

### 1.3°] Création de son propre Bot

La création d'un Bot sur Discord est devenue un jeu d'enfant, notamment avec les nombreuses documentations qui ont été réalisées pour cela.

La difficulté entre en jeu, lorsque vient l'envie d'ajouter des fonctionnalités complexes à son BOT.

Dans ce tutoriel, nous verrons comment créer un simple Bot répondant à des questions qui auraient été codées en dur.

Bien qu'il existe plusieurs façons de créer un Bot Discord, nous nous intéresserons à la réalisation d'un Bot à l'aide de NodeJS et de l'API DiscordJS, c'est-à-dire avec les mêmes technologies qu'utilise notre Bot que nous avons présenté précédemment.

La création du Bot se partagera en trois parties, premièrement l'installation d'un nouveau bot sur notre Serveur, puis deuxièmement l'installation de l'API DiscordJS, et les différents paramètres nécessaires afin d'avoir un environnement permettant le codage du Bot.

Et enfin en troisième partie le codage des fonctionnalités, afin de rendre notre bot "vivant".

#### Prérequis :

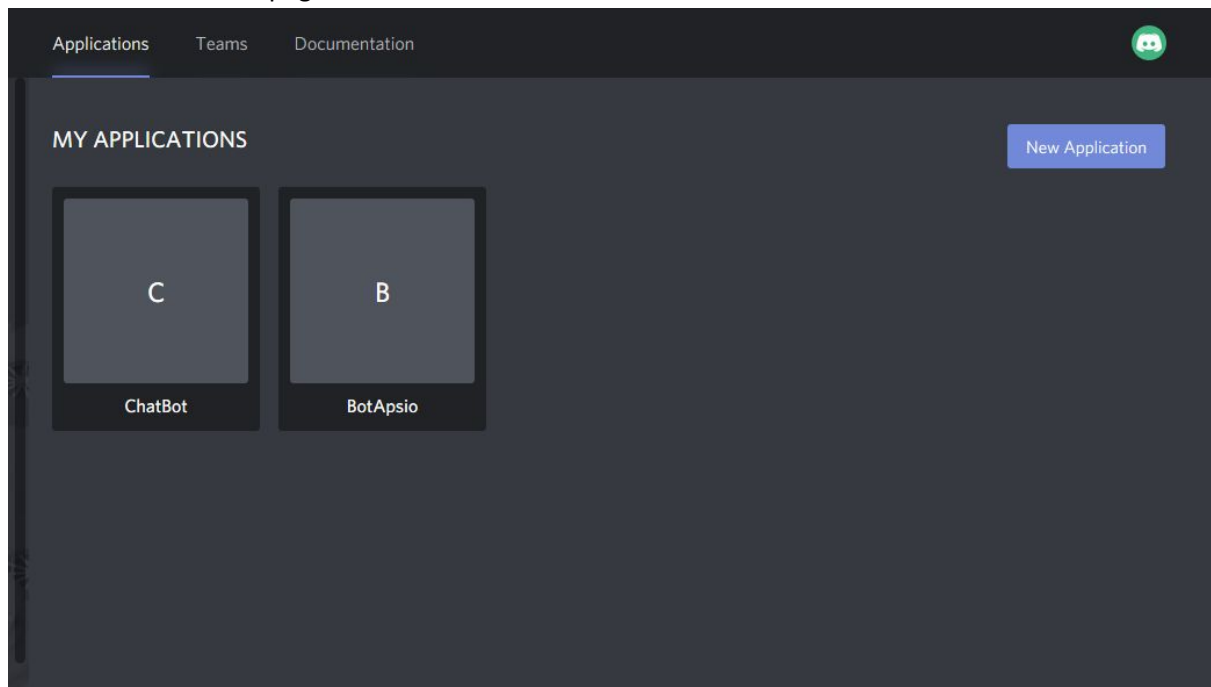
Les prérequis présents dans la partie Installation de notre BOT, sont les mêmes ici à savoir bien-sûr la possession d'un compte Discord, ainsi qu'un serveur dont vous êtes administrateur.

