



CYPRESS

DÉCOUVERTE D'UN FRAMEWORK DE TESTS E2E



SOMMAIRE

Introduction
Mise en place de l'environnement
Premier pas
Cibler un élément
Liste des commandes
Créer ses propres commandes
L'intégration continue avec GitHub Actions
Discord Webhook
Le déploiement continu avec Vercel
Tester une API REST
Aller plus loin



INTRODUCTION

CYPRESS

FRAMEWORK DE TEST

Mise en place facile

Exécution rapide

Simple à déboguer

Unitaires / Intégration / E2E

Une interface belle et pratique

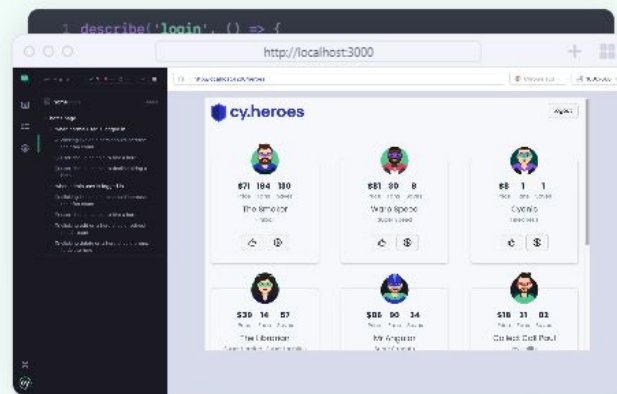


Test. Automate. Accelerate.

With Cypress, you can easily create tests for your modern web applications, debug them visually, and automatically run them in your continuous integration builds.

 npm install cypress

 Documentation



FORTUNE  classpass  Nielsen



MISE EN PLACE DE L'ENVIRONNEMENT

CHOIX D'UN NAVIGATEUR

OÙ NOS TESTS VONT S'EXÉCUTER ?

Firefox

Mozilla Corp.

Opera

Opera Software

Google Chrome

Google

Brave

Brave Software



Microsoft Edge

Microsoft

Veuillez éviter Microsoft Edge...

Les outils développeurs ainsi que le rendu de certains éléments ne sont pas à jour.



CHOIX D'UN IDE / ÉDITEUR DE CODE

NOTRE ENVIRONNEMENT DE DÉVELOPPEMENT



WebStorm

JetBrains



Visual Studio Code

Microsoft



VISUAL STUDIO CODE

CARACTÉRISTIQUES

Gratuit

Système d'extensions

Connecté au système de versionning






Le plus populaire

Et bien plus ...

Visual Studio Code

Editing evolved

Start

-  New File...
-  Open File...
-  Open Folder...
-  Clone Git Repository...
-  Connect to...

Recent

- sql-course E:\Labs\Git
- py-books-to-scrape E:\Labs\Git
- sylent-site E:\Labs\Git
- python-softdesk E:\Labs\Git
- py-eva-mentorat C:\Users\Aunim\Desktop
- More...



NODEJS

RUNTIME JAVASCRIPT



NodeJS

OpenJS Foundation

NodeJS est un runtime JavaScript basé sur le moteur V8 de Google Chrome. Lors de son installation, il arrive accompagné de NPM qui est un gestionnaire de dépendances. Il nous permettra d'installer Cypress facilement.

Pendant l'installation, veuillez à bien ajouter le chemin vers NodeJS parmi les variables d'environnement système.

```
node -v
v20.0.1
```

CRÉATION DU REPOSITORY

GITHUB



GitHub

Microsoft

GitHub vous permet de créer, stocker, modifier, fusionner
et collaborer sur des projets de développement



PREMIERS PAS



New repository



Sylent - © 2024


www.sylent.art

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner *

 aanimhnsn

Repository name *

/

Great repository names are short and memorable. Need inspiration? How about **silver-funicular** ?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

INSTALLER CYPRESS

INSTALLATION AVEC LE NODE PACKAGE MANAGER

cypress-course

```
> cypress ●
> node_modules ●
  .gitignore U
  cypress.config.js U
  package-lock.json U
  package.json U
```

Nous installons Cypress à l'aide de NPM, cette nouvelle dépendance apparaîtra dans le fichier package.json.

```
npm init -y
npm install cypress
```

.GITIGNORE

NE PAS TRAQUER LE DOSSIER NODE_MODULES

cypress-course

```
> cypress ●
> node_modules ●
  .gitignore U
  cypress.config.js U
  package-lock.json U
  package.json U
```

Le fichier .gitignore permet de ne plus traquer les éléments définis par le système de versioning git.

