TP 1: Premiers pas dans NodeJS

Dr Mad généré par une IA

Objectifs

- Création et importation de modules
- Travailler avec le système de fichiers NodeJS
- Création d'une application de terminal
- Interagir avec les fichiers JSON

Préparation du projet

Initialisation avec npm

Naviguez vers le dossier où vous souhaitez créer votre nouveau projet à l'aide de votre terminal ou ligne de commande.
Créez un nouveau répertoire pour votre projet :

mkdir [nom_du_projet]
cd [nom_du_projet]

Pour initialiser un nouveau projet npm, tapez la commande suivante :

npm init

Répondez aux questions qui s'affichent pour configurer votre package.json. Si vous n'êtes pas sûr de certaines réponses, appuyez simplement sur "Entrée" pour utiliser les valeurs par défaut.

Installation des dépendances

Installez les modules suivants à l'aide de npm :

- chalk: Pour ajouter des couleurs à la sortie de la console.
- yargs: Pour analyser les arguments de ligne de commande.

Utilisez la commande suivante pour les installer :

npm install chalk yargs

Création de la structure du projet

Créez un fichier JavaScript nommé drmad.js à la racine du projet.
Créez ensuite les dossiers suivants à la racine de votre projet:

- data
- inputs
- modules

Utilisez ces commandes pour créer les dossiers et le fichier :

touch drmad.js
mkdir data inputs models

Vérification

Votre structure de projet devrait ressembler à ceci :

```
[nom_du_projet]/
├── data/
├── inputs/
├── models/
├── drmad.js
└── package.json
```

Assurez-vous que les dépendances sont bien répertoriées dans votre fichier package.json sous la section dependencies.

Description des données sur les virus mortels

Dans cette section, nous allons traiter des données liées à différents virus mortels. Notre objectif est de créer un terminal de ligne de commande permettant d'ajouter, de sélectionner tous les éléments, de sélectionner un élément par son identifiant, et de supprimer un élément par son identifiant dans un fichier JSON.

Structure JSON
Les objets du fichier JSON contiennent les propriétés suivantes :

- **_id**: L'identifiant unique de l'élément.
- **links**: Un tableau contenant des liens vers des ressources externes, typiquement vers Wikipedia pour des informations supplémentaires sur le virus.
- **name**: Le nom du virus.
- **description**: Une courte description ou un autre nom pour le virus.
- **stock**: Le nombre d'unités en stock.
- **wait**: La date à laquelle l'élément a été ajouté ou mis à jour pour la dernière fois.
- **sold**: Un booléen indiquant si l'élément a été vendu ou non.
- **price**: Le prix de l'élément.
- **promotion**: Un tableau contenant des objets qui décrivent les promotions disponibles. Chaque promotion a un **_id**, un **discount** indiquant le pourcentage de réduction et un amount indiquant le nombre d'articles nécessaires pour obtenir cette réduction.
- **object**: Un objet contenant le code génétique du virus.

Modèle JSON vide

Voici un modèle JSON vide que vous pouvez utiliser comme point de départ pour ajouter de nouveaux éléments :

```
{
  "_id": "",
  "links": [],
  "name": "",
  "description": "",
  "stock": 0,
  "wait": {
    "$date": ""
  },
  "sold": false,
  "price": 0,
  "promotion": [],
  "object": {
    "code": ""
  }
}
```

Création et utilisation du modèle JSON pour l'ajout d'articles

Pour faciliter l'ajout de nouveaux éléments à votre base de données, vous allez travailler avec un modèle JSON. Ce modèle vous permettra de remplir les informations nécessaires directement dans le fichier, plutôt que de les saisir un par un dans la ligne de commande.

Étapes pour créer le modèle JSON :

1. Ouvrez votre éditeur de texte ou IDE préféré.
2. Copiez le modèle JSON vide fourni précédemment.
3. Créez un nouveau fichier dans le dossier inputs de votre projet et nommez-le item.json.
4. Collez le modèle JSON dans ce fichier et enregistrez-le.

Ajouter un nouvel élément à l'aide du modèle :
Chaque fois que vous souhaitez ajouter un nouvel élément de virus :

1. Ouvrez le fichier item.json dans le dossier inputs.
2. Remplissez les champs nécessaires en suivant la structure du modèle. Assurez-vous de ne pas modifier les noms des clés, mais seulement leurs valeurs.
3. Une fois que vous avez saisi toutes les informations nécessaires, enregistrez le fichier item.json.

Utilisation de la ligne de commande pour ajouter l'élément :
Après avoir rempli les informations dans le modèle item.json, vous pouvez ajouter cet élément à votre base de données en utilisant la ligne de commande :

node drmad.js add --file=./inputs/item.json

Cette commande indique à votre script drmad.js de chercher le fichier **item.json** dans le dossier **inputs** et d'ajouter les informations contenues dans ce fichier à votre base de données.

En suivant cette méthode, vous pouvez facilement et systématiquement ajouter de nouveaux éléments sans avoir à saisir manuellement chaque détail dans la ligne de commande. Cela garantit également une plus grande cohérence et réduit le risque d'erreurs lors de la saisie.

Exemples de données à ajouter

Pour mieux comprendre comment remplir le modèle JSON, vous pouvez vous inspirer de ces exemples pour ajouter vos propres entrées.