#### 1.3.1°] Création d'une application de type Bot sur Discord

Pour créer un bot sur votre serveur, il nous faut nous rendre dans la partie **developpers**, de votre Discord. A l'aide du lien ci-dessous :

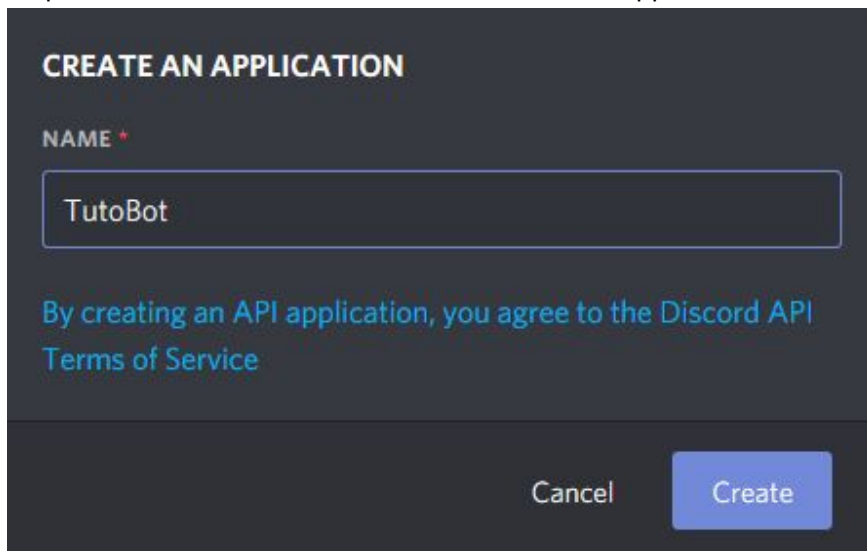
<https://discordapp.com/developers/applications/>

Vous arrivez sur cette page :



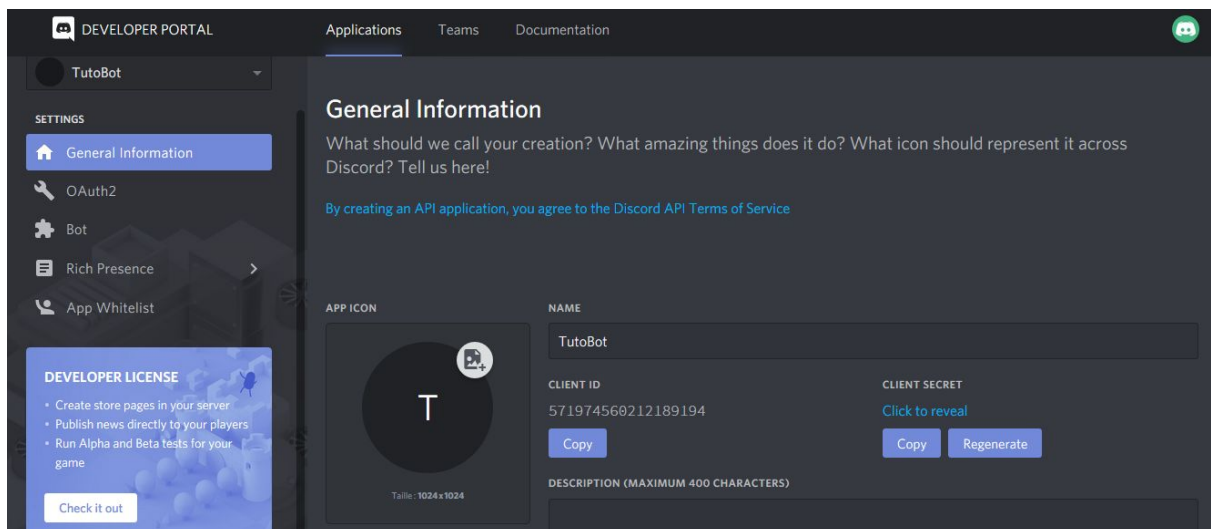
Ici, il existe déjà 2 Bots, dans votre cas vous n'aurez normalement rien d'afficher s'il s'agit de votre première création.

