Créer une application React

Utilisez Vite pour créer facilement une application React sur votre propre ordinateur.

React est une puissante bibliothèque JavaScript développée par Meta (anciennement Facebook) pour créer des interfaces utilisateur dynamiques. Son architecture basée sur des composants et son rendu performant ont largement contribué à son adoption par les développeurs, la classant au deuxième rang des frameworks web les plus appréciés selon l' enquête StackOverflow 2024 auprès des développeurs . Cet article vous guidera dans la configuration de votre première application React et suppose que vous maîtrisez les éditeurs de texte et la navigation en ligne de commande.

Remarque : Nous avons déjà recommandé l'utilisation de `create-react-app`(CRA) pour amorcer un projet React. Depuis, l'équipe React a abandonné cet outil. Cet article se concentre sur l'utilisation de Vite (prononcé « veet »), l'outil de build moderne devenu la solution légère non-framework préférée pour le développement React.

Se préparer

Nous utiliserons le gestionnaire de paquets Node (npm). Node doit donc être installé sur votre ordinateur. Pour vérifier si Node est installé, exécutez la commande suivante dans votre terminal :

```
node -v
```

Si vous avez installé Node, cette commande renverra un numéro de version, comme `v23.1.0`.

S'il n'est pas déjà installé, suivez les étapes de la section Configuration locale du nœud avant de continuer.

Pour utiliser Vite, vous aurez besoin de Node v18+ ou 20+, alors assurez-vous d'installer la bonne version avant de continuer !

Lorsque vous installez Node, npm est également installé automatiquement sur votre ordinateur. Cependant, npm est distinct de Node.js et a tendance à se mettre à jour plus fréquemment. Par conséquent, même si vous venez d'installer Node (et donc npm), il est conseillé de mettre à jour votre npm. Heureusement, npm sait se mettre à jour tout seul !

Pour mettre à niveau vers la dernière version de npm sur *nix (OSX, Linux, etc.), vous pouvez exécuter cette commande dans votre terminal :

```
sudo npm install -g npm@latest
```

Pour effectuer une mise à niveau sous Windows, suivez les étapes décrites dans la documentation npm .

Configuration de l'application standard

Bien qu'il soit possible de créer manuellement une application React, l'utilisation d'un outil de création comme Vite nous permet de générer rapidement une application React standard avec toute la configuration nécessaire.

L'utilisation de Vite offre plusieurs avantages :

- une structure de projet cohérente que vous reconnaîtrez dans différents projets React
- un processus de construction optimisé et prêt à l'emploi
- un serveur de développement rapide avec remplacement de module à chaud
- outils front-end modernes avec une configuration minimale

Pour créer un projet Vite, vous pouvez utiliser la commande npm `create`, qui exécute le `create-vite`package sans l'installer globalement au préalable. Lors de l'exécution de la commande, npm vous demandera l'autorisation de télécharger et d'exécuter le package. Cette approche préserve la propreté de vos dépendances globales et vous garantit de toujours utiliser la dernière version de l'outil.

Ouvrez votre terminal et exécutez l'une des commandes suivantes en fonction de votre gestionnaire de paquets préféré :

Avec npm :

```
npm create vite@latest
```

Avec du fil :

```
yarn create vite
```

Avec pnpm :

```
pnpm create vite
```

Après avoir exécuté la commande, suivez les invites interactives :

1. entrez le nom de votre projet (par exemple, `my-react-app`)
2. sélectionnez `React`comme cadre
3. choisissez votre variante préférée (JavaScript ou TypeScript)

Nous utiliserons la variante JavaScript, qui ne dispose pas de vérification de type. Localement, vous devez utiliser la variante adaptée à votre cas d'utilisation.

Vous pouvez également spécifier ces options directement dans la commande :

```
# npm
npm create vite@latest my-react-app -- --template react

# yarn
yarn create vite my-react-app --template react

# pnpm
pnpm create vite my-react-app --template react
```

Une fois terminé, vous recevrez un récapitulatif de votre sélection d'invite et de l'emplacement du projet :

```
> npx
> create-vite

|
◇  Project name:
|  my-react-app
|
◇  Select a framework:
|  React
|
◇  Select a variant:
|  JavaScript
|
◇  Scaffolding project in /Users/codecademy/Desktop/vite/my-react-app...
|
└  Done. Now run:

 cd my-react-app
 npm install
 npm run dev
```

En suivant les instructions, accédez au répertoire de votre projet (`cd my-react-app`), puis exécutez la commande pour installer vos dépendances (`npm install`, `yarn`, ou `pnpm install`). Cela dépend du gestionnaire de paquets choisi. Avant d'exécuter l'application, examinons la structure du projet pour voir ce que Vite a créé pour nous.

