TP

Initiation à Node.js

V1 - Décembre 2020

TP01 - Création d'un projet Node.js

Ressources

https://docs.npmjs.com/cli/v6/commands/npm

Objectifs

- utiliser votre terminal de commandes,
- créer un dossier helloworld et se placer à sa racine,
- créer un projet Node.js avec le CLI de NPM,
- créer un fichier app.js dans le dossier helloworld (avec le terminal ou l'éditeur de code de votre choix),
- ajouter un script "start" dans le fichier package.json permettant de démarrer le fichier app.js,
- exécuter le projet avec la commande "npm start",
- afficher dans le terminal "Hello World !" avec l'objet console de JavaScript,

TP02 - Hot Reloading

Ressources

https://nodemon.io/

Objectifs

- importer le module NPM "Nodemon" dans les dépendances de développement,
- ajouter un script "dev" dans le fichier package.json permettant de démarrer le fichier app.js avec nodemon,
- exécuter le projet avec la commande "npm run dev",
- afficher dans le terminal "Hello World !",
- modifier la valeur retournée par l'objet console,
- vérifier que le texte affiché dans le terminal a été mis jour en cohérence avec la modification,

TP03 - Variables d'environnement

Ressources

https://nodejs.dev/learn/how-to-read-environment-variables-from-nodejs

Objectifs

- dans le script start du fichier package.json créer une variable d'environnement nommée ENV avec la valeur prod,
- dans le script dev du fichier package.json créer une variable d'environnement nommée ENV avec la valeur dev,
- afficher dans le terminal, la valeur de la variable d'environnement ENV depuis le fichier app.js,
- vérifier que la valeur affichée correspond à la valeur attendue lorsque vous démarrez le projet avec la commande "npm start" et avec la commande "npm run dev",
- avec l'objet console, afficher "hello dev" si la variable d'environnement est égale à "dev" et "hello prod" si la variable d'environnement est égale à "prod",

TP04 - import / export de modules (syntaxe ES5)

Ressources

https://nodejs.org/api/modules.html

Objectifs

- créer un répertoire "utils" à la racine du répertoire courant,
- créer un fichier "say.js" dans le répertoire "utils",
- dans le fichier say.js, créer une méthode "hello" acceptant un paramètre "who",
- la méthode "say" doit permettre d'afficher dans la console le texte "Hello " + la valeur de la variable "who",
- exporter cette méthode depuis le fichier say.js,
- importer la méthode hello du module say.js depuis le fichier app.js,
- à l'aide de la méthode hello importée, afficher "Hello Node.js" dans le terminal,

TP05 - import / export multiples et affectation par décomposition

Ressources

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Op%C3%A9rateurs/Affecter_par_d%C3%A9composition

Objectifs

- dans le fichier say.js, créer une méthode "bonjour" acceptant un paramètre "qui",
- la méthode "bonjour" doit permettre d'afficher dans la console le texte "Bonjour " + la valeur de la variable "qui",
- exporter à la fois la méthode hello et la méthode bonjour,
- dans le fichier app.js importer les méthodes hello et bonjour avec la syntaxe de décomposition ES6+,
- afficher dans le terminal "Hello World" et "Bonjour tout le monde" en utilisant les méthodes hello et bonjour,

TP06 - POO ES6+

Ressources

- https://nodejs.org/api/modules.html
- https://medium.com/developers-writing/fibonacci-sequence-algorithm-in-javascript-b253dc7e320e
- https://fr.wikipedia.org/wiki/Suite_de_Fibonacci
- https://dev.to/orighoprecious/object-oriented-programming-in-javascript-es5-es6-explained-4jbk

Objectifs

- créer un répertoire "classes" à la racine du répertoire courant,
- créer un fichier "Robot.js" dans le répertoire "classes",
- dans le fichier "Robot.js" créer une classe Robot avec la syntaxe ES6+,
- exporter la classe,
- la classe doit disposer d'un attribut privé "ref",
- le constructeur de classe doit recevoir un paramètre "ref" permettant de renseigner l'attribut "ref" d'une occurence de classe Robot,
- le constructeur doit vérifier que le paramètre ref est fourni, dans le cas contraire il doit lancer une exception,
- lors de l'instanciation d'une occurence, le terminal doit afficher un message "Hi, my name is <ref> !" via son constructeur,
- l'attribut "ref" doit disposer d'accesseurs (getter et setter),
- la classe Robot doit disposer d'un attribut statique "YEAR_OF_CONCEPTION" égale à 2001,
- la classe Robot doit disposer d'une méthode publique "fibonacci" recevant un nombre entier supérieur ou égale à 1 en paramètre,
- vérifier le nombre fourni à la méthode "fibonacci",
- la méthode "fibonacci" calcule et retourne le résultat en suivant la "suite de Fibonacci" à partir du nombre fourni en paramètre,