Cliquez ensuite sur le bouton en haut à droite New Application.

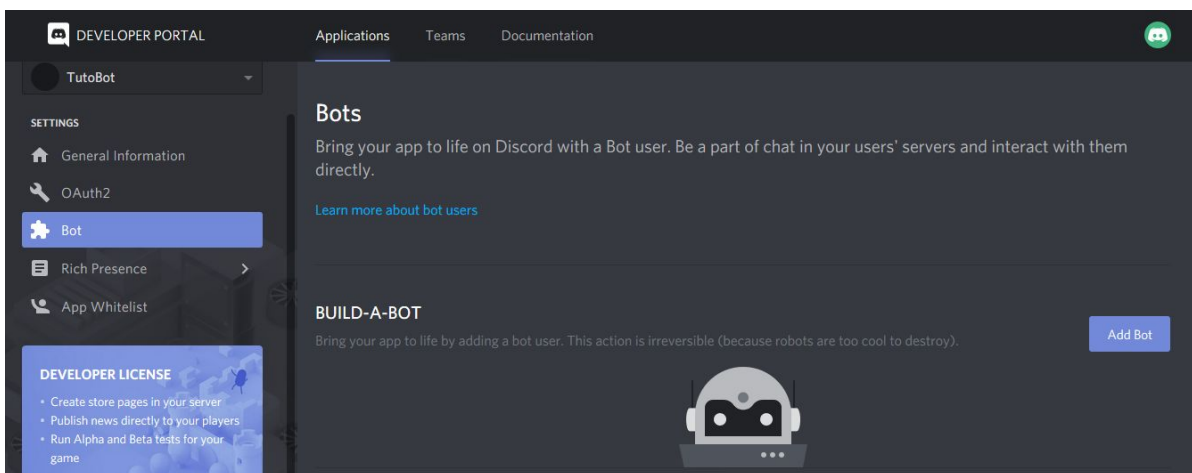


Une fenêtre s'affiche vous demandant le nom de votre application, dans notre cas le nom de votre BOT que vous souhaitez créer.

Pour la démonstration, nous l'appellerons *TutoBot*, une fois le nom entré cliquez sur Create.



Vous arrivez ensuite sur cette Page, par défaut il s'agit d'une simple application, il faut maintenant transformer cette application en BOT, pour cela rendons-nous dans le 3 ème onglet **Bot**, du menu Settings à gauche de la page.




Nous arrivons sur cette page, il vous suffit simplement de cliquer sur Add Bot.

A wild bot has appeared!

### BUILD-A-BOT

Bring your app to life by adding a bot user. This action is irreversible (because robots are too cool to destroy).

ICON



USERNAME

TutoBot

#3537

TOKEN

[Click to Reveal Token](#)

CopyRegenerate

PUBLIC BOT

Public bots can be added by anyone. When unchecked, only you can join this bot to servers.

Votre bot est maintenant créé, il ne manque plus qu'à l'ajouter sur votre serveur.

Pour cela, il faut d'abord passer par une étape, qui est celle des ajouts de permissions à votre Bot. Le paramétrage de ces permissions va vous générer un lien permettant d'installer le bot sur votre serveur.

