

420KBG – Laboratoire 1 (individuel)

- Télécharger le cadriciel API-Server du répertoire <https://github.com/Nicolas-Chourot/API-Server>
- Ajoutez un contrôleur MathsController dérivant de la classe Controller qui offrira les fonctionnalités suivantes à travers le seul point d'entrée GET :

Fonction	Exemple d'appel
Documentation d'utilisation en html (voir exemple en annexe)	api/maths?
Soit deux nombres x et y , calculer et retourner $x + y$	api/maths?op=+&x=50&y=25
Soit deux nombres x et y , calculer et retourner $x - y$	api/maths?op=-&x=50&y=25
Soit deux nombres x et y , calculer et retourner $x \times y$	api/maths?op=*&x=50&y=25
Soit deux nombres x et y , calculer et retourner $\frac{x}{y}$	api/maths?op=/&x=50&y=25
Soit deux nombres x et y , calculer et retourner $x\%y$	api/maths?op=%&x=50&y=7
Soit un entier n , calculer et retourner $n!$	api/maths?op=!&n=5
Étant donné un entier n , retourner <code>true</code> seulement si n est premier	api/maths?op=p&n=5
Étant donné un entier n , retourner la valeur du n ième premier	api/maths?op=np&n=5

Notez que chaque fonction est paramétrée par un code d'opération (p.ex. : `+` pour la somme, `!` pour la factorielle, `p` pour n ième premier, ...) et par des paramètres nommés (x , y , n) comme le veut l'usage dans de tels services.

- Héberger votre version d'API-Server sur Glitch (le lien vers le service devra être de la forme suivante : [https://\[votre NAD-API-Server\].glitch.me/api/maths](https://[votre NAD-API-Server].glitch.me/api/maths))

Décoder des paramètres

Notez que le membre d'instance de la classe Controller possède un membre `HttpContext.path.params` avec l'url de requête

GET : `[host]/api/maths?op=/&x=123&y=5`

contiendra l'objet

```
{op: '/', x: '123', y: '5'}
```

Note importante : Le protocole http traite le symbole `+` en le remplaçant par un espace.

Alors si `op = '+'` considérez que c'est l'opération `+`.

Tests à réaliser

Vous devez rapporter au moins les problèmes suivants dans la réponse (de format JSON) :

- Liste de paramètres incorrecte :
 - paramètres manquants,
 - paramètres en trop,
 - opération inexistante,
- Types incohérents. Notez que JavaScript n'est pas un langage fortement typé, mais il y a tout de même des enjeux dont il vous faudra tenir compte, par exemple le cas où on attendrait un nombre mais on recevrait quelque chose qui ne se convertit pas en nombre, ou encore le cas où on attendrait un entier mais où le nombre fourni ne serait pas convertible en entier sans perte.

Notez qu'il existe une liste de code de succès ou d'erreur [http¹](http://en.wikipedia.org/wiki/List_of_HTTP_status_codes), parmi lesquels 422 peut servir à signaler des paramètres incorrects.

Exemples :

Avec [/api/maths?op=p&n=6](#) on obtient dans le corps de la réponse

```
{
  "n": "6",
  "op": "p",
  "value": false
}
```

Avec [/api/maths?op=*&x=10&y=abc](#) on obtient dans le corps de la réponse

```
{
  "op": "*",
  "x": "10",
  "y": "abc",
  "error": "'y' parameter is not a number"
}
```

Avec [/api/maths?op=+&x=3&y=2](#) ou [/api/maths?op=&x=3&y=2](#) on obtient dans le corps de la réponse

```
{
  "op": "+",
  "x": "3",
  "y": "2",
  "value": 5
}
```

etc.

