420-KBG : Atelier-1 (Préparé par Nicolas Chourot)

Objectifs :

* S’approprier des modules de base de Node.js
* Créer un serveur http embryonnaire
* Utiliser GitHub pour conserver une copie de sécurité
* Utiliser Glitch.com pour héberger votre service

1. Créer un répertoire *atelier-1*
2. Une image contenant texte

   Description générée automatiquementLancer VS Code et installer le greffon suivant :
3. Ouvrir le répertoire *atelier-1*
4. Ouvrir le panneau du terminal (Terminal/New Terminal)
5. Inscrire la commande suivante : npm init
6. Une image contenant texte

   Description générée automatiquementRépondez aux questions de l’assistant de création du fichier *package.json*
7. Créer le fichier *server.js* et entrer le code suivant

const http = require('http');

const server = http.createServer((req, res) => {

  console.log(req.url);

});

const PORT = process.env.PORT || 5000;

server.listen(PORT, () => console.log(`Server running on port ${PORT}`));

1. Ajouter un fichier de configuration du débogueur : Run/Add Configuration… et sélectionner *Node.js*. Cela ajoutera le fichier *lauch.json* qui servira lors de la prochaine exécution du code se trouvant dans *server.js*
2. Lancer server.js avec le raccourci clavier F5 et regarder dans le la console *DEBUG CONSOLE*. Il devrait être inscrit :  *Server running on port 5000*
3. Ouvrir une instance de Google Chrome, et entrer l’url suivant : *localhost:5000/api/contacts/15*
4. Observer l’impact dans *DEBUG CONSOLE* de *VS Code*
5. Arrêter le server avec le raccourci clavier SHIFT-F5
6. Ajouter le code suivant après l’instruction *console.log(…)* :

let reqInfo = {url:req.url, method:req.method};

    res.writeHead(200, {"Content-Type": "application/json"});

    res.end(JSON.stringify(reqInfo));

1. Tester à nouveau (penser à ajouter le greffon JSON Formatter à votre Google Chrome)

<https://chrome.google.com/webstore/detail/json-formatter/bcjindcccaagfpapjjmafapmmgkkhgoa>

1. Lancer l’application POSTMAN (télécharger et installer si pas déjà fait)
2. Ajouter une requête GET et régler l’URL à *localhost:5000/api/contacts/15*
3. Une image contenant texte

   Description générée automatiquementLancer la requête et observer le résultat.
4. Ajouter le module *query-string* avec la commande de console : *npm install query-string*
5. Ajouter dans server.js la référence à ce module :

const queryString = require('query-string');

1. Remplacer le corps du callback (req, res) => {…} par le suivant :

console.log(req.url);

let reqInfo = { url: req.url, method: req.method, contentType: req.headers['content-type'] };

res.writeHead(200, { "Content-Type": "application/json" });

if (req.method == 'GET') {

    res.end(JSON.stringify(reqInfo));

} else {

    if (req.method == 'POST') {

        let body = [];

        req.on('data', chunk => {

            body.push(chunk);

        }).on('end', () => {

            try {

                if (req.headers['content-type'] === "application/json")

                    reqInfo.body = JSON.parse(body);

                else

                if (req.headers['content-type'] === "application/x-www-form-urlencoded")

                    reqInfo.body = queryString.parse(body.toString());

                else

                    reqInfo.body = body.toString();

                res.end(JSON.stringify(reqInfo));

            } catch (error) {

                console.log(error);

            }

        });

    }

}

Note : Vous pouvez formatter le code en faisant bouton de droite et appeler « Format Document » ou appliquer l’équivalence clavier SHIT+ALT+F

1. Ajouter une nouvelle requête dans *POSTMAN*:

Méthode : POST,

URL : localhost:5000/api/contacts,

CONTENT-TYPE : application/json

BODY : {"FirstName": "Kyle", "LastName": "Ross", "Email": [Kyle.Ross@clg.qc.ca](mailto:Kyle.Ross@clg.qc.ca) }

1. Lancer la requête et observer le résultat
2. Ajouter un fichier formData.html et coller le contenu suivant :

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Document</title>

    <style>

        input {

            margin: 10px;

            height: 28px;

        }

    </style>

</head>

<body>

    <form action="http://localhost:5000/api/contacts" method="POST">

        <input type="text" placeholder="Firstname" name="FirstName" required> <br>

        <input type="text" placeholder="Lastname" name="LastName" required><br>

        <input type="email" placeholder="Email" name="Email" required><br>

        <input type="submit" value="Soumettre">

    </form>

</body>

</html>

1. Installer les greffons suivants dans VS Code :

Une image contenant texte

Description générée automatiquement

1. Lancer la page *formData.html* par l’entremise de ce dernier greffon.
2. Installer le greffons GitHub Pull Requests and Issues dans VS Code :

