d'émuler des appareils mobiles, il ne remplace pas les tests sur de vrais appareils mobiles. Pour les applications qui nécessitent des tests approfondis sur mobile, cela peut être une limitation par rapport à des solutions spécialisées dans les tests mobiles.

Ces inconvénients ne signifient pas que Playwright n'est pas un bon outil, mais ils soulignent des aspects qui pourraient nécessiter des considérations ou des ajustements spécifiques selon les besoins de l'équipe ou du projet.