Pour créer ces permissions, il vous faut vous rendre dans la partie Calcul des permissions à l'aide du lien suivant : <https://discordapi.com/permissions.html>

Vous arrivez sur cette page :

Permissions: 0

Equation: 0 =

#### General Permissions

☐ Administrator

☐ View Audit Log

☐ Manage Server

☐ Manage Roles

☐ Manage Channels

☐ Kick Members

☐ Ban Members

☐ Create Instant Invite

☐ Change Nickname

☐ Manage Nicknames

☐ Manage Emojis

☐ Manage Webhooks

#### Text Permissions

☐ Read Messages

☐ Send TTS Messages

☐ Embed Links

☐ Read Message History

☐ Use External Emojis

☐ Send Messages

☐ Manage Messages

☐ Attach Files

☐ Mention @everyone

☐ Add Reactions

#### Voice Permissions

☐ View Channel

☐ Connect

☐ Mute Members

☐ Move Members

☐ Speak

☐ Deafen Members

☐ Use Voice Activity

☐ Priority Speaker

Colored = bot owner must have 2 Factor Authentication enabled

#### OAuth URL Generator

Client ID:

☐ Require Code Grant

Redirect URI:

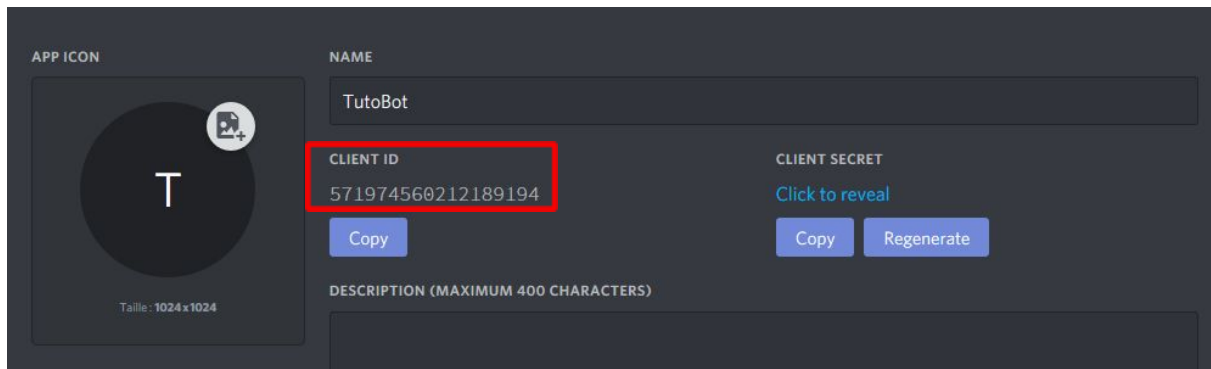
Scope:

Link: [https://discordapp.com/oauth2/authorize?client\\_id=INSERT\\_CLIENT\\_ID\\_HERE&scope=bot&permissions=0](https://discordapp.com/oauth2/authorize?client_id=INSERT_CLIENT_ID_HERE&scope=bot&permissions=0)