Structure de l'application React

Ouvrez l'application avec l'éditeur de texte de votre choix. Vous devriez voir la structure de fichier suivante :

```
└── my-react-app
    ├── public
    │   └── vite.svg
    ├── src
    │   ├── App.css
    │   ├── App.jsx
```

```
    │   ├── assets
    │   │   └── react.svg
    │   ├── index.css
    │   └── main.jsx
    ├── .gitignore
    ├── eslint.config.js
    ├── index.html
    ├── package-lock.json
    ├── package.json
    ├── README.md
    └── vite.config.js
```

Vite a configuré la structure principale de l'application et plusieurs paramètres de développement. La plupart des éléments affichés ne seront pas visibles pour les visiteurs de votre application web. Vite utilise un système de compilation moderne qui transforme les répertoires et fichiers en ressources statiques optimisées. Ces ressources statiques sont accessibles aux visiteurs de votre site.

Ne vous inquiétez pas si vous ne maîtrisez pas les outils de build. L'un des avantages d'utiliser Vite pour configurer notre application React est que nous pouvons éviter toute configuration manuelle du processus de build. Vite gère cette complexité pour nous tout en offrant un développement extrêmement rapide et des builds de production optimisés.

Explorons ce que fait chaque fichier et répertoire :

Répertoires clés

`/node_modules`

Ce répertoire contient toutes les dépendances et sous-dépendances spécifiées dans votre `package.json`fichier. Aucune modification n'est requise ici ; il est automatiquement ajouté, `.gitignore`car ces fichiers n'ont pas besoin d'être validés dans votre dépôt.

`/public`

Ce répertoire contient des ressources statiques qui seront servies directement sans être traitées par Vite. Les fichiers de ce répertoire seront copiés dans le répertoire de build pendant la production. Il contient :

- `vite.svg`- Le logo Vite, que vous pouvez remplacer par votre propre favicon.

`/src`

C'est ici que se trouve le code de votre application React et que vous passerez la majeure partie de votre temps. La structure initiale comprend :

- `/assets`- un répertoire pour stocker des ressources statiques comme des images qui seront traitées par Vite
- `App.jsx`- le composant React principal de votre application
- `App.css`- styles pour le composant App
- `main.jsx`- le point d'entrée de votre application React (ce fichier sert le même objectif que `index.js`ou `index.jsx`dans d'autres configurations React)
- `index.css`- styles globaux pour votre application

Remarque : Tout au long de la section « Apprendre React », nous utiliserons `index.jsx`le terme « fichier de point d'entrée », courant dans de nombreux projets React. Dans le modèle par défaut de Vite, ce fichier est nommé `main.jsx`, mais il a exactement le même objectif. Si vous préférez suivre les leçons à la lettre, vous pouvez le renommer `main.jsx`et `index.jsx`mettre à jour la référence du script en `index.html`conséquence.

Fichiers clés

`index.html` Situé dans le répertoire racine, ce fichier HTML sert de point d'entrée à votre application. Vite y injecte votre JavaScript lors du processus de compilation. Vous pouvez modifier directement ce fichier pour ajouter des balises méta, modifier le titre de la page ou inclure des scripts ou des styles supplémentaires.

`package.json` Ce fichier décrit tous les paramètres de votre application React, notamment :

- dépendances requises pour l'application
- scripts pour le développement, la création et la prévisualisation de votre application
- autres métadonnées sur votre projet

Ici, vous trouverez des scripts importants tels que :

- `dev`: démarre le serveur de développement
- `build`: crée une version prête pour la production
- `preview`: sert la version de production localement pour les tests

`vite.config.js` Il s'agit du fichier de configuration de Vite, qui vous permet de personnaliser le comportement de l'outil de build. Par défaut, il inclut simplement le plugin React, mais vous pouvez l'étendre pour ajouter des fonctionnalités telles que :

- chemins d'alias pour les importations
- options de construction personnalisées
- plugins supplémentaires
- paramètres de proxy pour les requêtes API

`eslint.config.js` Ce fichier configure ESLint pour votre projet, ce qui permet de détecter les erreurs et d'appliquer le style de code. Le modèle Vite est fourni avec une configuration de base qui étend les règles recommandées par React.