node_modules/

.gitignore

EXÉCUTER CYPRESS

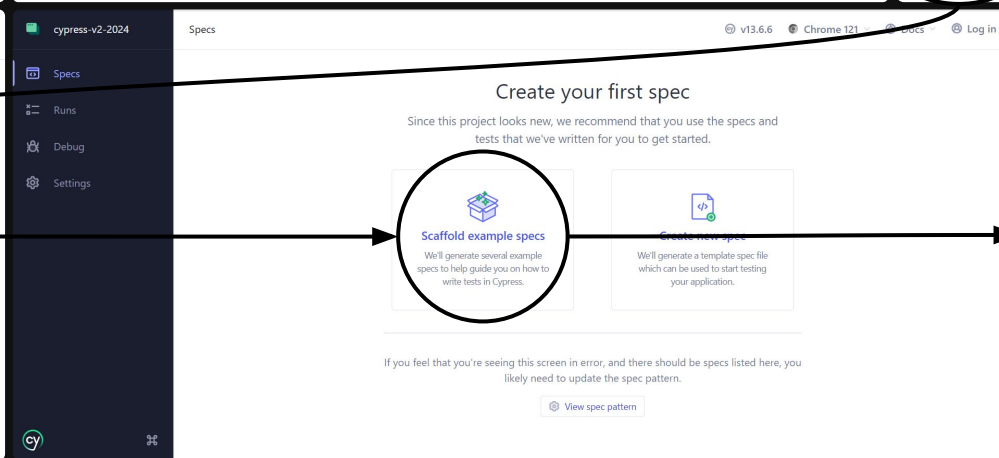
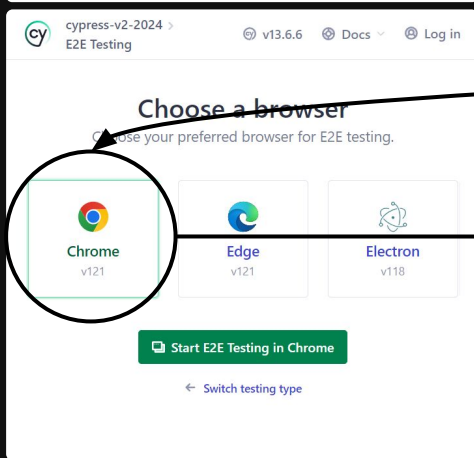
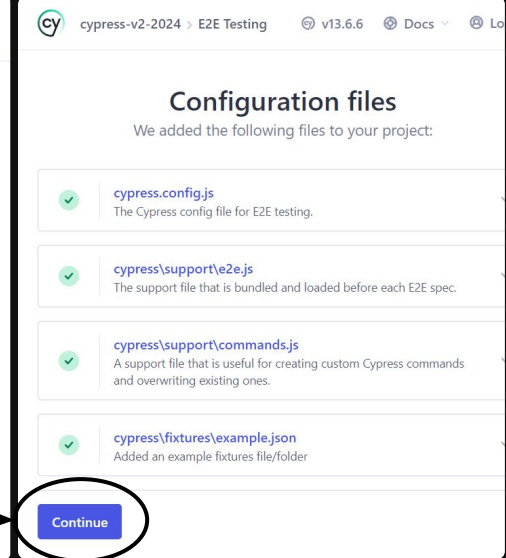
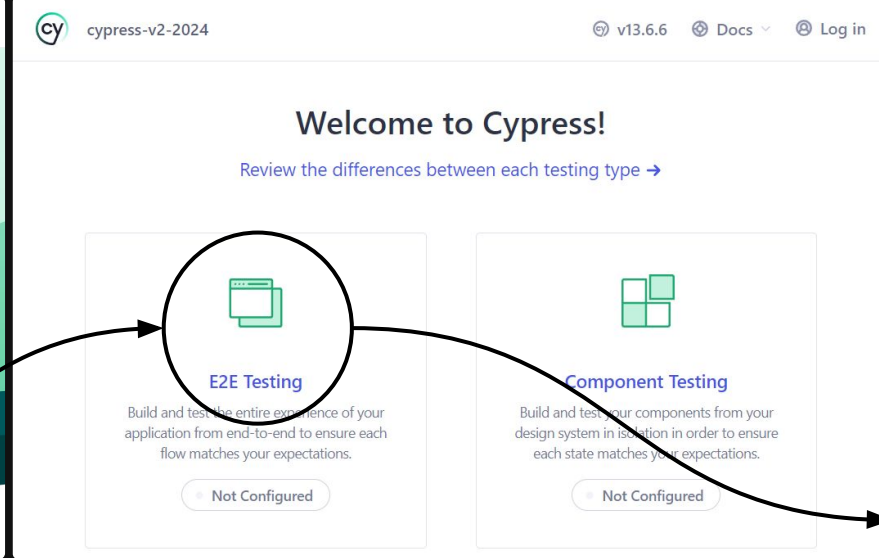
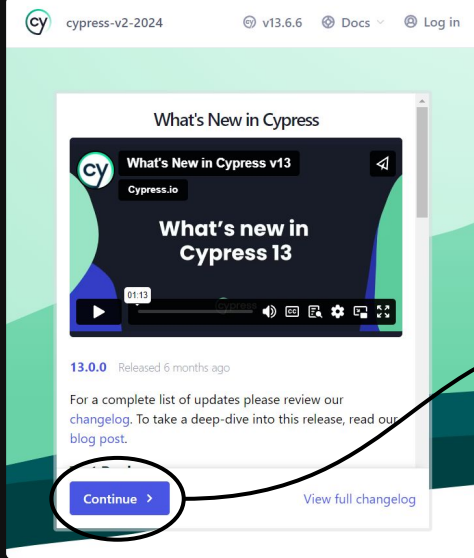
KICK START CYPRESS

Cypress offre une interface jolie et pratique pour visualiser et suivre l'exécution des tests e2e. L'API simple mais puissante de Cypress s'exécute aussi vite que votre navigateur peut restituer le contenu.

```
npx cypress open
```

console / terminal



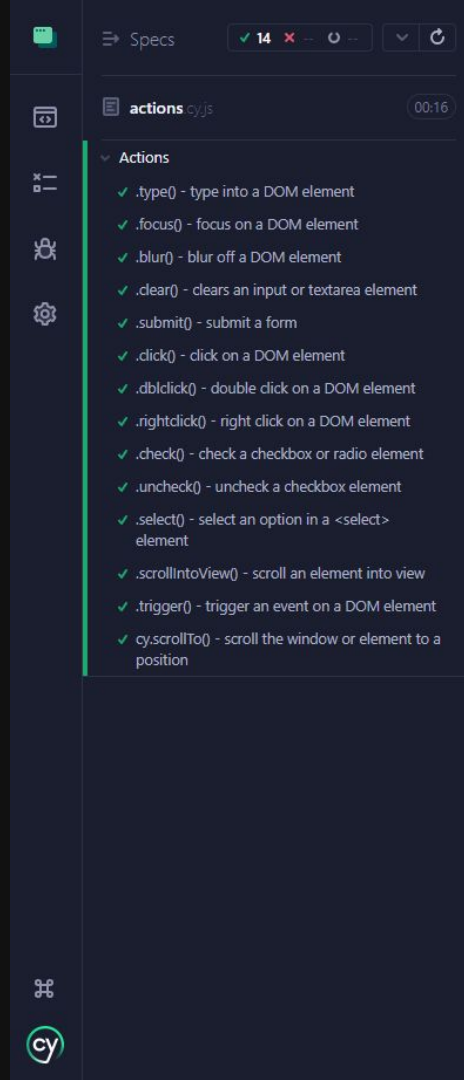


Vous êtes arrivé.e

VOIR UN EXEMPLE

CYPRESS ARRIVE AVEC DES TESTS D'EXEMPLE

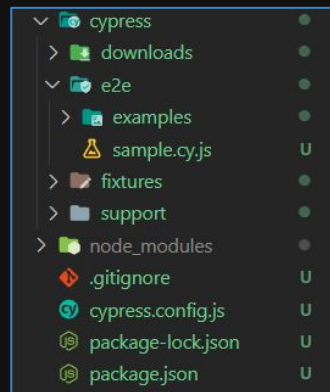
Un premier exemple intéressant à regarder est le fichier `actions.cy.js`, il comprend de nombreuses actions utilisateurs que l'API Cypress peut automatiser.



NETTOYER LES EXEMPLES

EXCLURE LES EXEMPLES DANS L'APPLICATION CYPRESS

cypress-course



Pour exclure les tests d'exemples, il faut modifier le fichier de configuration "cypress.config.js" comme ci-contre.

```
const { defineConfig } = require("cypress");

module.exports = defineConfig({
  e2e: {
    excludeSpecPattern: '**/examples/*.cy.js',
    setupNodeEvents(on, config) {
      // implement node event listeners here
    },
  },
});
```


ÉCRITURE ET EXÉCUTION DU PREMIER TEST

HELLO WORLD!

Dans ce premier test, nous vérifions à l'aide de Cypress, si sur notre page index.html, il y a bien un paragraphe contenant le texte "Hello World!"

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  ...
  <title>Hello World!</title>
</head>
<body>
  <p>Hello World!</p>
</body>
</html>
```

```
/// <reference types="cypress" />

context('Hello World!', () => {
  beforeEach(() => {
    cy.visit('.../app/index.html');
  });

  it('p should have text Hello World!', () => {
    cy.get('p')
      .should('have.text', 'Hello World!');
  });
});
```



MISE À JOUR DU .GITIGNORE

NE PLUS TRACKER LES TESTS D'EXEMPLE

Le tests d'exemple étant là juste pour nous aider à apprendre la syntaxe de l'API cypress, nous n'avons pas besoin de les traquer avec git.

