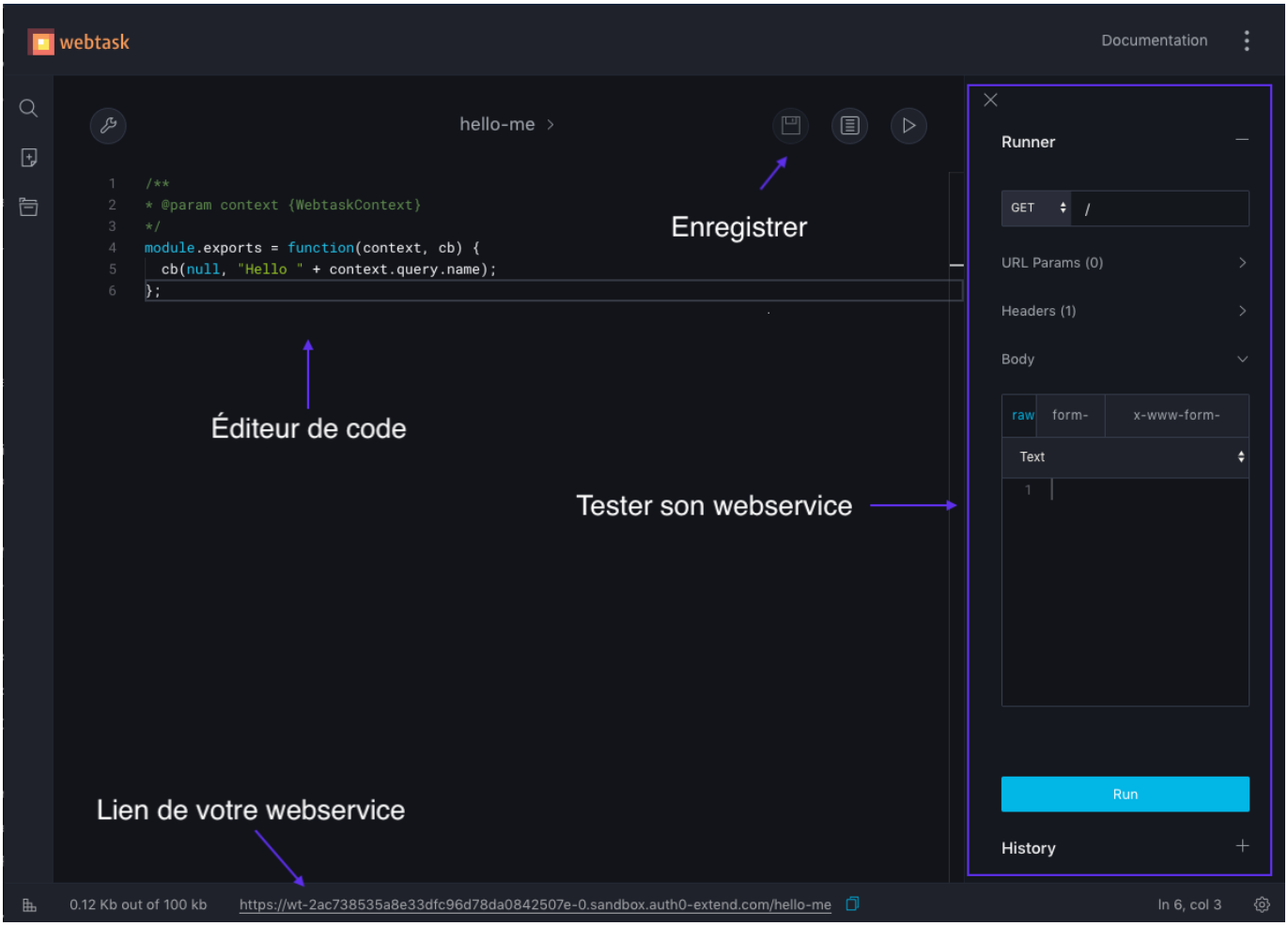
TP5 - FAAS avec Webtask

WIS 3 / Module Cloud et système web

I. Création d'un compte webtask

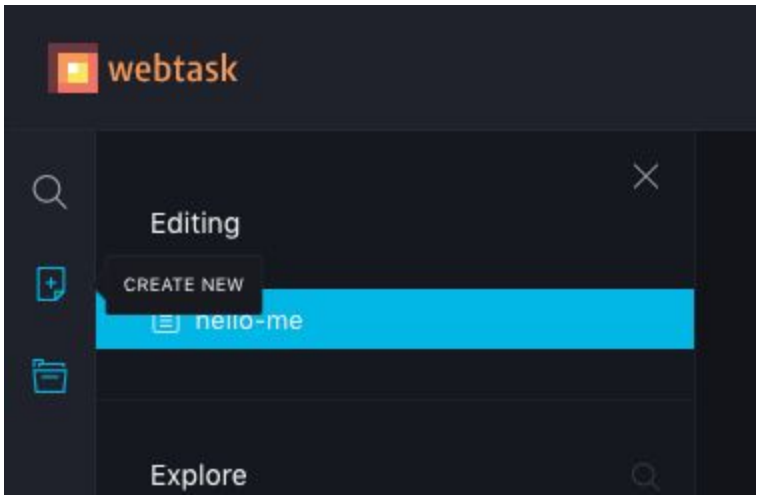
Découvrez le site internet de Webtask. Créez un compte (gratuit) pour accéder à la console Webtask

Pour simplifier l'exercice, nous développerons nos fonctions directement dans l'éditeur en ligne : https://webtask.io/make

