

## 420KBG – Laboratoire 2 (Équipe de deux)

- Télécharger le cadriciel API-Server du répertoire <https://github.com/Nicolas-Chourot/API-Server-beta-1.85>
- 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)	<a href="#">api/maths?</a>
Soit deux nombres $x$ et $y$ , calculer et retourner $x + y$	<a href="#">api/maths?op=+&amp;x=50&amp;y=25</a>
Soit deux nombres $x$ et $y$ , calculer et retourner $x - y$	<a href="#">api/maths?op=-&amp;x=50&amp;y=25</a>
Soit deux nombres $x$ et $y$ , calculer et retourner $x \times y$	<a href="#">api/maths?op=*&amp;x=50&amp;y=25</a>
Soit deux nombres $x$ et $y$ , calculer et retourner $\frac{x}{y}$	<a href="#">api/maths?op=/&amp;x=50&amp;y=25</a>
Soit deux nombres $x$ et $y$ , calculer et retourner $x \% y$	<a href="#">api/maths?op=%&amp;x=50&amp;y=7</a>
Soit un entier $n$ , calculer et retourner $n!$	<a href="#">api/maths?op=!&amp;n=5</a>
Étant donné un entier $n$ , retourner <code>true</code> seulement si $n$ est premier	<a href="#">api/maths?op=p&amp;n=5</a>
Étant donné un entier $n$ , retourner la valeur du $n$ ième premier	<a href="#">api/maths?op=np&amp;n=5</a>

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://\[...\].glitch.me/api/maths](https://[...].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 [http<sup>1</sup>](https://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.

Note: Faites en sorte que les requêtes de verbes POST, PUT et DELETE retournent la réponse appropriée.

---

<sup>1</sup> [https://en.wikipedia.org/wiki/List\\_of\\_HTTP\\_status\\_codes](https://en.wikipedia.org/wiki/List_of_HTTP_status_codes)

**Site de tests du service** <https://votre-serveur.glitch.me/api/math>

Vous produire un site web qui permet de lancer une suite de tests pour vérifier le bon fonctionnement de votre service. Voici un exemple d'interface :

## Test du service /api/maths

### Url du service

/api/maths

### Tests

```
OK ---> {"op":"+","x":-111,"y":-244,"value":-355}
OK ---> {"op":"-","x":1,"y":"abc","error":"y parameter is not a number"}
OK ---> {"n":"a","op":"p","error":"n parameter is not a number"}
OK ---> {"op":"-","x":111,"y":244,"value":-133}
OK ---> {"op":"*","x":11.56,"y":244.12345,"value":2822.067082}
OK ---> {"op":"/","x":99,"y":11.06,"value":8.95117540687161}
OK ---> {"op":"/","x":99,"y":0,"value":"Infinity"}
OK ---> {"op":"/","x":0,"y":0,"value":"NaN"}
OK ---> {"op":"%","x":5,"y":5,"value":0}
OK ---> {"op":"%","x":100,"y":13,"value":9}
OK ---> {"op":"%","x":100,"y":0,"value":"NaN"}
OK ---> {"op":"%","x":0,"y":0,"value":"NaN"}
OK ---> {"n":0,"op":"!", "error":"n parameter must be an integer > 0"}
OK ---> {"n":0,"op":"p","error":"n parameter must be an integer > 0"}
OK ---> {"n":1,"op":"p","value":false}
OK ---> {"n":2,"op":"p","value":true}
OK ---> {"n":5,"op":"p","value":true}
OK ---> {"n":6,"op":"p","value":false}
OK ---> {"n":6.5,"op":"p","error":"n parameter must be an integer > 0"}
OK ---> {"n":113,"op":"p","value":true}
OK ---> {"n":114,"op":"p","value":false}
OK ---> {"n":1,"op":"np","value":2}
OK ---> {"n":30,"op":"np","value":113}
OK ---> {"X":"111","op":"+","y":244,"error":"x parameter is missing"}
OK ---> {"Y":"244","op":"+","x":111,"error":"y parameter is missing"}
OK ---> {"op":"+","x":111,"y":244,"z":"0","error":"too many parameters"}
OK ---> {"n":5,"op":"!", "z":"0","error":"too many parameters"}
OK ---> {"n":5.5,"op":"!", "error":"n parameter must be an integer > 0"}
OK ---> {"z":"0","error":"'op' parameter is missing"}
OK ---> {"n":-5,"op":"!", "error":"n parameter must be an integer > 0"}
OK ---> {"x":null,"error":"'op' parameter is missing"}
```