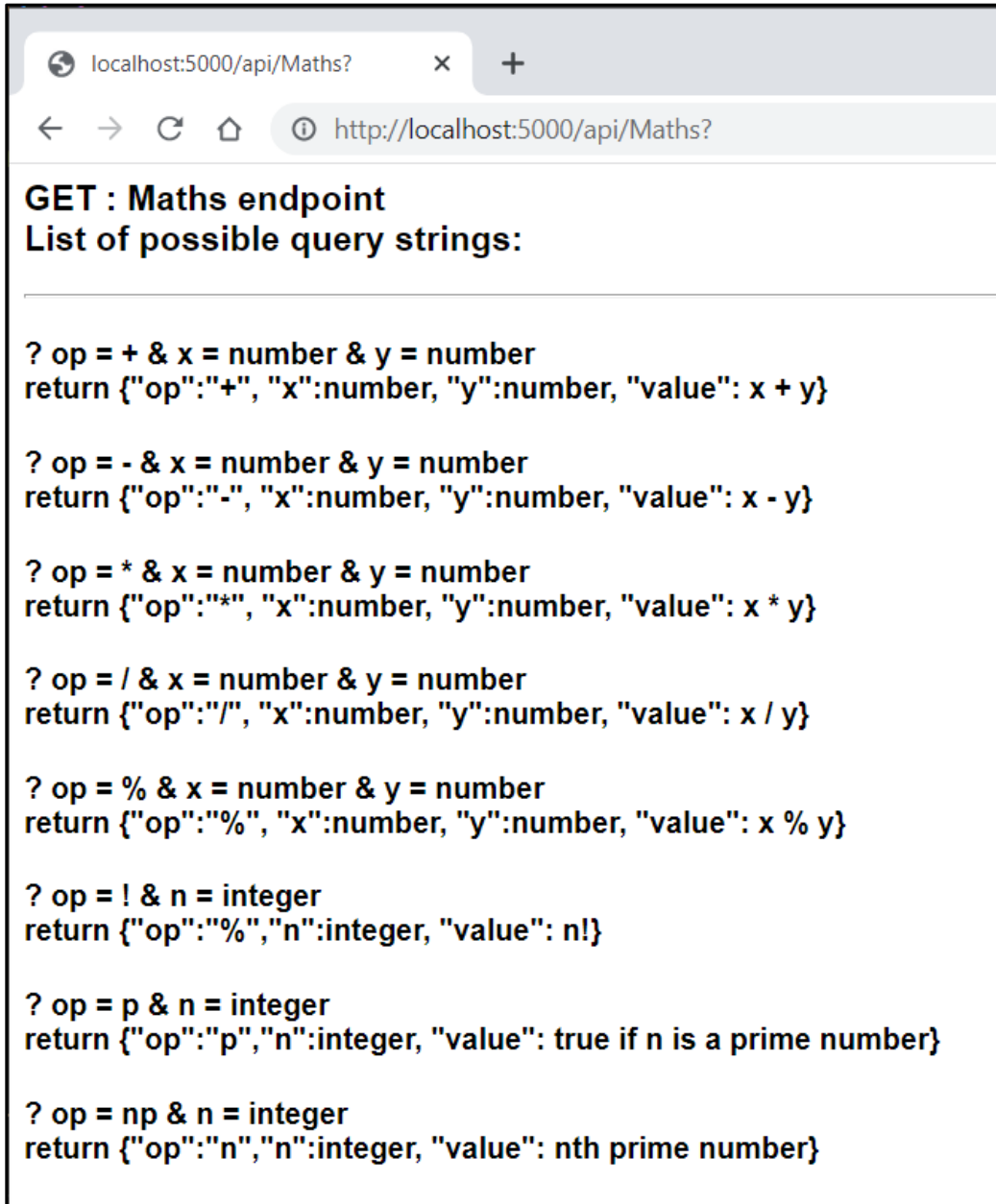
¹ https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

Modalités de remise

Organisation	Travail individuel
Date de remise	Jeudi le 15 septembre avant minuit (au retard accepté)
Remise par Colnet	Une archive zip nommée comme suit : Laboratoire-1.zip Cette archive doit contenir vos fichiers sources Node.js (moins le répertoire node_modules) ainsi qu'un fichier nommé glitch.txt contenant l'hyperlien vers votre serveur hébergé sur Glitch par exemple : https://202212345-API-Server.glitch.me/api/maths

Note : Les rencontres du 9 et 12 septembre seront dédiées à ce laboratoire.

Amusez-vous bien!

Annexe**Exemple de réponse à /api/maths?**

The screenshot shows a web browser window with the address bar displaying 'localhost:5000/api/Maths?'. The page content is as follows:

GET : Maths endpoint
List of possible query strings:

? op = + & x = number & y = number
return {"op":"+", "x":number, "y":number, "value": x + y}

? op = - & x = number & y = number
return {"op":"-", "x":number, "y":number, "value": x - y}

? op = * & x = number & y = number
return {"op":"*", "x":number, "y":number, "value": x * y}

? op = / & x = number & y = number
return {"op":"/", "x":number, "y":number, "value": x / y}

? op = % & x = number & y = number
return {"op":"%", "x":number, "y":number, "value": x % y}

? op = ! & n = integer
return {"op":"%", "n":integer, "value": n!}

? op = p & n = integer
return {"op":"p", "n":integer, "value": true if n is a prime number}

? op = np & n = integer
return {"op":"n", "n":integer, "value": nth prime number}