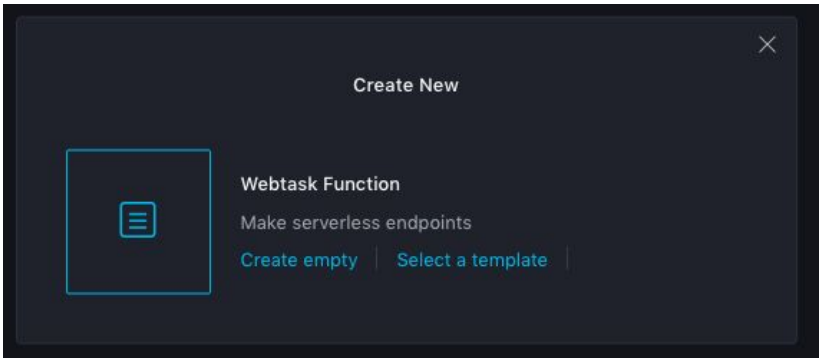


Si vous préférez écrire votre code en local, vous pouvez utiliser la CLI web task : https://webtask.io/docs/101

II. Créer son premier web service



"Create new" depuis la barre de menu



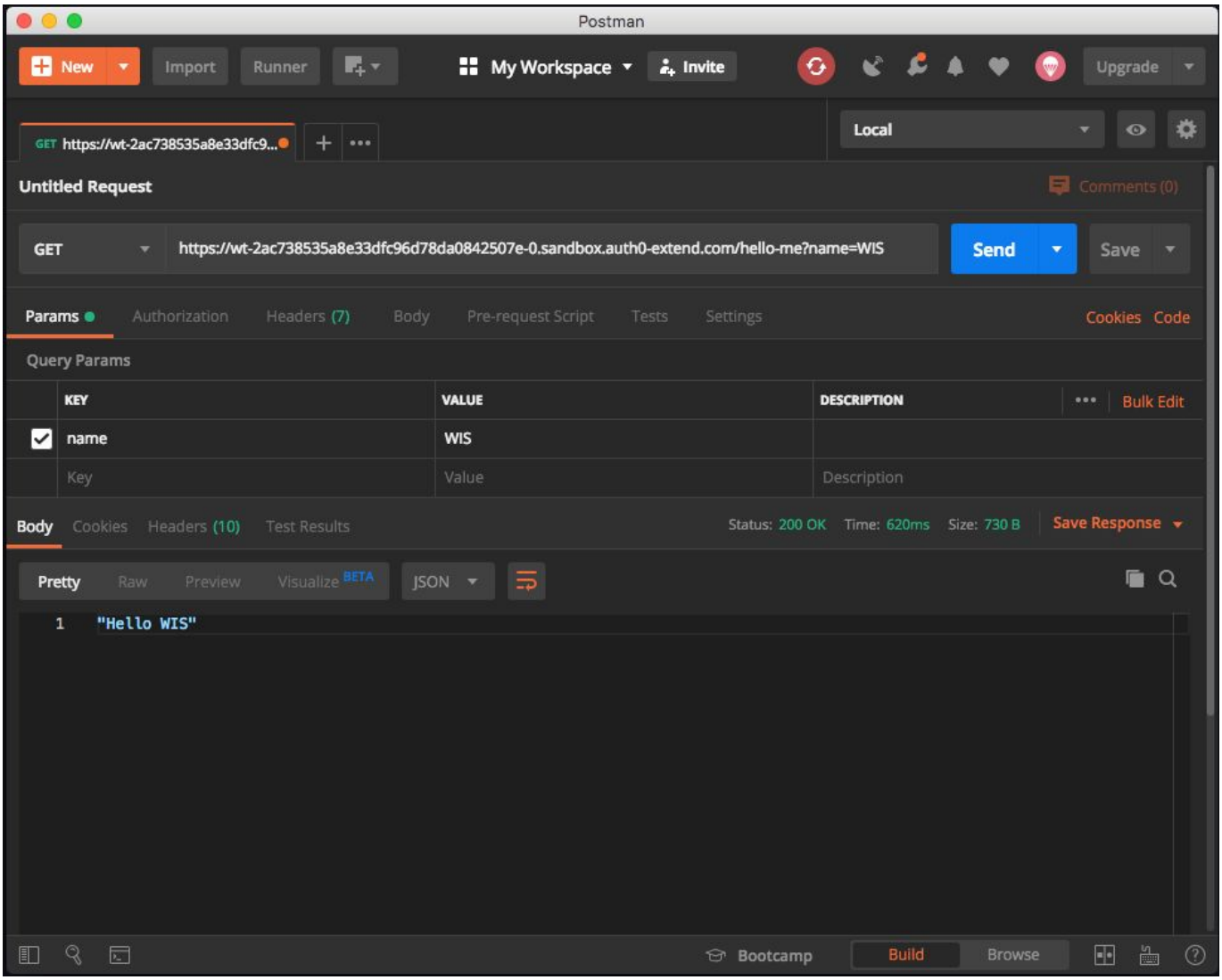
Sélectionnez "Create empty" pour créer une fonction vide