```
node_modules/  
**/examples/*
```

.gitignore



TRAVAUX PRATIQUES

CHIFFRE DE CÉSAR

CRÉER ET TESTER UN COMPOSANT

Caesar Cypher

Vanilla JS

Cypher key

6

Hello World!

Cypher

Nkrru Cuxrj!

Créer à la racine du repository un dossier nommé caesar. A l'intérieur, créer et implémenter les fichiers index.html, main.css et main.js afin d'obtenir le résultat ci-contre.

Une fois le composant réalisé, vous allez devoir à l'aide de Cypress tester le composant. Le test se déroule en trois étapes qui sont :

- 1 Automatiser la complétion du formulaire
- 2 Automatiser la validation du formulaire
- 3 Vérifier l'affichage du bon résultat



CYPRESS AVEC FIRESHIP

100 *SECONDS OF*





CIBLER UN ÉLÉMENT

LA MÉTHODE .GET

CIBLER UN OU PLUSIEURS ÉLÉMENTS VIA UN SÉLECTEUR CSS

La méthode `.get(...)` prend en paramètre une chaîne de caractères correspondant à un sélecteur CSS. Cette méthode peut retourner un ou plusieurs éléments HTML.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  ...
  <title>La méthode .get</title>
</head>
<body>
  <button id="submit" class="btn">
    Valider
  </button>
</body>
</html>
```

```
/// <reference types="cypress" />

context('Retrieve element.s', () => {
  beforeEach(() => {
    cy.visit('.../app/index.html');
  });

  it('shows examples of get() method', () => {
    cy.get('button'); // Tag
    cy.get('#submit'); // Id
    cy.get('.btn'); // Class
  });
});
```


LA MÉTHODE .FIND

CIBLER UN OU PLUSIEURS ÉLÉMENTS ENFANTS À PARTIR D'UN SÉLECTEUR PARENT

La méthode `.find(...)` est appelée à partir d'un élément HTML parent et prend en paramètre une chaîne de caractères correspondant à un sélecteur CSS. Cette méthode peut retourner un ou plusieurs éléments HTML.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  ...
  <title>La méthode .find</title>
</head>
<body>
  <nav id="navbar" data-cy="navbar">
    <a href="#">Articles</a>
    <a href="#">Sign In</a>
    <a href="#">Login</a>
  </nav>
</body>
</html>
```

```
/// <reference types="cypress" />

context('Retrieve element.s', () => {
  beforeEach(() => {
    cy.visit('.../app/index.html');
  });

  it('shows an example of find() method', () => {
    cy.get('[data-cy=navbar]').find('a');
  });
});
```



LA MÉTHODE .CONTAINS

CIBLER UN ÉLÉMENT VIA SON CONTENU TEXTUEL

La méthode `.contains(...)` est en général appelée à partir d'un élément HTML parent et prend en paramètre une chaîne de caractères correspondant au contenu textuel de cet élément HTML. Cette méthode retourne seulement la première occurrence correspondante rencontrée.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  ...
  <title>La méthode .contains</title>
</head>
<body>
  <nav id="navbar" data-cy="navbar">
    <a href="#">Articles</a>
    <a href="#">Sign In</a>
    <a href="#">Login</a>
  </nav>
</body>
</html>
```

```
/// <reference types="cypress" />

context('Retrieve element.s', () => {
  beforeEach(() => {
    cy.visit('.../app/index.html');
  });

  it('shows an example of find() method', () => {
    cy.get('[data-cy=navbar]')
      .find('a')
      .contains('Sign In');
  });
});
```

BONNES PRATIQUES

DÉCOUPLER LES TESTS DU STYLE ET DE LA GESTION DES ÉVÉNEMENTS JS

```
cy.get('button').click();
```



Le pire, trop générique...

```
cy.get('.btn.btn-large').click();
```



Mauvais, couplé au style...

```
cy.get('#main').click();
```



Mauvais, couplé aux événements JS

```
cy.contains('Submit').click();
```



Mieux, mais couplé au contenu textuel...

```
cy.get('[data-cy=submit]').click();
```



Le top ! Isolé de tous changements.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  ...
  <title>Bonnes pratiques</title>
</head>
<body>
  <button
    id="main"
    class="btn btn-large"
    role="button"
    data-cy="submit"
  >
    Submit
  </button>
</body>
</html>
```





LISTE DES COMMANDES

LES 4 TYPES DE COMMANDES

APPRENDRE À CONNAÎTRE L'API CYPRESS

Ciblage / Requêtes

Les commandes de "requêtes" interrogent l'état de l'application et peuvent retourner un ou plusieurs éléments HTML sur lesquels il est possible d'agir par la suite.

Actions

Les actions sont des commandes Cypress qui interagissent avec votre application comme le ferait un utilisateur. Ils attendent que les éléments soient exploitables avant d'interagir avec la page.

Assertions

Les assertions sont des commandes qui vérifient l'état de l'application. Le test est suspendu tant que la condition n'est pas vérifiée ou que le temps accordé à l'assertion soit expiré.

Autres

Cypress dispose d'une variété de commandes supplémentaires pour faciliter l'écriture de tests.



CRÉER SES PROPRES COMMANDES

CRÉER SA PROPRE COMMANDE

COMPRENDRE L'ENJEU DU FICHIER COMMANDS.JS

Le framework Cypress offre la possibilité de créer nos propres méthodes afin de rendre l'écriture des tests plus concis et lisible.

Dans l'exemple ci-contre, nous créons une nouvelle commande personnalisée ".dataCy(..)" afin de simplifier la sélection d'un ou plusieurs éléments à travers l'attribut HTML "data-cy".

```
Cypress.Commands.add('dataCy', (value) => {  
  cy.get(`[data-cy=${value}]`);  
});
```

```
/// <reference types="cypress" />  
  
context('Retrieve element.s', () => {  
  beforeEach(() => {  
    cy.visit('.../app/index.html');  
  });  
  
