MODULE 1 EXERCICE

[**Introduction**](#_9gfdtz9sad3r) **3**

[**Exercices**](#_9iv3quw3p602) **3**

# Introduction

Maintenant que le web, node et les modules async/request/fs n’ont plus de secrets pour vous nous allons pouvoir effectuer des exercices bien stylés !

Attention, pour tout ces exercices, les seuls modules autorisés sont async / request et fs.  
Interdiction d’utiliser les fonction sync de fs.

Pas de callback dans un callback dans un callback, le code doit être beau et mis correctement dans un waterfall.

# Exercices

**Exercice 3.modules.1 :**

1. Récupérez via une requête GET une liste d’url présente à l’adresse <https://stage.optionizr.com/urls/json> exemple : *{"urls":["https://stage.optionizr.com/url/json/1","https://stage.optionizr.com/url/json/2","https://stage.optionizr.com/url/json/30","https://stage.optionizr.com/url/json/40"]}*

1. Pour chacune des urls, effectuez une requête GET permettant de récupérer le résultat associé à l’adresse, exemple : *{"result":10}*
2. Avec une requête POST sur <https://stage.optionizr.com/api/formjson> envoyez l’ensemble des résultats via un objet json, cet url prends tous les paramètres en entrée et les additionne, exemple : input *{“value1”: 10, “value2”: 200}* , retourne *{result: 210}*
3. Affichez le résultat dans la console

*Tips: Waterfall, request.get, async.eachOfLimit, request.post*

**Exercice 3.modules.2 :**

1. En parallèle (avec async parallel) effectuez une requête GET sur <https://stage.optionizr.com/urls/products/1> , une requête GET sur <https://stage.optionizr.com/urls/products/2> et une requête GET sur <https://stage.optionizr.com/urls/products/3>, ces requête renvoi une liste d’url de produits :*{"urls":["*[*https://stage.optionizr.com/url/product/vache","https://stage.optionizr.com/url/product/chevre","https://stage.optionizr.com/url/product/cheval*](https://stage.optionizr.com/url/product/vache%22,%22https://stage.optionizr.com/url/product/chevre%22,%22https://stage.optionizr.com/url/product/cheval)*"]}*
2. Toujours dans le parallel, pour chacune des urls, effectuez une requête GET sur l’adresse pour récupérer un nombre associé à chacun des éléments, exemple sur [*https://stage.optionizr.com/url/product/cheval*](https://stage.optionizr.com/url/product/vache%22,%22https://stage.optionizr.com/url/product/chevre%22,%22https://stage.optionizr.com/url/product/cheval) *: {"number":20}*
3. Toujours dans le parallel, créez un fichier par élément contenant number ligne avec le nom de l'élément, exemple fichier cheval :  
     
   cheval  
   cheval  
   cheval  
   …

**Exercice 3.modules.3:**

1. En parallel effecutez une requête POST sur <https://stage.optionizr.com/api/formpost1> et <https://stage.optionizr.com/api/formpost2> ces deux urls prennent un paramètre du form urlencoded (form html standard, header : Content-Type: application/x-www-form-urlencoded), effectuent des somme, multiplication avec les input puis affiche le résultat dans une page html. Exemple : input cheval=3 , chevre=2 , affiche une page html contenant 5.
2. Extraire le résultat de chacun des body html récupérés : *<span class="result">****5****</result>*, et additionnez les 2, créez ensuite X fichiers (ou x est le résultat additionné) vides
3. Séquentiellement (les uns après les autres), remplir les fichiers avec le lorem ipsum, le premier fichier contient un mot, le deuxième 2 mots, etc...:   
     
   *Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.*