Faites passer une librairie jQuery vers React

WealthHealth

Récapitulatif des principales tâches

- Convertir l'ensemble du projet HRNet en React.
- Convertir l'un des quatre plugins jQuery actuels en React.
- Remplacer les 3 plugins jQuery restants par des composants React.
- Effectuer des tests de performance Lighthouse en comparant l'ancienne et la nouvelle application.

Convertir l'ensemble du projet HRNet en React

- Nouvelle version des pages "Create Employee" et "Employee List".
  - Formulaire de création d’employé
  - Liste des employés
- Système de gestion d'état.
  - Stocker le résultat des requêtes
- Cohérence au niveau du style.
- Création d’un fausse api asynchrone.
  - getEmployees
  - saveEmployee

Pages "Create Employee" et "Employee List".

- Utilisation de React Router pour gérer les routes de l'application.
  - C'est le package recommandé par React pour cette tache.
- Utilisation de React Hook Form
  - Package très utilisé et bien maintenu. Performant, flexible et extensible.
  - Avantages :
    - Validation des champs
    - Moins de code à écrire
    - Environ 2 175 525 de téléchargements par semaine

Système de gestion d'état. Plusieurs types d'état

- Les états du formulaire sont gérés par le package React Hook Form.
- L'état de la fenêtre modale est géré par le hook useState.
  - Création d'un état « value » : booléen et d'une fonction toggle.
  - Il n'y pas de raison pour que cet état soit global.
- L'état de la requête est géré par React Query.
  - La meilleur façon, selon l'équipe react, de gérer l'état des requêtes asynchrones, est d'utiliser un framework tel que celui-ci. React query se charge de garder les donnés du client synchronisé avec le serveur.
  - Gestion du cache, mise à jour de la requête basé sur la réponse du serveur en cas de mutation.
  - Bien maintenu. Environ 1 478 806 de téléchargement par semaine.
  - Code beaucoup plus déclaratif. Ne plus gérer manuellement l'état des promesses. Le développeur se concentre sur le code métier.

Cohérence au niveau du style

- Utilisation de sass
- Quelques petites améliorations par rapport au design actuel

Conversion du plugin de table de données

- Pourquoi ce choix ?
  - C'est le plugin qui contient le plus de fonctionnalités.
- Analyse du plugin jquery
  - Fonctionnalités :
    - Filtrage
    - Pagination
    - Tri
  - Usage :
    - Il faut fournir un objet avec les labels et les données
- Le composant react réalise ces fonctionnalités avec le même usage.

Fonctionnement du plugin de table de données

- Le plugin ne fait qu'une chose
  - Il affiche sous forme de tableau les données passées en props et ajoute des fonctionnalités
  - Il ne gère pas la récupération de donnée.
- Utilisation de context API et de useReducer pour la gestion des états
  - Système de gestion d'état de react.
  - Implique moins de dépendance à des packages externes.
- Création du package et publication sur npm
  - Documentation et exemple inclus.

Plugin de sélection de date

- Simple utilisation de l'api du navigateur.
  - Utilisation de <input type="date">
- Aucun besoin de créer un composant spécial.
  - Gain de performance. Bundle plus petit.

Plugin : fenêtre modal

- Un simple composant <Modal></Modal>
  - Prend en props :
    - Un titre
    - Un état (booléen)
    - Une fonction, optionnelle, pour fermer la modale
    - Des enfants
- Utilisation d'un hook qui renvoi un état et une fonction pour changer cette état.

Plugin : menu déroulant

- Création d'un composant <Select />
  - Props :
    - name: le tag «for» de la balise « label » et l'id de la balise « select »
    - label : le text du label
    - options : un tableau contenant les différents choix

Test de performance lighthouse

- Pour la page « create employee »
  - Jquery = score 94, First Contentful Paint 0,7s
  - React = score 100, First Contentful Paint 0,3s
- Pour la page « employee list »
  - Jquery = score 100, First Contentful Paint 0,6s
  - React = score 100, First Contentful Paint 0,4s

Conclusion :

- Léger gain de performance