  it('uses our dataCy command', () => {  
    cy.dataCy('submit');  
  });  
});
```



TRAVAUX PRATIQUES

LIGHTBOX

CRÉER ET TESTER UN COMPOSANT

Lightbox

Alpine JS & Tailwind CSS



Suivre les indications du fichier PDF qui vous sera fourni.



L'INTÉGRATION CONTINUE AVEC GITHUB ACTIONS

INTÉGRATION CONTINUE

AUTOMATISER LE LANCEMENT DES TESTS

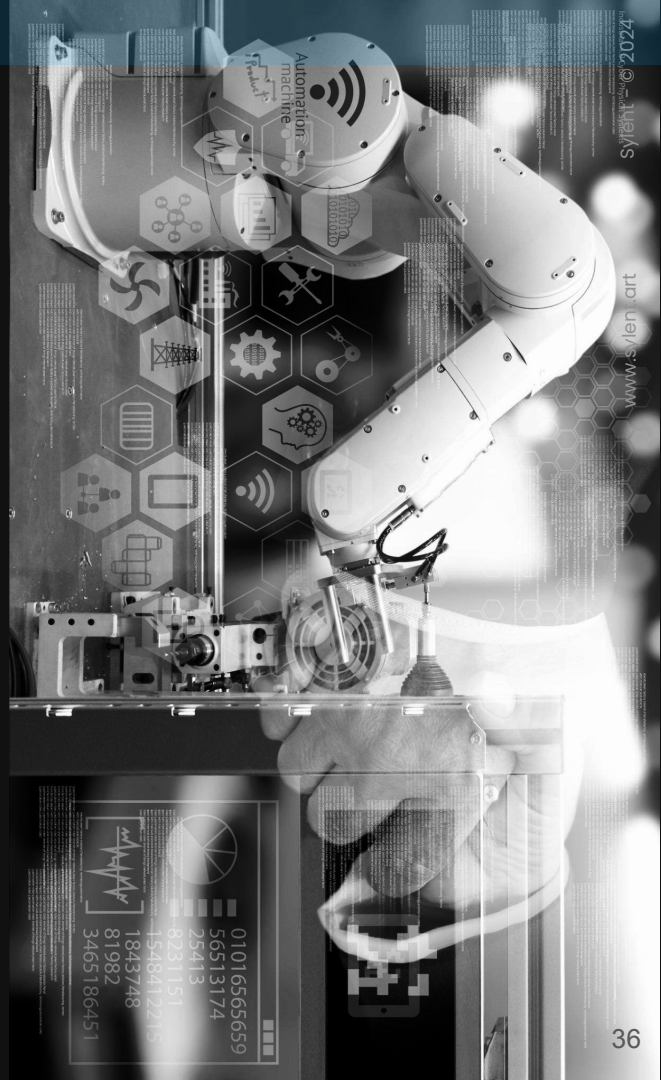
L'intégration continue est une méthode de développement de logiciel DevOps avec laquelle les développeurs intègrent régulièrement leurs modifications de code à un référentiel centralisé, suite à quoi des opérations de création et de test sont automatiquement menées.



DEVOPS

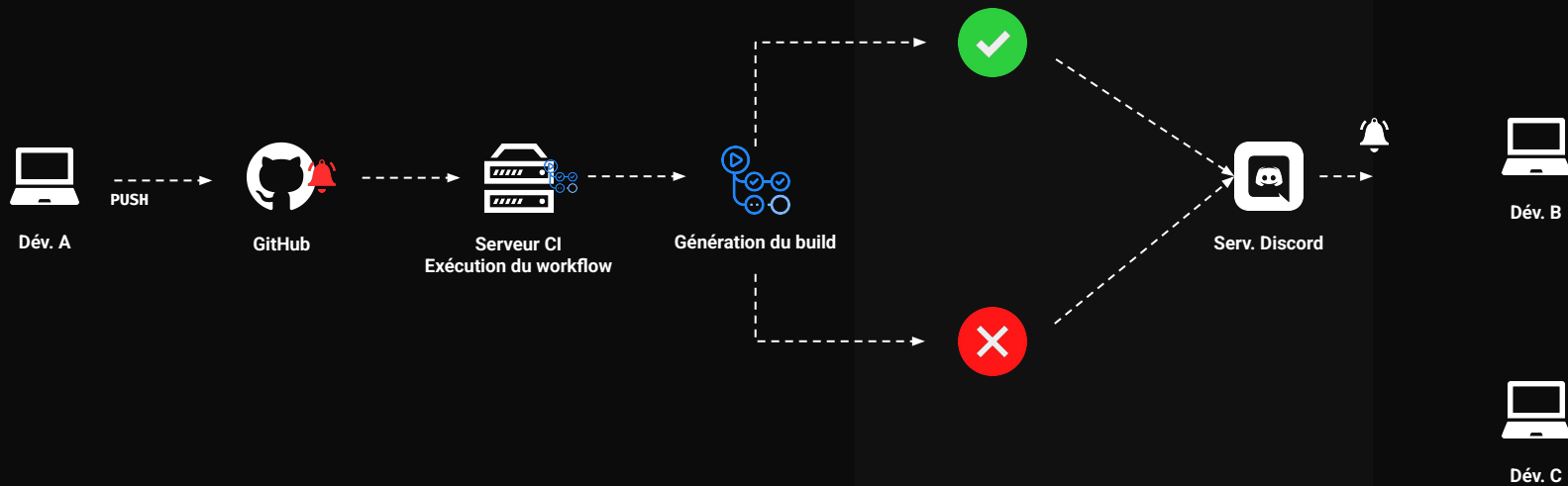
Automatisez, personnalisez et exécutez vos workflows de développement logiciel directement dans votre référentiel avec GitHub Actions.

Vous pouvez découvrir, créer et partager des actions pour effectuer n'importe quel travail de votre choix, y compris des tâches de CI/CD, et combiner des actions en un workflow entièrement personnalisé.



INTÉGRATION CONTINUE

AUTOMATISATION - ANG. : PIPELINE CI

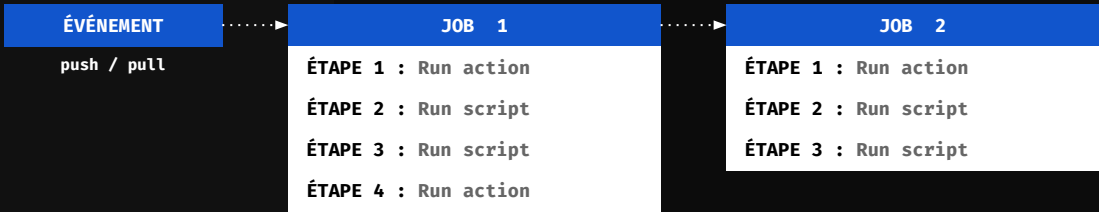


WORKFLOWS

GITHUB ACTIONS : FLUX DE TRAVAIL

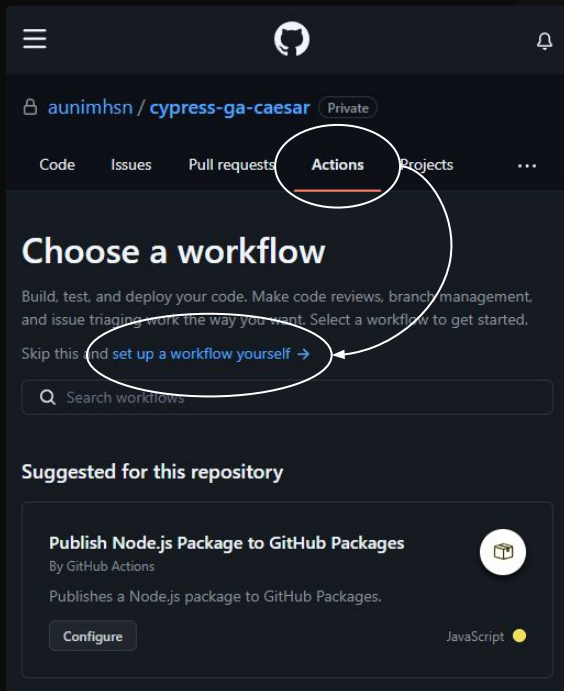
Un workflow ou flux de travail est un processus automatisé configurable qui exécutera un ou plusieurs jobs (travaux). Le workflow est défini par un fichier de configuration YAML et sera déclenché par un événement, par exemple un "push".

Dans notre cas, nous souhaitons lancer une série de tests Cypress au push de notre code vers le repository distant Github.



MISE EN PLACE DU WORKFLOW

CRÉATION DU FICHIER WORKFLOW.YAML