Copiez le code suivant :

```
module.exports = function(context, cb) {
 cb(null, "Hello " + context.query.name);
};
```

Ce code permet d'afficher "Hello" suivit du nom passé en paramètre GET "?name=".

Vous pouvez tester votre code en utilisant la partie de droite de d'éditeur. Mais pour rester générique et boucler avec les TP précédents, nous utiliserons Postman.



Renseignez les champs nécessaires pour afficher "Hello prénom" (URL et paramètre GET).

Vous avez créé votre premier web service en FAAS !

III. Créer un webservice pour gérer une TODO liste

Nous allons créer deux fonctionnalités sur ce nouveau web service :

- Ajouter un élément dans la liste des choses à faire
- Lister les choses à faires

Dans la bonne pratique il faudrait créer deux fonctions : une pour ajouter, une pour lister. Étant donné que le stockage est propre à chaque fonction sur webtask, nous créons une seule fonction qui fera les deux.

Exemples d'appels (résultat attendu) ::

- GET http://webtask/abcdef?task=acheter du pain => Ajoute "Acheter du pain"

- GET http://webtask/abcdef?task=poster mon colis => Ajoute “Poster mon colis” dans la liste
- GET http://webtask/abcdef => Liste des choses saisies précédemment : [“Acheter du pain”, “poster mon colis”]

Stratégie : Nous stockerons la liste des choses à faire dans un tableau Javascript. Ce tableau sera écrit dans le “storage” de webtask pour le retrouver à chaque nouvel appel.

Vous pouvez vous aider du pseudo code suivant :

1. Si le client envoie une task (variable context.query.task) :
    a. Récupérer le storage actuel dans une variable “list” avec la fonction storage.get()
    b. Si le résultat de storage.get est vide (première exécution), initialiser la variable list à vide
    c. Ajouter l'élément envoyé par l'utilisateur (context.query.task) dans le tableau “list”
    d. Sauvegarder la “list” modifiée dans le storage avec storage.set
    e. Retourner “OK” avec la fonction “cb”
2. Sinon :
    a. Récupérer le storage actuel dans une variable “list” avec la fonction storage.get()
    b. Retourner la variable “list” avec la fonction “cb”

Votre webservice est ainsi disponible directement depuis le cloud. Vous pouvez l’utiliser dans une application Angular ou PHP.

IV. Utiliser le webservice dans une application PHP

Créer une page en PHP contenant un formulaire relié à votre webservice pour ajouter des choses à faire plus facilement.

Vous pouvez utiliser la fonction PHP “file_get_content” pour faire appel à un webservice.