_id	links	name	description	stock	wait	sold	price	promotion	object
1	Variole - Wikipedia	variole	variole	0	2023-05-23T16:47:00.354Z	false	750000	[]	`{ "code": "ACBCBBACADDCADDBCDDCABBBCADDCADDBCD"}`
2	Prion - Wikipedia	prion	prion	1	2023-05-23T16:47:00.353Z	true	100000	[{"_id":"1","discount":10,"amount":2}]	`{ "code": "ABCACACABCAB"}`

3	Ebola - Wikipedia	ebola	ebola	0	2023-05-23T16:47:00.354Z	true	650000	[]	{ "code": "AAAABBBBCCCCDDDDDDDDCCCCBBBBAAAA"}
4	Adenovirus - Wikipedia	adeno	adenovirus	10	2023-05-23T16:47:00.348Z	true	50000	[{"_id":"2","discount":5,"amount":2},{"_id":"3","discount":10,"amount":5}]	{ "code": "ABABABABABAB"}
5	COVID-19 - Wikipedia	covid	covid19	50	2023-05-23T16:47:00.352Z	true	100000	[{"_id":"4","discount":10,"amount":10},{"_id":"5","discount":20,"amount":50}]	{ "code": "ABADBADCCCBADCBABADBADCCCBADCB"}
6	Staphylocoque - Wikipedia	staphy	staphylocoque	100	2023-05-23T16:47:00.352Z	true	500	[{"_id":"6","discount":5,"amount":5},{"_id":"4","discount":10,"amount":10},{"_id":"5","discount":20,"amount":50}]	{ "code": "ABBCDDDDCBBB"}

Description du projet : Application NodeJS pour gérer une base de données de virus

1. Structure de l'application :

L'application NodeJS que vous allez développer se compose principalement d'un fichier principal drmad.js qui utilisera la bibliothèque yargs pour gérer les commandes. Les fonctions associées à chaque commande seront stockées dans le module items.js du dossier models. Ces fonctions vont interagir avec un fichier **data.json** dans le dossier data pour stocker, lire ou supprimer des éléments.

2. Commandes à implémenter :

- Ajouter (add) :
    - Argument : --file
    - Cette commande doit lire le fichier spécifié (e.g., **./inputs/item.json**) et ajouter son contenu à **data.json** s'il est valide.
    - Vérification : Avant d'ajouter un élément, vérifiez que l'ID n'existe pas déjà dans data.json.
- Lister (list) :
    - Pas d'arguments nécessaires.
    - Cette commande doit afficher tous les éléments présents dans **data.json**.
- Obtenir (get) :
    - Argument : --id
    - Cette commande doit rechercher et afficher l'élément ayant l'ID spécifié dans data.json.
- Supprimer (delete) :
    - Argument : --id
    - Cette commande doit retirer l'élément ayant l'ID spécifié de data.json.

3. Module items.js :

Ce module, placé dans le dossier **models**, contiendra les fonctions pour :

- Lire data.json.
- Ajouter un élément à data.json.
- Lister tous les éléments de data.json.
- Obtenir un élément spécifique de data.json basé sur son ID.
- Supprimer un élément spécifique de data.json basé sur son ID.

4. Vérification du code du virus :

Avant d'ajouter un virus à la base de données, vous devez valider le code génétique du virus. Ce code ne doit contenir que les lettres A, B, C et D. Si ce n'est pas le cas, une erreur est affichée à l'utilisateur et le virus n'est pas ajouté.

5. Flux de travail pour l'ajout d'un élément (commande add) :

- Ouvrez **data.json** et chargez son contenu.
- Ouvrez le fichier spécifié avec l'argument **--file**.
- Extraire l'ID de l'élément à ajouter et vérifier s'il existe déjà dans **data.json**.
- Si l'ID n'existe pas, validez le code génétique du virus.
- Si le code est valide, ajoutez le nouvel élément à data.json.
- Enregistrez le fichier data.json.

6. Lister tous les éléments (commande list) :

- Ouvrez le fichier data.json et chargez son contenu.
- Affichez chaque élément (virus) avec ses propriétés (ID, nom, description, etc.) dans la console.

7. Obtenir un élément spécifique (commande get) :

- Ouvrez le fichier data.json et chargez son contenu.
- Recherchez l'élément ayant l'ID spécifié (fourni avec l'argument --id).
- Si l'élément est trouvé, affichez ses propriétés dans la console.
- Sinon, affichez un message d'erreur indiquant que l'élément avec cet ID n'a pas été trouvé.

8. Supprimer un élément spécifique (commande delete) :

- Ouvrez le fichier data.json et chargez son contenu.
- Recherchez l'élément ayant l'ID spécifié (fourni avec l'argument --id).
- Si l'élément est trouvé :
    - Retirez cet élément de la liste.
    - Enregistrez la nouvelle liste (sans l'élément supprimé) dans data.json.
    - Affichez un message de confirmation indiquant que l'élément a été supprimé avec succès.
- Si l'élément n'est pas trouvé, affichez un message d'erreur indiquant que l'élément avec cet ID n'a pas été trouvé et, par conséquent, n'a pas été supprimé.

À travers cette description, vous avez une vue d'ensemble claire des étapes à suivre pour développer cette application. Vous serez en mesure de mettre en œuvre les commandes, de gérer les erreurs et d'interagir avec le fichier de données tout en appliquant les meilleures pratiques de programmation. Bonne chance !