```
name: cypress-testing
```

```
on: push
```

```
jobs:
```

```
  cypress-run:
```

```
    runs-on: ubuntu-22.04
```

```
    steps:
```

```
      - name: Checkout
```

```
        uses: actions/checkout@v4
```

```
      # Install npm dependencies, cache them correctly
```

```
      # and run all Cypress tests
```

```
      - name: Cypress run
```

```
        uses: cypress-io/github-action@v6
```

NE PAS OUBLIER DE PULL

SYNCHRONISER LE REPO. DISTANT AVEC LE REPO. LOCAL

Nous venons de créer un fichier configuration YAML pour l'intégration continue directement sur la plateforme GitHub. Pour que le repository distant soit synchroniser avec notre repository local, on pull directement les modifications apportées.








LE RAPPORT DE BUILD

SUPERVISION - ANG. MONITORING







Voilà, nous venons d'automatiser le lancement des tests Cypress lorsque l'on push sur le repository distant. On peut maintenant consulter le rapport de build directement sur GitHub

cypress-run
succeeded 28 minutes ago in 28s

- >  Set up job
- >  Checkout
- >  Cypress run
- >  Post Checkout
- >  Complete job

cypress-run summary

Cypress Results

Result	Passed 	Failed 	Pending 	Skipped 	Duration 
Passing 	1	0	0	0	0.248s

Job summary generated at run-time



CI/CD AVEC FIRESHIP

100 seconds of



CI

CD

DEVOPS

EXPLAINED

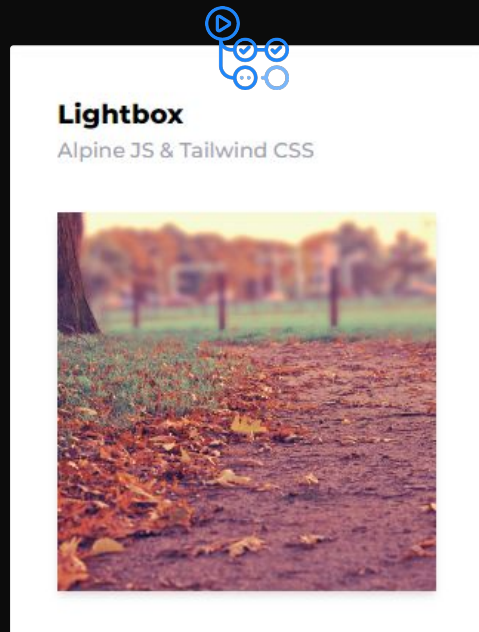




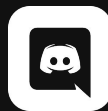
TRAVAUX PRATIQUES

LIGHTBOX AVEC GITHUB ACTIONS

MISE EN PLACE DE L'INTÉGRATION CONTINUE



Suivre les indications du fichier PDF qui vous sera fourni.



DISCORD WEBHOOK

DISCORD

OUTIL DE COMMUNICATION

Discord est un outil de communication en ligne qui comprend des fonctionnalités de messagerie instantanée, d'appels audio et vidéo. Principalement utilisé par les joueurs, Discord permet la création de serveurs personnalisés où les utilisateurs peuvent échanger des messages textuels, partager des fichiers, participer à des appels vocaux ou vidéo, et même diffuser en direct.

Avec une interface conviviale, des fonctionnalités de personnalisation avancées et une grande flexibilité, Discord est devenu populaire au-delà du monde du jeu, servant également de lieu de discussion pour diverses communautés, éducatives, professionnelles et sociales.



QU'EST-CE QU'UN WEBHOOK ?

NOTIFICATION EN TEMPS RÉEL

Un webhook, également appelé "rappel HTTP", est un moyen pour les applications web de communiquer entre elles en temps réel.

Les webhooks sont largement utilisés dans le développement d'applications, l'intégration de services tiers et la mise en place de notifications en temps réel dans divers domaines tels que l'intégration continue, le commerce électronique, les médias sociaux et bien plus...




CRÉER UN SERVEUR ET UN SALON DISCORD

OUTIL DE COMMUNICATION



Customize Your Server

Donne une personnalité à ton nouveau serveur en choisissant un nom et une icône. Tu pourras toujours les modifier plus tard.



NOM DU SERVEUR

En créant un serveur, tu acceptes la [Charte d'Utilisation de la Communauté Discord](#).

Retour Créer

Créer un salon

dans Salons textuels

TYPE DE SALON

Text
Partage des images, des GIF, des émojis, des opinions et des blagues ☒

Voice
Rejoins tes amis en vocal, en vidéo et avec le partage d'écran ☐

NOM DU SALON

☒ **Salon privé**
Seuls les membres et rôles sélectionnés pourront voir ce salon.

Annuler Créer un salon

- 1 Créer un serveur discord nommée cypress-server
- 2 Créer un salon textuel cypress-alert

AJOUT D'UN WEBHOOK SUR LE SALON CYPRESS-ALERT

PARTIE DISCORD

The image shows a Discord interface with several panels and annotations:

- Left Panel (Server List):** Shows the 'cypress-server' dropdown. Under 'SALONS TEXTUELS', the '# cypress-alert' channel is selected, with a gear icon circled.
- Middle Panel (Channel Settings):** The 'Intégrations' tab is selected and circled. Below it, the 'Webhooks' section shows '1 webhook' and a 'Créer un webhook' button circled.
