React JS

Exercices Props et Composants

Exercice 1: Créer un composant simple

Créez un composant nommé SimpleComponent qui rend un élément avec le texte "Hello, World!".

Exercice 2: Utilisation de props dynamiques

Créez un composant Greeting qui prend une prop name et rend un élément avec le texte "Bonjour, [name] !". Le composant App doit utiliser l'état (state) pour dynamiquement changer la valeur de la prop name en fonction de ce que l'utilisateur entre dans un champ de saisie (input).

Exercice 3: Liste de noms

Créez un composant NameList qui prend une prop names (un tableau de noms) et rend une liste <ul> de ces noms. Le tableau de noms sera saisi en 'dur' directement dans App. Et App doit rendre le composant NameList.

Exercice 4: Condition de rendu

Objectif : Dans cet exercice, vous allez créer un composant nommé ConditionalRender qui affiche un message différent en fonction de la réponse de l'utilisateur à une question.

Dans le composant App, posez la question suivante à l'utilisateur : "Aimez-vous React ?" et fournissez un champ de saisie pour qu'il puisse répondre avec "Oui" ou "Non".

Stockez la réponse de l'utilisateur dans l'état local du composant App.

En fonction de la réponse de l'utilisateur, passez une prop condition au composant ConditionalRender. Si la réponse de l'utilisateur est "Oui" (indépendamment de la casse), la prop condition doit être true, sinon elle doit être false.

Dans le composant ConditionalRender, utilisez la prop condition pour rendre le bon message :

Si condition est true, affichez "Vous aimez React !"

Si condition est false, affichez "Vous n'aimez pas React."

Contrainte : Vous devez utiliser l'état local pour stocker la réponse de l'utilisateur et la prop condition pour contrôler le rendu du composant ConditionalRender.

Exercice 5: Événements onclick : statistiques du nombre de click

Objectif : Créez un page avec plusieurs boutons sur la page, et afficher un compteur avec le nombre de clique total peu importe le bouton cliqué.

Il faudra plusieurs composants : un composant Section.js qui aura un titre

de niveau 2, un texte et un bouton. Ce bouton sera lui-même un composant. Il faudra également un composant Counter.js. Dans le composant App.js nous aurons 3 composants section et le composant counter.

Le compteur sera affiché en bas à gauche de la page avec du css en inline style.

Le count sera initialisé à 0 dans le state.

Exercice 6: Formulaire contrôlé

Objectif : Utilisation de onChange et de e.target

Créez un composant Form avec un champ de saisie de texte. L'état du champ de saisie doit être stocké dans l'état local du composant App et être affiché en même temps que la saisie de l'utilisateur, dans un paragraphe directement en dessous.

Exercice 7: Passage de fonctions en tant que props

Créez un composant parent ParentComponent qui rend un composant enfant ChildComponent. Le parent doit avoir une fonction handleClick qui est passée en tant que prop à l'enfant. L'enfant doit avoir un bouton et, lorsqu'il est cliqué, il doit appeler la fonction handleClick du parent. Vous pouvez afficher un message à l'écran au clic, ou dans la console.

Exercice 8: Gestion d'une liste dynamique

Objectif : Dans cet exercice, vous allez créer une application qui permet d'ajouter des éléments à une liste de manière dynamique

Créez un composant List qui prend un tableau d'éléments en tant que prop et rend une liste non ordonnée (ul) qui affiche chaque élément sous forme d'éléments de liste (li).

Créez un composant Button qui prend une fonction de clic (onClick) et un libellé (label) en tant que props. Ce composant rend un bouton qui appelle la fonction de clic lorsqu'il est cliqué.

Créez un composant Title qui prend un texte en tant que prop et rend un titre (h1).

Dans le composant App, initialisez un état local avec un tableau d'éléments (items) contenant quelques éléments par défaut.

Créez une fonction addItem dans le composant App qui utilise la méthode setState pour ajouter un nouvel élément à la liste.

Utilisez les composants Title, Button et List dans le composant App pour créer l'interface utilisateur suivante :

Un titre "Liste Dynamique".

Un bouton "Ajouter un élément".

Une liste affichant les éléments actuels.

L'utilisateur doit pouvoir ajouter des éléments avec un prompt.

Exercice 9 : Application complète : Gestionnaire de tâches

Objectif : Créez une application simple qui combine plusieurs des concepts précédents. Vous allez donc créer une application de gestion de tâches. L'application doit permettre à l'utilisateur d'ajouter, supprimer et marquer des tâches comme complétées. Elle doit afficher des compteurs avec le nombre de taches actuelles totales, le nombre de taches terminées et le nombre de taches restants à faire.

Créez un composant TaskList qui doit retourner une liste de taches sous forme de tableau. Chaque objet de tâche doit avoir une clé qui sera son id, le texte de la tâche, un statut (complétée ou non) et une prop deleteTask.

Créez un composant Task qui prend une tâche en tant que prop et rend un élément de liste (li) avec une case à cocher pour marquer la tâche comme complétée et un bouton pour supprimer la tâche.

Créez un composant TaskForm qui permet à l'utilisateur d'ajouter une nouvelle tâche. Il doit avoir un champ de texte et un bouton "Ajouter".

Créez un composant App qui gère l'état local des tâches. Il doit avoir un tableau de tâches comme état initial. Dans ce tableau, il y aura un objet avec un id, un text et une propriété "completed" initialisé à false par défaut.

Implémentez les fonctions suivantes dans le composant App :

addTask(text) : Ajoute une nouvelle tâche à la liste.

deleteTask(id) : Supprime une tâche de la liste.

toggleTaskStatus(id) : Change le statut de complétion d'une tâche.

Utilisez les composants TaskList, Task, et TaskForm dans le composant App pour créer l'interface utilisateur suivante :

Une liste de tâches.

Chaque tâche doit avoir une case à cocher pour marquer la tâche comme complétée.

Chaque tâche doit avoir un bouton pour la supprimer.

Un formulaire pour ajouter de nouvelles tâches.

Ensuite vous devez avoir un composant TaskCounter qui doit retourner 3 paragraphes avec les 3 compteurs, un sur le nombre de taches totales, un nombre sur les taches complétées, et un nombre sur les taches restantes.

Exercice 10 : Application complète : Application pour Vidéothèque

Créez une application pour gérer une vidéothèque. On devra pouvoir ajouter des vidéos, ces vidéos auront un type (film, dessin animé, documentaire etc…), une année de sortie, un réalisateur, un genre (action, comédie, etc…), un pays, une durée. Et une liste des principaux acteurs.

Tout cela sera dans un tableau, l'utilisateur pourra créer, modifier, supprimer. Il pourra donc ajouter un film par exemple, avec son type, son année de sortie, son réalisateur, son genre, sa durée et une liste d'acteur principaux.

Il doit également y avoir une case à cocher pour qu'on puisse savoir si on l'a vu ou non. On ne pourra pas ajouter un film déjà existant. Il faudra donc comparer ce que l'utilisateur souhaite entrer et ce qui existe déjà dans notre tableau.