### Verdict

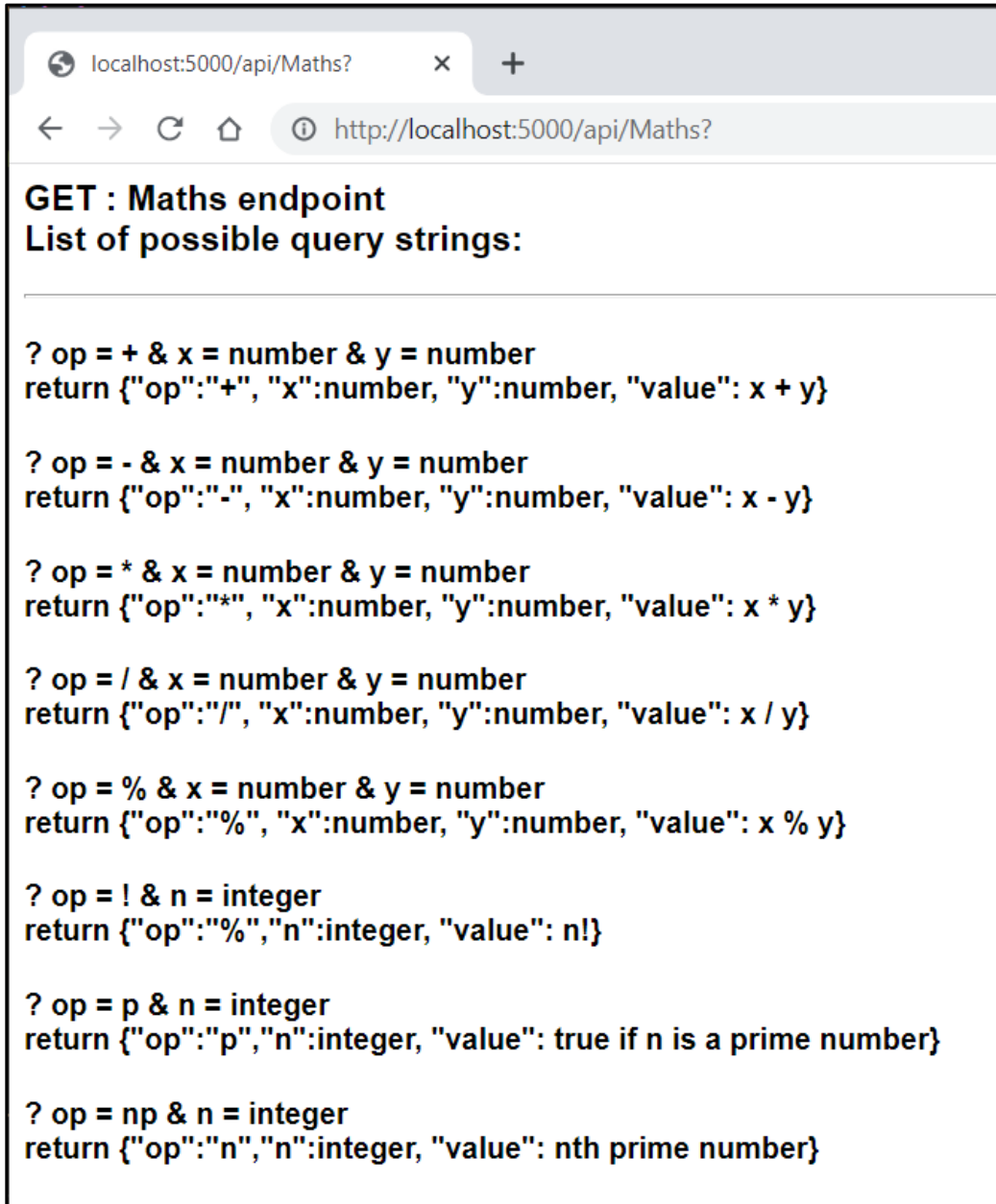
Bravo!!! Aucun problème

**Modalités de remise**

<b>Organisation</b>	Travail d'équipe
<b>Date de remise</b>	<b>Mardi le 8 octobre en classe</b>
<b>Remise par Colnet</b>	Une archive zip nommée comme suit : <b>Laboratoire-2.html</b>  Contenant les hyperliens vers votre serveur hébergé sur Glitch et votre répertoire GitHub qui inclut les sources de votre serveur.

**Note :** Les rencontres du 26 septembre et 1 et 3 octobre 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}

**Fonctions qui pourraient vous intéresser**

```
// Module mathUtilities.js

export function factorial(n) {
  if (n === 0 || n === 1) {
    return 1;
  }
  return n * factorial(n - 1);
}

export function isPrime(value) {
  for (var i = 2; i < value; i++) {
    if (value % i === 0) {
      return false;
    }
  }
  return value > 1;
}

export function findPrime(n) {
  let primeNumber = 0;
  for (let i = 0; i < n; i++) {
    primeNumber++;
    while (!isPrime(primeNumber)) {
      primeNumber++;
    }
  }
  return primeNumber;
}
```