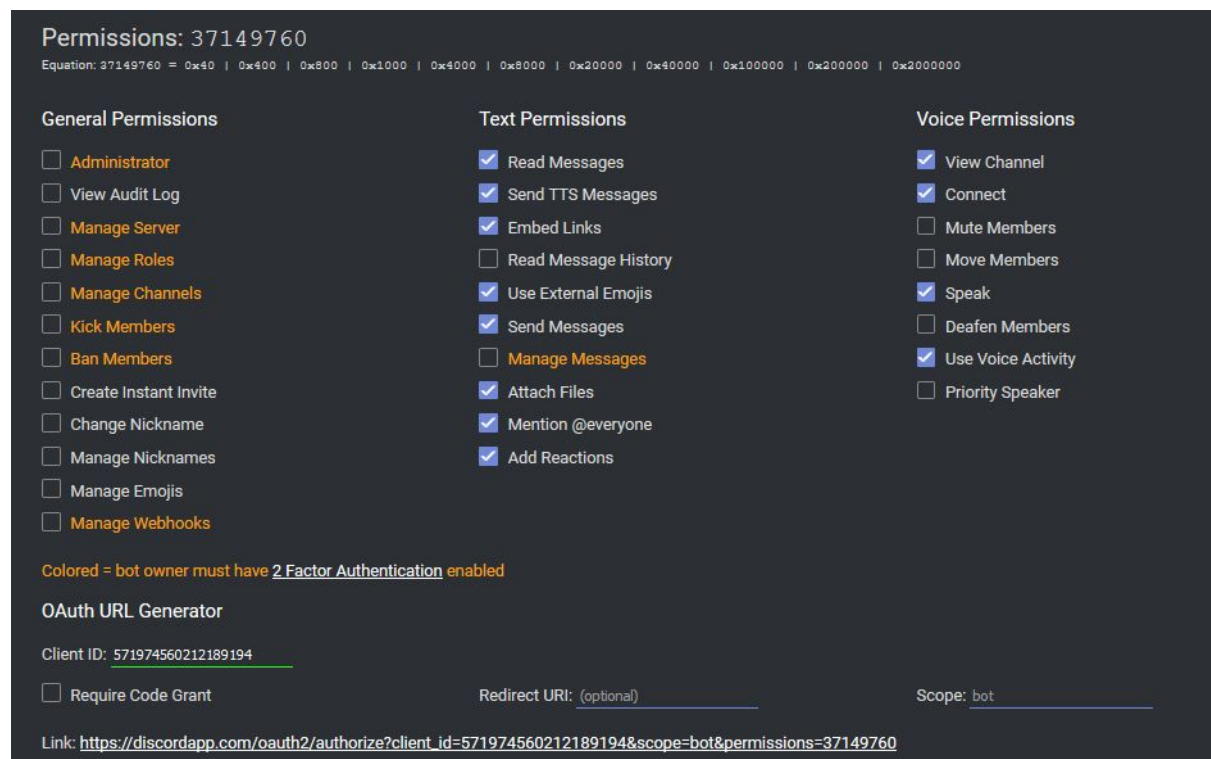
Cette page, contient 3 listes de permissions, à vous de cocher les actions que votre bot aura l'autorisation de faire sur les serveurs sur lesquels il sera installé.  
Les différentes cases cochées vont vous générer un nombre de permission, afficher en haut à gauche de la page.

Vous devrez aussi insérer le **Client ID**, de votre Bot dans le champ prévu à cet effet en bas de la page à gauche.

Le client ID de votre Bot se trouve dans la page précédente. Ici ci-dessous encadré en rouge.



Les différentes permissions assignées et le client ID inséré, il ne vous reste plus qu'à cliquer sur le lien qui a été généré.



Permissions: 37149760  
Equation: 37149760 = 0x40 | 0x400 | 0x800 | 0x1000 | 0x4000 | 0x8000 | 0x20000 | 0x40000 | 0x100000 | 0x200000 | 0x2000000

General Permissions	Text Permissions	Voice Permissions
<input type="checkbox"/> Administrator	<input checked="" type="checkbox"/> Read Messages	<input checked="" type="checkbox"/> View Channel
<input type="checkbox"/> View Audit Log	<input checked="" type="checkbox"/> Send TTS Messages	<input checked="" type="checkbox"/> Connect
<input type="checkbox"/> Manage Server	<input checked="" type="checkbox"/> Embed Links	<input type="checkbox"/> Mute Members
<input type="checkbox"/> Manage Roles	<input type="checkbox"/> Read Message History	<input type="checkbox"/> Move Members
<input type="checkbox"/> Manage Channels	<input checked="" type="checkbox"/> Use External Emojis	<input checked="" type="checkbox"/> Speak
<input type="checkbox"/> Kick Members	<input checked="" type="checkbox"/> Send Messages	<input type="checkbox"/> Deafen Members
<input type="checkbox"/> Ban Members	<input type="checkbox"/> Manage Messages	<input checked="" type="checkbox"/> Use Voice Activity
<input type="checkbox"/> Create Instant Invite	<input checked="" type="checkbox"/> Attach Files	<input type="checkbox"/> Priority Speaker
<input type="checkbox"/> Change Nickname	<input checked="" type="checkbox"/> Mention @everyone	
<input type="checkbox"/> Manage Nicknames	<input checked="" type="checkbox"/> Add Reactions	
<input type="checkbox"/> Manage Emojis		
<input type="checkbox"/> Manage Webhooks		