- Right Panel (Webhook Creation):** A modal titled 'Tu n'as pas de webhook !' with a 'Créer un webhook' button circled.
- Bottom Panel (Webhook Configuration):** A modal for 'Captain Hook' (created 5 mars 2024 by sylent.art). It shows a 'cy' logo with a 'Supprimer' button. The 'NOM' field contains 'cypress-hook' and the 'SALON' dropdown is set to '#cypress-alert'. A 'Copier l'URL du webhook' button is circled, with a red arrow labeled 'Suppression de webhook' pointing to it.

The screenshot shows the GitHub 'Settings' page for a repository, specifically the 'Webhooks' section. The left sidebar contains various settings categories: General, Access, Collaborators, Code and automation, Branches, Tags, Rules, Actions, Webhooks (highlighted with a blue bar and a white circle), Codespaces, Pages, Security, Code security and analysis, Deploy keys, Secrets and variables, Integrations, GitHub Apps, and Email notifications. The main content area is titled 'Webhooks / Add webhook'. It includes a description of webhooks, a 'Payload URL' field containing 'https://discord.com/api/webhooks/1214608711973212280/Xdvi-ikf /github', a 'Content type' dropdown set to 'application/json', a 'Secret' field, an 'SSL verification' section with 'Enable SSL verification' selected, and a 'Which events would you like to trigger this webhook?' section with 'Send me everything.' selected. At the bottom, there is an 'Add webhook' button. White circles and arrows highlight the 'Webhooks' menu item, the 'Payload URL' field, the 'Send me everything.' option, and the 'Add webhook' button. A vertical watermark 'Sylent - © 2024' and 'www.sylent.art' are visible on the right side.

Projects Security Insights Settings

General

Access

Collaborators

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Codespaces

Pages

Security

Code security and analysis

Deploy keys

Secrets and variables

Integrations

GitHub Apps

Email notifications

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *

`https://discord.com/api/webhooks/1214608711973212280/Xdvi-ikf /github`

Content type

application/json

Secret

SSL verification

By default, we verify SSL certificates when delivering payloads.

☒ Enable SSL verification ☐ Disable (not recommended)

Which events would you like to trigger this webhook?

☐ Just the push event.

☒ Send me everything.

☐ Let me select individual events.

☒ Active

We will deliver event details when this hook is triggered.

Add webhook

Sylent - © 2024

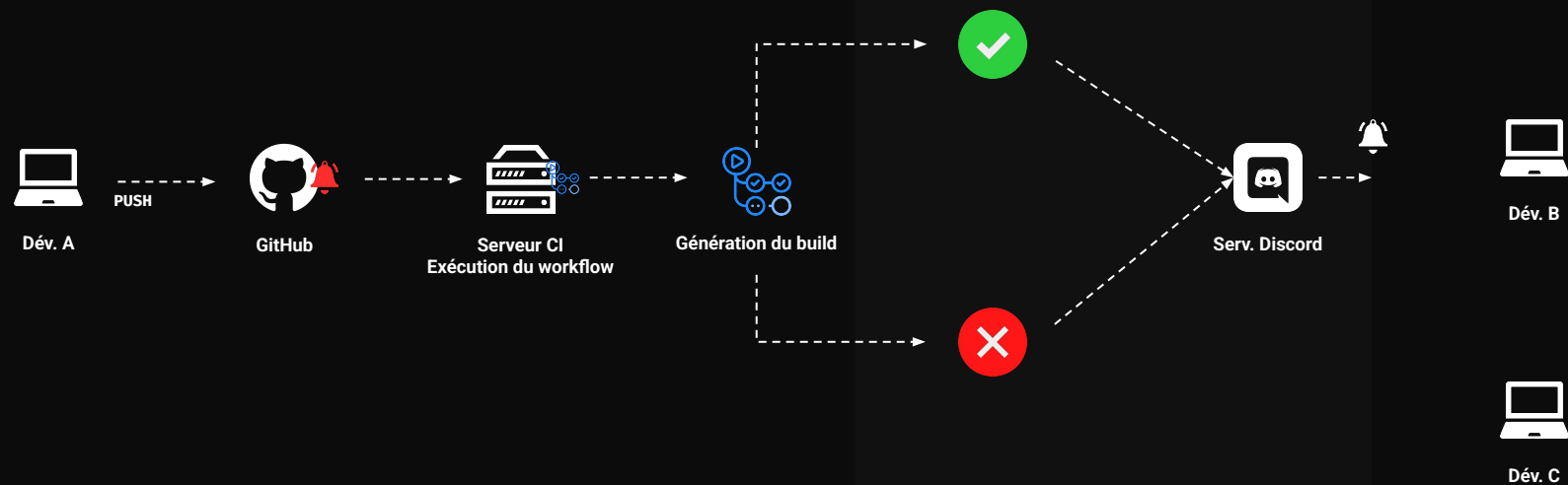
www.sylent.art

CONNECTER LE WEBHOOK AU REPO. DISTANT GITHUB



INTÉGRATION CONTINUE

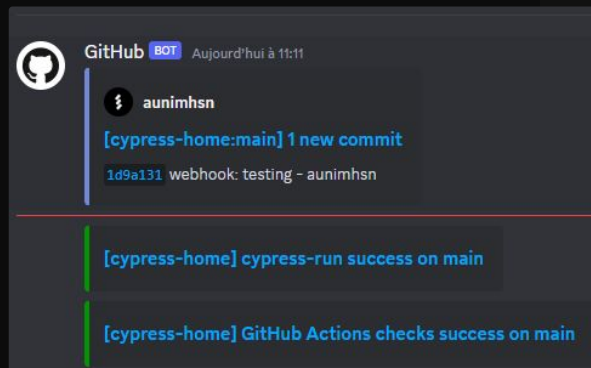
AUTOMATISATION - ANG. : PIPELINE CI



TESTER NOTRE WEBHOOK

MAINTENANT QUE TOUT EST EN PLACE, POUSSONS UNE MODIFICATION

Apportons une modification sur notre base de code afin de déclencher l'intégration continue et être alerté.e sur discord.

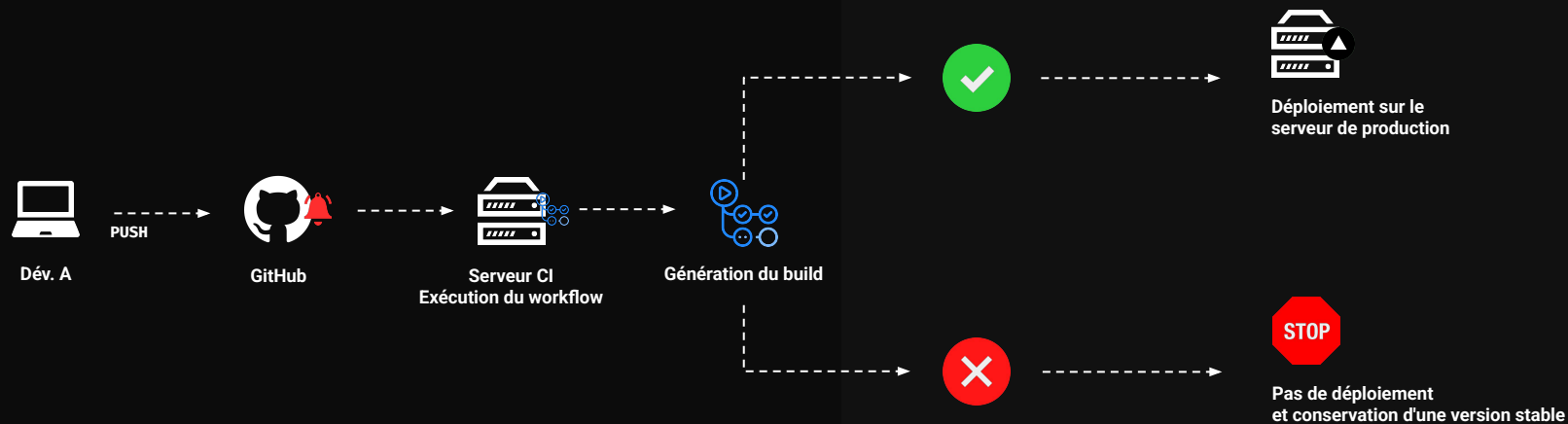




LE DÉPLOIEMENT CONTINU AVEC VERCEL

LE DÉPLOIEMENT CONTINU

DÉPLOYER AUTOMATIQUEMENT QUAND LES TESTS SONT AU VERT



VERCEL

OUTIL DE DÉPLOIEMENT CONTINU

Vercel est une plateforme cloud qui permet le déploiement rapide et continu d'applications web modernes. Il est particulièrement optimisée pour les projets frontend (React, Next.js, Svelte...)

Gratuit

Connecté à un système de versionning

Optimisé pour le frontend

Infrastructure serverless

Scalabilité transparente

Vercel named a Visionary for Cloud Native
Application Platforms

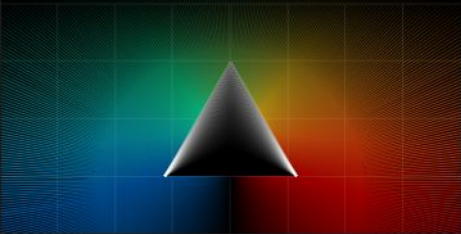
[Read the Report](#) >

Build and deploy on the AI Cloud.

Vercel provides the developer tools and cloud infrastructure to build, scale, and secure a faster, more personalized web.


▲ [Deploy](#)

[Get a Demo](#)



[AI Apps](#) [Web Apps](#) [Ecommerce](#) [Marketing](#)

runway build times
went from 7m to 40s.

 **Leonardo.Ai** saw a 95%


CONNEXION AVEC LE COMPTE GITHUB

C'EST BEAUCOUP SIMPLE AVEC LE COMPTE GITHUB