TP07 - Asynchronisme et parallélisme

Ressources

- https://openclassrooms.com/fr/courses/5543061-ecrivez-du-javascript-pour-le-web/5577651-comprenez-comment-fonctionne-lasynchrone-en-js
- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous/Introducing
- https://nodejs.org/en/docs/guides/event-loop-timers-and-nexttick/
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Concurrence_et_boucle_des_%C3%A9v%C3%A9nements
- https://www.w3schools.com/jsref/met_win_settimeout.asp
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Instructions/async_function
- https://nodejs.org/en/docs/guides/timers-in-node/

Objectifs

A partir de l'exemple fourni et à l'aide des méthodes disponibles dans le fichier ./utils/asynchronous.js, coder la recette des "Spaghetti al pomodoro" ci-dessous.

Pour gérer l'asynchronisme utiliser uniquement :

- setTimeout (cf. méthode "do_after"),
- les méthodes "callbacks",

L'objectif est d'afficher dans le terminal chaque étape de la recette dans l'ordre logique et avec le délai correspondant.

Pour l'exercice, 1mn correspondra à 10s (soit 10000ms), donc 9mn = 9000 ms.

- faire chauffer de l'eau dans une casserole dans pendant 9mn,
- faire chauffer de l'huile d'olive dans une poêle pendant 1mn,
- pendant ce temps, couper de l'ail,
- mettre l'ail dans l'huile d'olive chaude dans la poêle pendant 1mn (en option, ajouter du piment avec l'ail),
- ajouter de la tomate dans la poêle et laisser cuire jusqu'à ce que les spaghetti soient cuits,
- ajouter du sel, du poivre et quelques feuilles de basilic,
- lorsque l'eau est bouillante, mettre du gros sel dans l'eau,
- mettre 500g de spaghetti à cuire pendant 10mn dans la casserole d'eau bouillante, remuer les spaghetti toutes les 2 mn,
- essorer les spaghetti et les verser dans la poêle avec la sauce tomate,
- terminer la cuisson des spaghetti dans la poêle avec la sauce tomate pendant 1mn,
- servir et déguster ;)

TP08 - Asynchronisme avec Promise

Ressources

- https://openclassrooms.com/fr/courses/5543061-ecrivez-du-javascript-pour-le-web/5577651-comprenez-comment-fonctionne-lasynchrone-en-js
- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous/Introducing
- https://nodejs.org/en/docs/guides/event-loop-timers-and-nexttick/
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Concurrence_et_boucle_des_%C3%A9v%C3%A9nements
- https://www.w3schools.com/jsref/met_win_settimeout.asp
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Instructions/async_function

Objectifs

A partir de l'exemple fourni et à l'aide des méthodes disponibles dans le fichier ./utils/asynchronous.js, coder la recette des "Spaghetti al pomodoro" ci-dessous.

Pour gérer l'asynchronisme utiliser uniquement :

- setTimeout (cf. méthode "do_after"),
- Promise,

L'objectif est d'afficher dans le terminal chaque étape de la recette dans l'ordre logique et avec le délai correspondant.

Pour l'exercice, 1mn correspondra à 10s (soit 10000ms), donc 9mn = 90000 ms.

- faire chauffer de l'eau dans une casserole dans pendant 9mn,
- faire chauffer de l'huile d'olive dans une poêle pendant 1mn
- pendant ce temps, couper de l'ail,

- mettre l'ail dans l'huile d'olive chaude dans la poêle pendant 1mn (en option, ajouter du piment avec l'ail),
- ajouter de la tomate dans la poêle et laisser cuire jusqu'à ce que les spaghetti soient cuits,
- ajouter du sel, du poivre et quelques feuilles de basilic,
- lorsque l'eau est bouillante, mettre du gros sel dans l'eau,
- mettre 500g de spaghetti à cuire pendant 10mn dans la casserole d'eau bouillante, remuer les spaghetti toutes les 2 mn,
- essorer les spaghetti et les verser dans la poêle avec la sauce tomate,
- laisser les spaghetti dans la poêle avec la sauce tomate pendant 1mn,
- servir et déguster ;)

TP09 - Asynchronisme avec async / await

Ressources

- https://openclassrooms.com/fr/courses/5543061-ecrivez-du-javascript-pour-le-web/5577651-comprenez-comment-fonctionne-lasynchrone-en-js
- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous/Introducing
- https://nodejs.org/en/docs/guides/event-loop-timers-and-nexttick/
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Concurrence_et_boucle_des_%C3%A9v%C3%A9nements
- https://www.w3schools.com/jsref/met_win_settimeout.asp
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Instructions/async_function

Objectifs

A partir de l'exemple fourni et à l'aide des méthodes disponibles dans le fichier ./utils/asynchronous.js, coder la recette des "Spaghetti al pomodoro" ci-dessous.