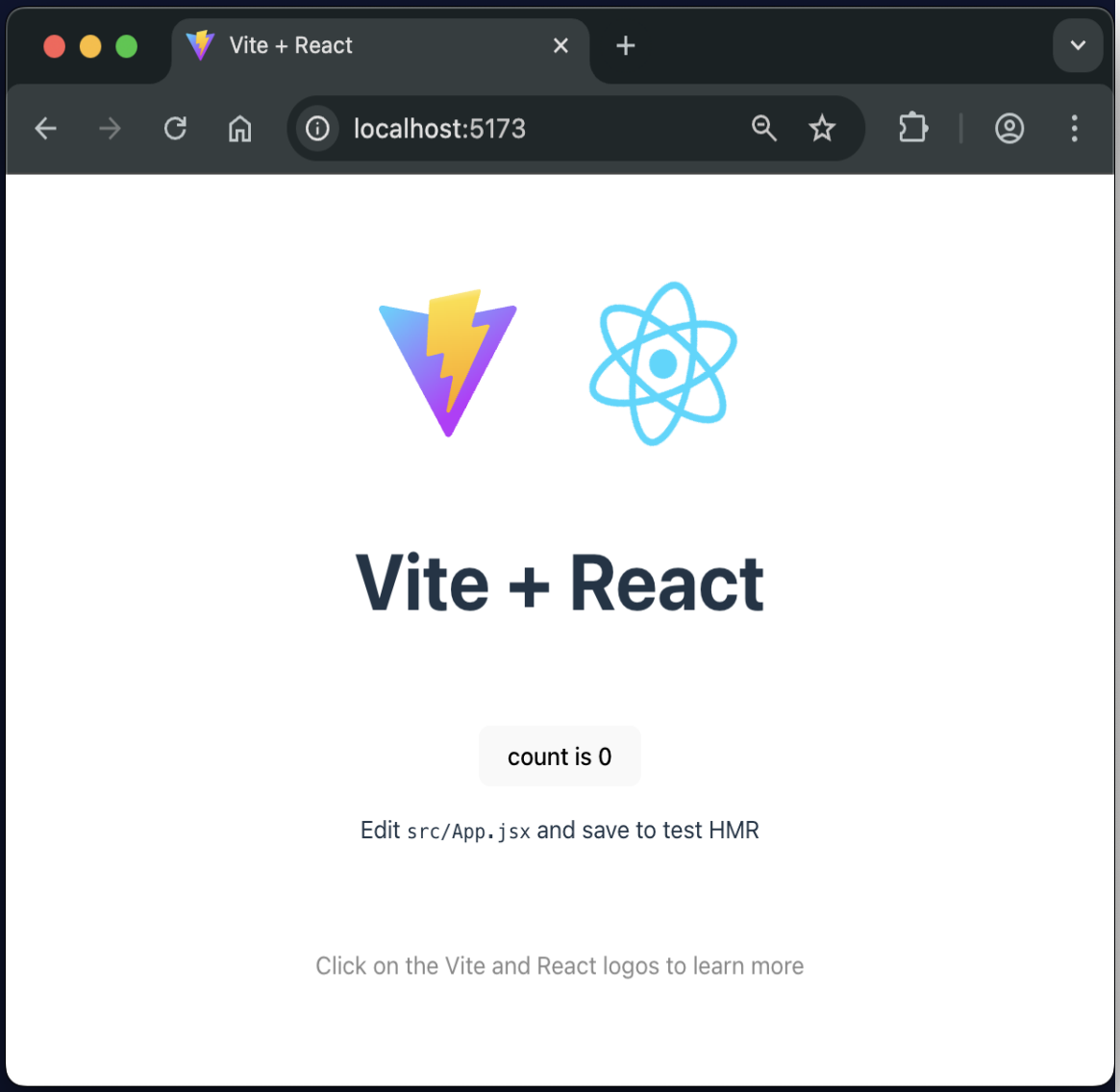
`.gitignore` Ce fichier indique à Git quels fichiers et répertoires ignorer lors de la validation du code, tels que node_modules, les répertoires de build et les fichiers d'environnement.

À mesure que votre application React se développe, il est courant d'ajouter un `/components`répertoire pour organiser les composants et les fichiers liés aux composants et un `/views`répertoire pour organiser les vues React et les fichiers liés aux vues.

Démarrage du serveur de développement d'applications React

Auparavant, vous avez changé de répertoire dans votre projet, la seule chose qui reste à faire maintenant est de démarrer votre projet avec `npm run dev`, `yarn dev`, ou `pnpm dev`.

L'application devrait maintenant être activée `http://localhost:5173/`. Vous pouvez saisir manuellement cette adresse dans la barre d'adresse de votre navigateur. Vous verrez alors une page ressemblant à l'image suivante :



Toute modification apportée au code source sera mise à jour ici. Voyons cela en action. Laissez l'onglet du terminal actif (il est occupé à gérer l'application React) et ouvrez-le `src/App.jsx`dans votre éditeur de texte préféré. Vous verrez ce qui ressemble à un mélange de JavaScript et de HTML. Il s'agit de JSX, qui permet à React d'ajouter la syntaxe XML à JavaScript. Il offre une méthode intuitive pour créer des composants React et est compilé en JavaScript à l'exécution. Nous en parlerons plus en détail dans un autre article, mais pour l'instant, effectuons une simple modification et observons la mise à jour dans le navigateur.