Log in to Vercel


Email Address

Continue with Email

 Continue with Google

 Continue with GitHub

 Continue with SAML SSO


 Continue with Passkey

Show other options

Don't have an account? [Sign Up](#)

Install Vercel

Where do you want to install Vercel?

 aunimhsn



Install Vercel

Install on your personal account **Aunim HASSAN**

for these repositories:

☒ All repositories

This applies to all current and future repositories owned by the resource owner. Also includes public repositories (read-only).

☐ Only select repositories

Select at least one repository. Also includes public repositories (read-only).

with these permissions:

- ✓ Read access to metadata
- ✓ Read and write access to administration, checks, code, commit statuses, deployments, issues, pull requests, and repository hooks

User permissions

Vercel can also request users' permission to the following resources. These permissions will be requested and authorized on an individual-user basis.

- ✓ Read access to email addresses

Install

Cancel

Next: you'll be directed to the GitHub App's site to complete setup.

IMPORTER LE REPOSITORY

LIER LE REPOSITORY AUQUEL ON SOUHAITE APPLIQUER LE DÉPLOIEMENT CONTINU

Import Git Repository

aunimhsn

Search...

py-ga-vercel · 7m ago	Import
py-battle-game-an21 · Jul 30	Import
json-course · Jul 24	Import
tailwindcss-components · Jul 10	Import
cypress-course-b3sopra-1 · Jul 10	Import

[Import Third-Party Git Repository →](#)



CRÉATION D'UN NOUVEAU PROJET



DERNIÈRE ÉTAPE AVANT LE DÉPLOIEMENT

Un projet Vercel permet la mise en ligne d'une application en quelques clics, sans gérer de serveur, avec une URL publique respectant la nomenclature suivante <https://<repository-name>.vercel.app>.

Une fois le repository importé, il faudra créer un nouveau projet Vercel où il faudra préciser des éléments de configuration.


New Project

Importing from GitHub

 aunimhsn/py-ga-vercel  main

Choose where you want to create the project and give it a name.

Vercel Team

 aunimhsn's projects

Hobby

Project Name

py-ga-vercel

Framework Preset

 Other

Root Directory

./

Edit

Build and Output Settings

Build Command

npm run vercel-build or npm run build

Output Directory

public if it exists, or .

Install Command

pip install -r requirements.txt

Environment Variables

Key

Value

EXAMPLE_NAME

I9JU23NF394R6HH

+ Add More

Tip: Paste an .env above to populate the form. [Learn more](#)

Deploy

Framework Settings

When using a framework for a new project, it will be automatically detected. As a result, several project settings are automatically configured to achieve the best result. You can override them below.

Framework Preset

Other

Build Command ?	<code>python3 -m pytest</code>	Override <input checked="" type="checkbox"/>
Output Directory ?	<code>.</code>	Override <input checked="" type="checkbox"/>
Install Command ?	<code>python3 -m pip install -r requirements.txt</code>	Override <input checked="" type="checkbox"/>
Development Command ?	<code>None</code>	Override <input type="checkbox"/>

[Learn more about Build and Development Settings](#) [↗](#)

Save

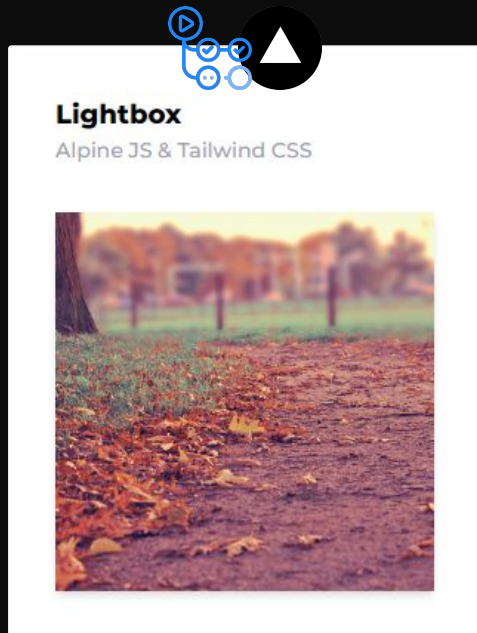
Mise en application étape par étape à travers un mini-projet Python
Réalisons cela ensemble



TRAVAUX PRATIQUES

LIGHTBOX AVEC GITHUB ACTIONS ET VERCEL

MISE EN PLACE DU DÉPLOIEMENT CONTINUE



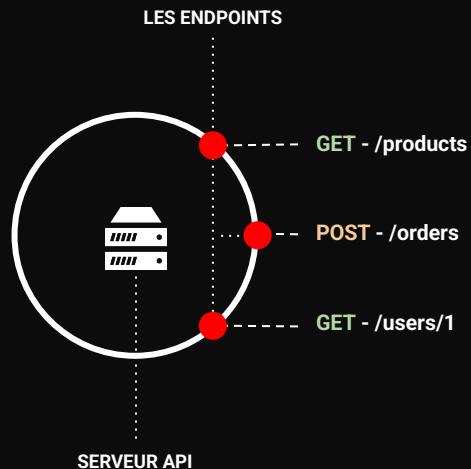
Mettre en place le déploiement continue avec Vercel sur le repository du projet LightBox.



TESTER UNE API REST

FONCTIONNEMENT D'UNE API REST

API : APPLICATION PROGRAMMING INTERFACE



L'API est une solution informatique qui permet à des applications de communiquer entre elles et de s'échanger mutuellement des services ou des données.

Nous pouvons faire l'analogie avec une télécommande (outil de requête normalisé) et un téléviseur proposant différentes chaînes (l'API et ses endpoints).

JSON-SERVER

AVOIR UNE FAUSSE API REST RAPIDEMENT

