Instructions

1 .

Nous utiliserons des objets avec des états pour créer un formulaire de saisie.

La variable d'état local `profile` et la fonction de définition d'état `setProfile` sont chargées de garder la trace des valeurs d'entrée de nos utilisateurs. Dans notre JSX, nous recherchons les propriétés stockées dans l' objet `profile`. Cela génère une erreur lors de notre premier rendu car nous essayons d'obtenir la valeur d'une propriété à partir d'un objet qui n'a pas encore été défini.

Pour résoudre ce problème, initialisez `profile` en tant qu'objet vide.

2 .

Vous devriez maintenant voir le formulaire rendu, mais rien ne se passera lorsque nous taperons dans les zones de saisie. Notre formulaire ne s'affiche pas encore pour afficher les frappes au clavier.

Pour résoudre ce problème, ajoutez l' écouteur d'événements `onChange` à nos balises JSX afin d'appeler `handleChange()` chaque fois qu'un utilisateur tape dans un champ de saisie. De cette façon, nous pouvons déterminer ce qui se passe lorsque l'utilisateur modifie la saisie en tapant dans le formulaire.

3.

Rendons notre fonction `handleChange()` un peu plus facile à lire. Utilisez la déstructuration d'objets pour initialiser `name` et `value` de manière plus concise.

Il y a un bug dans notre code ! L'avez-vous remarqué ? Essayez de saisir une entrée, puis saisissez une autre entrée. Ce qui se produit? Pourquoi?

Chaque fois que nous appelons notre gestionnaire d'événements `setProfile()`, nous donnons la valeur `profile` d'un nouvel objet avec le `name` et `value` de l'entrée qui a récemment changé, mais nous perdons les valeurs qui ont été stockées pour les entrées avec n'importe quel autre `name`.

Utilisez l'opérateur spread pour corriger ce bug. Nous souhaitons copier toutes les valeurs de notre objet précédent chaque fois que nous appelons notre fonction de définition d'état. À utiliser `prevProfile` comme argument pour notre fonction de rappel de définition d'état.

5 .

Enfin, ajoutez un écouteur d'événement à la balise `<form>` pour appeler notre fonction `handleSubmit()` lorsque l'utilisateur soumet le formulaire.