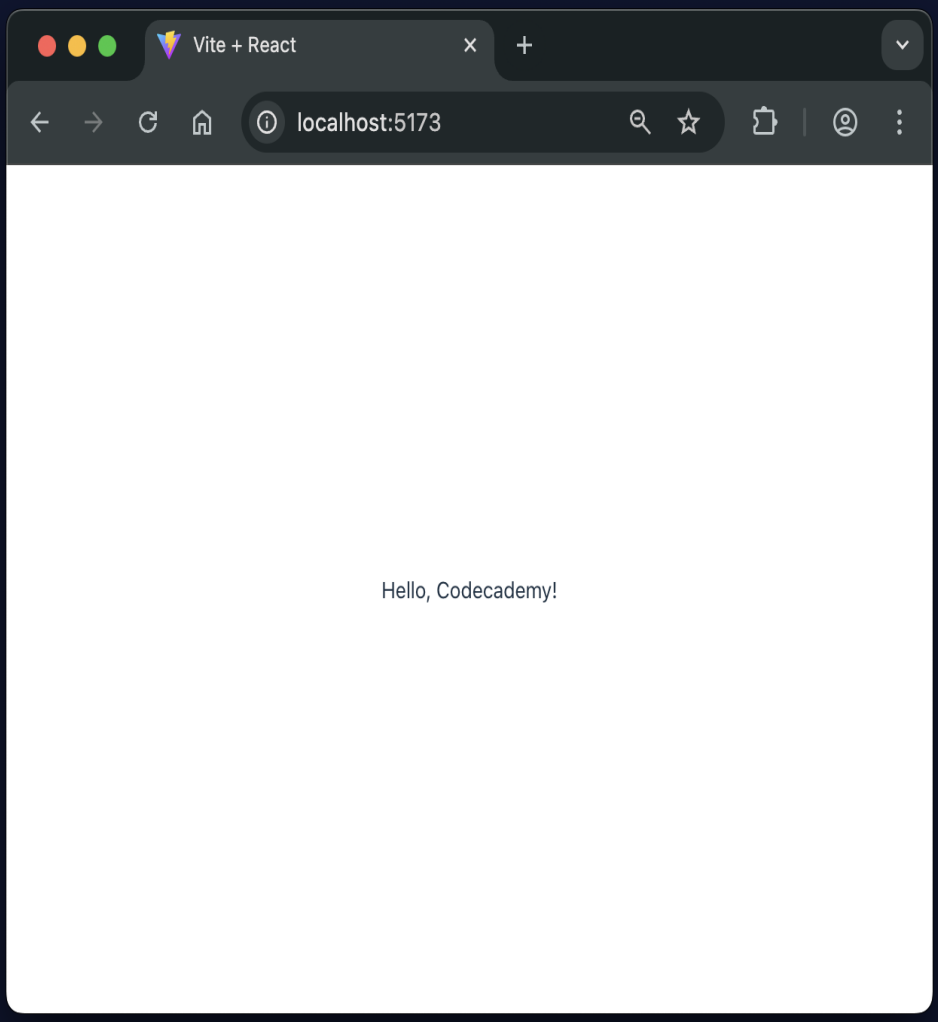
Dans `App.jsx`, remplacez le contenu par ce qui suit :

```
import './App.css'

function App() {
 return (
 <>
 Hello, Codecademy!
 </>
 )
}

export default App
```

Si vous avez laissé le terminal en cours d'exécution, vous devriez pouvoir basculer vers votre navigateur et voir la mise à jour :



Félicitations ! Vous êtes opérationnel avec React et pouvez commencer à ajouter des fonctionnalités à votre application.

Prochaines étapes

Maintenant que vous avez configuré votre première application React avec Vite, voici ce qu'il faut faire ensuite :

1. Terminez votre parcours d’apprentissage React pour maîtriser les fondamentaux de React.

2. Configurez votre environnement de débogage : consultez notre article sur les outils de développement React pour savoir comment inspecter les composants et suivre les performances.
3. Envisagez d'explorer les frameworks React : une fois que vous avez compris les bases de React, essayez Learn Next.js pour les fonctionnalités de production telles que le rendu côté serveur et le routage optimisé.

Entraînez-vous à créer de petits projets pour consolider vos apprentissages et continuez à explorer l'écosystème React à mesure que vous développez vos compétences. Bon codage !