Colored = bot owner must have 2 Factor Authentication enabled

OAuth URL Generator

Client ID: 571974560212189194

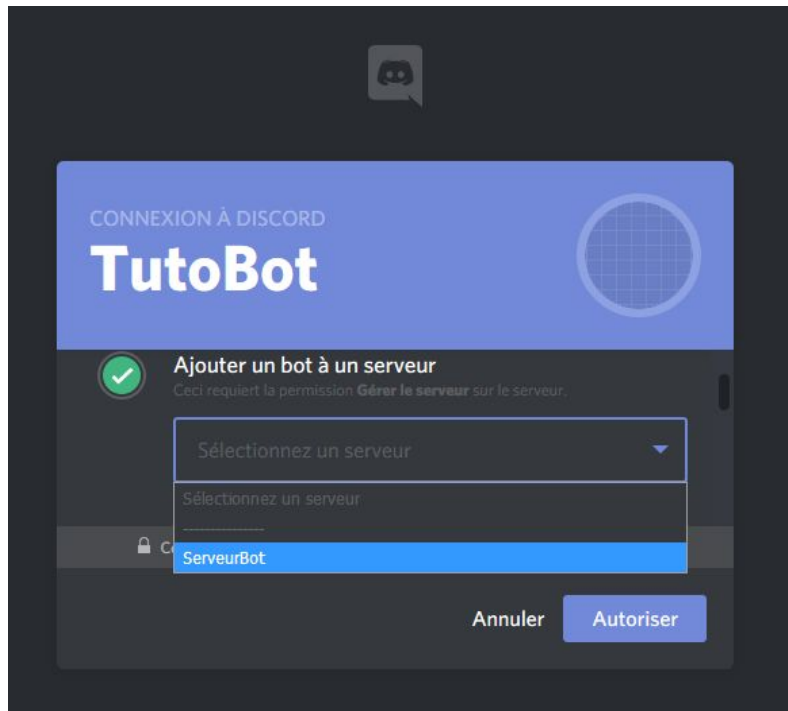
☐ Require Code Grant

Redirect URI: (optional)

Scope: bot

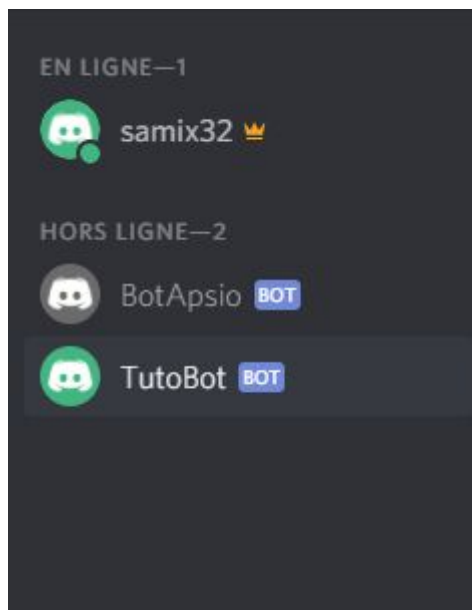
Link: [https://discordapp.com/oauth2/authorize?client\\_id=571974560212189194&scope=bot&permissions=37149760](https://discordapp.com/oauth2/authorize?client_id=571974560212189194&scope=bot&permissions=37149760)

Ce lien vous emmène sur cette page vous demandant sur quel serveur voulez-vous installer le bot.



Choisissez le Serveur, puis cliquez sur Autoriser.

C'est bon notre Bot est sur notre Serveur



### 1.3.2°] Installation de l'API + Paramétrage

Le but de cette partie est de créer un environnement qui va vous permettre de développer votre Bot fraîchement installé sur votre serveur.

