

TD4 – React

Introduction : installation et utilisation de React

Pour créer et initialiser un projet React, en ligne de commande :

1. tapez `npm init vite@latest app-react -- --template react` (les doubles `--` ne sont pas une faute de frappe)
2. `cd app-react`
3. `npm install`
4. `npm run dev`

Attention ! Sur les machines de la PPTI, il faut systématiquement saisir non pas `npm`, mais `/bin/npm`, et `/bin/node` au lieu de `node`.

Par exemple, vous taperez `/bin/npm init vite@latest app-react -- --template react`

Cela lancera un serveur qui fera tourner React. Vous avez alors accès à quelques commandes, dont vous pourrez consulter la liste en tapant `h`. Les deux plus utiles sont `o`, qui ouvre une fenêtre de votre navigateur sur le site sur lequel vous travaillez, et `q` qui arrête le serveur.

Le fichier `index.html` contient un appel vers `src/main.jsx`, qui lui-même appelle le composant stocké dans `src/App.jsx`. Vous pouvez ensuite soit modifier le composant `App` par défaut, soit en créer un autre que vous appellerez dans `main.jsx`.

1. Créer un mini-projet nommé `react-intro`
 2. Créer un composant nommé `Test`, dans un fichier `Test.jsx`. Ce composant affiche uniquement le paragraphe
- ```
<p>Ceci est un paragraphe.</p>
```
3. Modifier le fichier `main.jsx` pour qu'il appelle et affiche le composant `Test`
  4. Ajouter dans `Test` une propriété `texte` qui permet d'ajouter une phrase au paragraphe
  5. Dans `main.jsx`, utiliser cette propriété pour obtenir le code HTML suivant :

```
<p>Ceci est un paragraphe. Et une nouvelle phrase.</p>
```

---

### Exercice 1 – Mon premier composant React : Le jeu de cartes

---

**Q 1.1** On souhaite créer un composant `Cards` qui permet d'afficher le symbole de la carte. Ce composant renvoie un élément `div` possédant la classe `"Cards"`. Son contenu sera la valeur de la propriété `symbol` des `props`.

**Q 1.2** À quoi correspond `props` ?

**Q 1.3** On souhaite créer une variable associée au composant `Card` qui permet de dire si on affiche ou non la carte (variable `affichage` de valeur `visible` ou `hidden` récupérée des propriétés). Quelle est la meilleure façon de faire ? Écrire le code permettant de créer cette variable. Pour le moment, ne pas gérer cet affichage conditionnel.

**Q 1.4** Modifier le code afin de conditionner l'affichage du composant selon la règle suivante : si l'affichage est `'visible'`, alors on affiche le symbole, sinon on affiche `"-"`.

**Q 1.5** On souhaite rendre le code un petit peu interactif. Quand on clique sur une carte retournée (affichage `"-"`), on doit pouvoir changer son état et afficher son symbole. Modifier le code du composant `Card`.

---

### Exercice 2 – Les listes en React

---

**Q 2.1** Créer un composant `Cardlist` qui permet d'afficher 4 cartes. La première doit être visible, les trois suivantes cachées.

**Q 2.2** L'état du composant `Cardlist` stocke une liste de cartes `cartes`. Modifier le composant `Cardlist` pour ajouter son état, et l'utiliser pour afficher la liste des cartes. Pour cela, vous utiliserez la méthode `map()`.

**Q 2.3** Comment ce code est-il utilisé dans une application React pour pouvoir être affiché dans le navigateur ?

---

**Exercice 3 – Formulaire : Création d’une carte**

---

**Q 3.1** Créer le composant qui permet de créer une carte avec les champs suivants (il s’agit dans cette question de donner uniquement le code JSX, pas le code JavaScript permettant de créer effectivement une nouvelle carte) :

- Symbole (text)
- Affichage (menu déroulant)
- Bouton de soumission

De la même manière qu’on ne peut pas utiliser directement `class` pour l’attribut HTML du même nom dans le JSX, mais qu’on est obligé d’utiliser `className`, on ne peut pas utiliser directement `for`, mais plutôt `htmlFor`.

**Q 3.2** Que faut-il faire pour avoir une page qui affiche la liste des cartes et le formulaire de création de cartes ?