Une image contenant texte

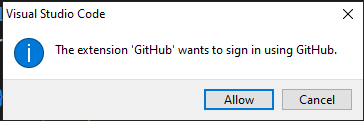
Description générée automatiquement

1. Publier le projet dans GitHub :

Une image contenant texte

Description générée automatiquement

1. Accepter de se connecter via l’extension GitHub :



1. Une instance de Google Chrome apparaîtra pour vous demander d’accepter la connexion :

Une image contenant texte

Description générée automatiquement

1. Sélectionner de publier dans un répertoire public :

Une image contenant texte

Description générée automatiquement

1. Accepter de publier tous les fichiers et répertoires :

Une image contenant texte

Description générée automatiquement

1. Message de confirmation de l’opération :

Une image contenant texte

Description générée automatiquement

1. Votre répertoire dans GitHub :

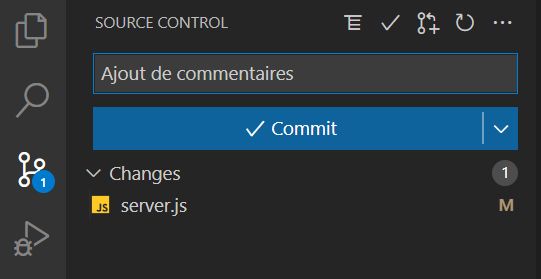
Une image contenant texte

Description générée automatiquement

1. Ajouter le commentaire au début du fichier server.js

// Mon premier server Http

1. Commettre les changements dans Git :



1. Synchroniser les changements dans GitHub :

Une image contenant texte

Description générée automatiquement

1. Créer un compte Glitch : https://glitch.com

Une image contenant texte

Description générée automatiquement

1. Sélectionnez votre compte GitHub :

Une image contenant texte

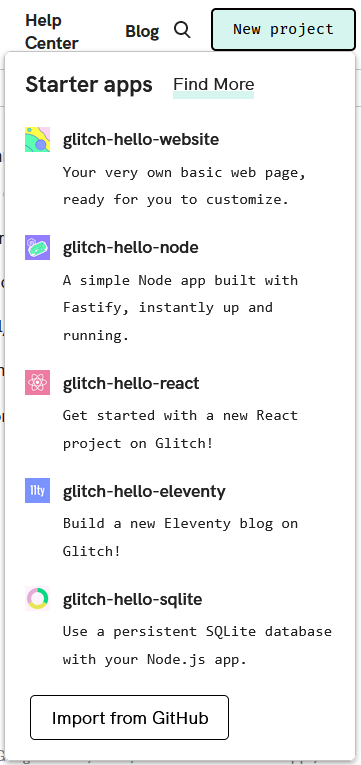
Description générée automatiquement

1. Une fois votre compte créer, ajoutez un nouveau projet :

Une image contenant texte

Description générée automatiquement

1. Importez à partir d’un répertoire GitHub :



1. Copier l’url du répertoire GitHub et coller le dans le dialogue :

Une image contenant texte

Description générée automatiquement

1. Interface de votre projet sur Glitch :

Une image contenant texte

Description générée automatiquement

1. Changer le domaine de votre site :

Une image contenant texte

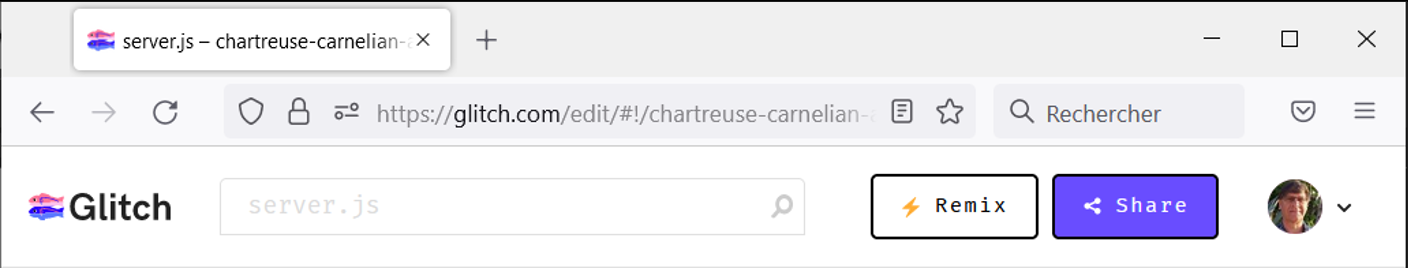
Description générée automatiquement

Une image contenant texte

Description générée automatiquement Une image contenant texte

Description générée automatiquement

1. Connaître l’url de votre site :



Une image contenant texte

Description générée automatiquement

1. Faire les modifications suivantes :

Dans formData.html :

<form action="https://202212345-atelier-1.glitch.me/api/contacts" method="POST">

Note : utilisez l’url de vote site Glitch

Dans package.json:

{

  "name": "atelier-1",

  "version": "1.0.0",

  "description": "",

  "main": "server.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1",

    "start": "node server.js"

  },

  "author": "Nicolas Chourot",

  "license": "ISC",

  "dependencies": {

    "query-string": "^7.1.1"

  }

}

Note : n’oubliez pas la virgule après l’attribut "test":... ,

1. Commettre les changements dans GitHub
2. Mettre à jour votre site Glitch avec la version GitHub :

Une image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquement

Note : cela peut prendre plus minutes

1. Vérifier le bon fonctionnement de votre site Glitch avec Postman et la version locale de formdata.html (utilisez l’url de votre site Glitch plutôt que <http://localhost:50000>...)

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Fin de l’atelier