Pour gérer l'asynchronisme utiliser uniquement :

- setTimeout (cf. méthode "do_after"),
- les méthodes "callbacks",

L'objectif est d'afficher dans le terminal chaque étape de la recette dans l'ordre logique et avec le délai correspondant.

Pour l'exercice, 1mn correspondra à 10s (soit 10000ms), donc 9mn = 90000 ms.

- faire chauffer de l'eau dans une casserole dans pendant 9mn,

- faire chauffer de l'huile d'olive dans une poêle pendant 1mn
- pendant ce temps, couper de l'ail,
- mettre l'ail dans l'huile d'olive chaude dans la poêle pendant 1mn (en option, ajouter du piment avec l'ail),
- ajouter de la tomate dans la poêle et laisser cuire jusqu'à ce que les spaghetti soient cuits,
- ajouter du sel, du poivre et quelques feuilles de basilic,
- lorsque l'eau est bouillante, mettre du gros sel dans l'eau,
- mettre 500g de spaghetti à cuire pendant 10mn dans la casserole d'eau bouillante, remuer les spaghetti toutes les 2 mn,
- essorer les spaghetti et les verser dans la poêle avec la sauce tomate,
- laisser les spaghetti dans la poêle avec la sauce tomate pendant 1mn,
- servir et déguster ;)

TP10 - Asynchronisme dans une boucle

Ressources

- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Math/random
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise/all

Objectifs

- Dans une boucle de 6 itérations, exécuter à chaque itération une opération asynchrone (via l'instruction setTimeout) avec une durée aléatoire,
- Attendre que toutes les opérations asynchrones soient résolues,
- Trier les opérations par le temps de résolution utile à leur résolution (par ordre croissant),
- Afficher un message de log pour chaque opération indiquant leur temps de résolution,
- Afficher un message de log indiquant la fin du programme,
- Gérer la résolution de plusieurs instructions asynchrones exécutées dans une boucle

TP11 - Interactions avec une API

Ressources

- https://github.com/axios/axios
- https://jsonplaceholder.typicode.com/
- https://jsonplaceholder.typicode.com/guide/

Objectifs

- utiliser le module Axios pour consommer l'API {JSON} Placeholder (https://jsonplaceholder.typicode.com/),
- afficher le titre de la todo dont l'id est égal à 7,
- afficher le nombre total de todos récupérés à l'adresse https://jsonplaceholder.typicode.com/todos,
- créer 1 todo : {title:"Learn Node.js", body:"Learn Node.js in LP Ciasie"}
- afficher le code http obtenu après une requête HTTP Post réussie

TP12 - Gestion d'événements

Ressources

- https://nodejs.org/api/events.html,
- https://fr.wikipedia.org/wiki/SOLID_(informatique)

Objectifs

- Implémenter un système de diffusion d'événements personnalisés en prenant pour exemple l'observation d'un feu tricolore par un piéton, une voiture et une moto,
- Lorsque le feu tricolore change de couleur, les observateurs doivent réagir en cohérence (avancer ou s'arrêter),
- Afficher des logs dans le terminal à chaque changement de couleur du feu tricolore :
    - couleur du feu,
    - réaction des observateurs,
- Exploiter au mieux les bonnes pratiques de la POO pour disposer d'un code "S.O.L.I.D",

TP13 - Design Pattern Observer

Ressources

- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Op%C3%A9rateurs/instanceof
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Classes
- https://subscription.packtpub.com/book/web_development/9781783287314/1/ch01lvl1sec12/the-observer-pattern
- https://dottedsquirrel.com/javascript/observer-pattern/
- https://fr.wikipedia.org/wiki/SOLID_(informatique)

Objectifs

- Implémenter le Design Pattern **"Observer"** du _"Gang of Four"_ en prenant pour exemple l'observation d'un feu tricolore par un piéton, une voiture et une moto,
- Lorsque le feu tricolore change de couleur, les observateurs doivent réagir en cohérence (avancer ou s'arrêter),
- Afficher des logs dans le terminal à chaque changement de couleur du feu tricolore :
    - couleur du feu,
    - réaction des observateurs,
- Exploiter au mieux les bonnes pratiques de la POO pour disposer d'un code "S.O.L.I.D",

Système de fichiers
Fetch / Axios
Serveur Node.js sans module
Socket.io