TEDONZE NICK

Td0 : Rapport de Travail

I- La première Partie du travail

Cette partie nous avions utilisés les commandes les plus basiques de node js pour réaliser chaque tache conformément à la documentation fournis ,
Une petite confusion avec la définition de chemin courant et chemin d ‘exécution mais je suis arrivé à la conclusion que le chemin courant nous avions défini avec __dirname et pour le chemin d’exécution avec process .cwd ()
-
pour le prog5 nous avions réalisés ce qui semble être un benchmark pour évaluer la performance de node js dans l’appel d’un callback lorsque le temps d’attente x est écroulé on se rend compte que l’erreur est de l’ordre de 0,001 soit une bonne précision en évaluant le temps qui s ‘écoule entre l’appel du seTime out (armement du callback) puis le déclenchement du callback par rapport au temps d’attente prévu,
- Le programme 6 écrit en une seule ligne de code affiche notre message de bienvenue dans une boucle infinie lorque on envoi un SIGINTpour interrompre le processus le programme s’arrête brusquement, Pour résoudre ce problème nous allons capturer ce signal et dire quoi faire dans ce cas nous avions affiché « A bientôt » puis nous avions fermer le processus avec la méthode exit()
- la raison est que la fonction setTimeout sera exécutée de façon asynchrone dont les instructions suivantes seront exécutées en attentant qu ‘un événement déclenche le callback pour que l’instruction précédente suivent son cour,
- Pour le programme 11 nous avions utilisés readline pour récupérer le code entrée ligne par ligne te ajouter dans un fichier text

- nous avions réalisés une fonction de copie en quatre lignes grâce au flux d’entré et sortie avec un pipe

-la fonction arg permet de spécifier d’autres variables qui permettront de spécifier le message afficher a l’écran

-Pour le programme 12 qui nous permet de copier nous l’avions fait très rapidement en quatre lignes de code en créant un pipe entre le flux d’entrée et le flux de sortie grâce à « readable.pipe(writable); »
-Pour le programme 13 nous avions puis définir une variable environnement qui sera pris en paramètre par la fonction log de notre programme

II- La deuxième partie pour le logging des informations

Cette partie nous invite à faire deux programmes liés entres eux mais le prog14 à soulever un gros soucis de compréhension bien que cette partie semble être une partie clés de ce TD nous l’avions pas fait

III- ## A la découverte des **Streams**

cette partie très intéressante nous avions réalisés les programmes suivants
-Pour le programme 16 nous avions utilisé la commande « const ls = spawn('ls'); » qui nous retourne la liste des fichiers dans un répertoire que nous écoutons à la sortie et affichons à l’écran

-Pour le programme 17 nous avions fait comme précédemment en combinant « ls- l|wc -l «» mais en utilisant plutôt la méthode « exec(commande) » plus pratique que « spawn() »
-Pour le programme 18 toujours similaire au précédent nous avions fait ecrire une fonction qui prend en paramètre une extension sous forme de String que nous allons concaténer avec une commande qui sera exécuter grâce à la méthode exec et le résultat affiché à l'écran,
-Pour le programme 19 il nous faut récupérer tous les extensions contenues dans le répertoire en listant les fichiers puis en utilisant la méthode split pour récupérer l'extension qui sera ensuite injecter dans la méthode précédente pour effectuer le décompte par extension