```
npm install -g json-server
```

```
json-server --watch db.json
```

- 1 Créer un fichier db.json et récupérer le contenu sur <https://github.com/aunimhsn/json-server/blob/main/app/db.json>
- 2 Installer globalement json-server
- 3 Lancer le serveur local de la fausse API REST
- 4 Ouvrir le navigateur et se rendre sur le lien du serveur local

LES CLIENTS REST

UN OUTIL PRATIQUE POUR TRAVAILLER AVEC LES APIS



Insomnia

Kong Inc.



Postman

Postman Inc.



Bruno

Bruno Software Inc.

LA PROPRIÉTÉ BASE URL

BONNES PRATIQUES

En ajoutant la propriété "baseUrl" dans le fichier de configuration, Cypress va préfixer toutes les urls passées en argument des méthodes `cy.visit()` et `cy.request()` de la valeur de "baseUrl".

```
const { defineConfig } = require("cypress");

module.exports = defineConfig({
  e2e: {
    baseUrl: 'http://localhost:3000',
    excludeSpecPattern: '**/examples/*.cy.js',
    setupNodeEvents(on, config) {
      // implement node event listeners here
    },
  },
});
```

TESTER UNE REQUÊTE GET

TESTER SI LES PROPRIÉTÉS DE CHACUNE DES VOITURES SONT BIEN EXISTANTES

La méthode `cy.request` permet de réaliser une requête HTTP en précisant la méthode / verbe et le endpoint.

Ainsi, en suivant la syntaxe de l'exemple, il est possible de tester le statut de la réponse et bien d'autres éléments via l'objet `response`.

```
/// <reference types="cypress" />

context('API Testing', () => {
  it('should test cars properties - GET /cars', () => {
    // Alias
    cy.request('GET', '/cars').as('cars');
    cy.get('@cars').should((response) => {
      expect(response.status).to.eq(200);
      response.body.forEach(car => {
        expect(car).to.have.all.keys(
          'id', 'brand', 'model',
          'color', 'year'
        );
      });
    });
  });
});
```

TESTER UNE REQUÊTE POST

TESTER L'AJOUT D'UNE NOUVELLE VOITURE

Lors d'une requête POST la méthode `cy.request` prend en troisième argument les données à envoyer. Ici une Fiat 500 bleu de 1972.

Parfois les données à envoyer via la méthode POST peuvent être bien plus volumineuses. C'est là qu'interviennent les fixtures afin de ne pas altérer la lisibilité des tests.

```
/// <reference types="cypress" />

context('API Testing', () => {
  it('should add a car - POST /cars', () => {
    let car = {
      "brand": "Fiat",
      "model": "500",
      "color": "blue",
      "year": 1972
    };
    cy.request('POST', '/cars', car);

    // Requête la BDD, faire une assertion sur le nbr de voitures
    // [...]
  });
});
```

LES FIXTURES

DÉFINITION ET UTILISATION

En test logiciel, une fixture se réfère à un ensemble prédéfini de données, d'états ou de conditions qui sont utilisés comme base pour effectuer des tests.

Les fixtures sont créées dans le but de garantir un environnement de test stable et reproductible.

```
{
  "brand": "Fiat",
  "model": "500",
  "color": "blue",
  "year": 1972
}
```

```
/// <reference types="cypress" />

context('API Testing', () => {
  it('should add a car - POST /cars', () => {
    cy.fixture('car').then((car) => {
      cy.request('POST', '/cars', car);
    })
  });
});
```

JSON-SERVER-RESET

DB.JSON - REVENIR À L'ÉTAT INITIAL APRÈS UNE REQUÊTE POST

```
npm install json-server-reset
```

1

Installer json-server-reset

```
json-server db.json --middlewares ./node_modules/json-server-reset
```

2

--

3

Créer une nouvelle fixture cars.json avec les trois voitures

4

Ajouter le code ci-dessous dans le hook beforeEach

```
cy.fixture('cars').then((cars) => {  
  cy.request('POST', '/reset', cars);  
});
```



ALLER PLUS LOIN

QUELQUES PRINCIPES

C'EST TOUJOURS BIEN DE LES GARDER EN TÊTE

KISS

Le principe Kiss pour "Keep it Simple, Stupid" signifie qu'il faut favoriser la simplicité dans la conception et éviter toute complexité inutile.

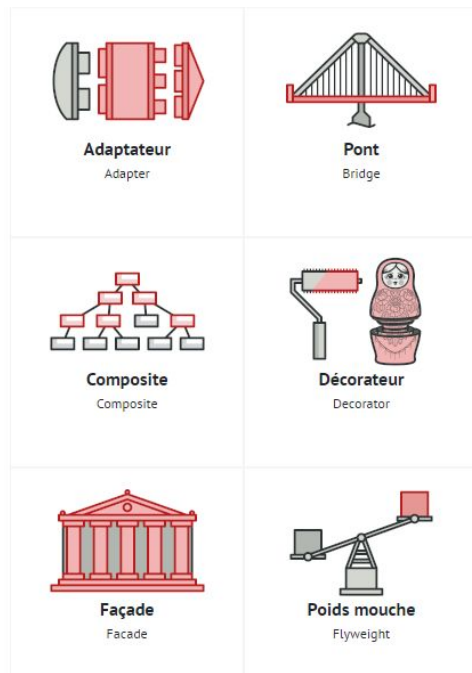
DRY

DRY pour "Don't Repeat Yourself" est un principe de programmation visant à exercer les développeurs à reconnaître des duplications puis de trouver le moyen de les supprimer. Ces duplications peuvent se produire évidemment dans le code mais aussi à tout niveau de l'application comme par exemple dans son architecture, dans les tests unitaires ou dans les processus de déploiement.

LES DESIGN PATTERNS

LES PATRONS DE CONCEPTION

Les patrons de conception plus communément appelés design patterns en anglais sont des solutions architecturale à des problèmes récurrents de la conception de logiciels. Chaque patron est une sorte de plan ou de schéma que vous pouvez personnaliser afin de résoudre un problème récurrent dans votre code.



• • •

LE LIVRE CLEAN CODE

LA RÉFÉRENCE

Grâce à cet ouvrage, vous apprendrez à rédiger du bon code, ainsi qu'à le nettoyer « à la volée », et vous obtiendrez des applications plus robustes, plus évolutives et donc plus durables.

Concret et pédagogique, ce manuel se base sur les bonnes pratiques d'une équipe de développeurs aguerris réunie autour de Robert C. Martin, expert logiciel reconnu. Il vous apprendra les valeurs d'un artisan du logiciel et fera de vous un meilleur programmeur.

Robert C. Martin Series



Sylent - © 2024

Clean Code

A Handbook of Agile Software Craftsmanship

www.sylent.art

Robert C. Martin



Foreword by James O. Coplien 75



I'm a never nester, CodeAesthetic



API testing in Python, ArjanCodes



MERCI DE VOTRE ATTENTION