Comme dit dans l'introduction, notre bot sera codé à l'aide l'API Discord.js, nous allons donc d'abord installer node.js.

Allez sur le lien : <https://nodejs.org/fr/>

Puis cliquez sur le premier bouton "Recommandé pour la plupart des utilisateurs".

## Téléchargements pour Windows (x64)

**10.16.0 LTS**

Recommandé pour la plupart des utilisateurs

[Autres téléchargements](#) | [Journal des modifications](#) | [API Docs](#)

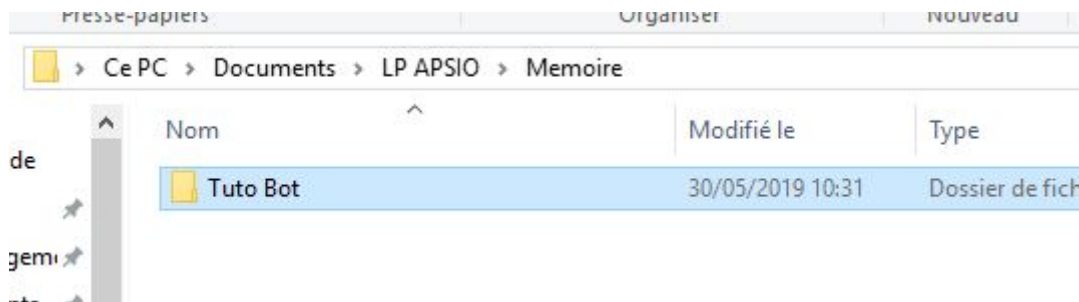
**12.3.1 Actuelle**

Dernières fonctionnalités

[Autres téléchargements](#) | [Journal des modifications](#) | [API Docs](#)

Procédez maintenant à l'installation de node.js en suivant les étapes qui vous seront indiquées.

NodeJs installé, créez-vous un dossier qui contiendra le futur programme du Bot.



Ouvrez ce dossier, puis à l'intérieur de ce dossier ouvrez une invite de commande. Pour ce faire dans ce dossier tapez dans la fenêtre de chemin "cmd", comme ci-dessous :



Puis tapez entrée, un invité de commande s'ouvrira dans le dossier, dans cet invité de commande tapez la commande "npm init", puis faites entrée.



```
Microsoft Windows [version 10.0.17134.765]
(c) 2018 Microsoft Corporation. Tous droits réservés.

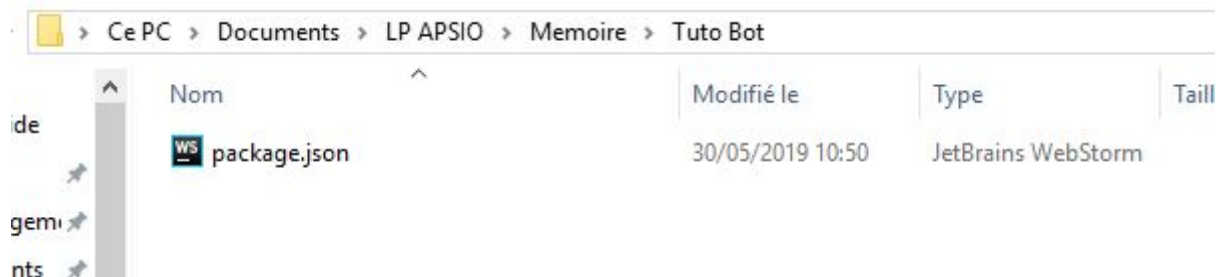
C:\Users\skyze\Documents\LP APSIO\Memoire\Tuto Bot>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (tuto-bot)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to C:\Users\skyze\Documents\LP APSIO\Memoire\Tuto Bot\package.json:
{
  "name": "tuto-bot",
```

A chaque question faites entrée. Cela vous générera un fichier package.json dans votre dossier :

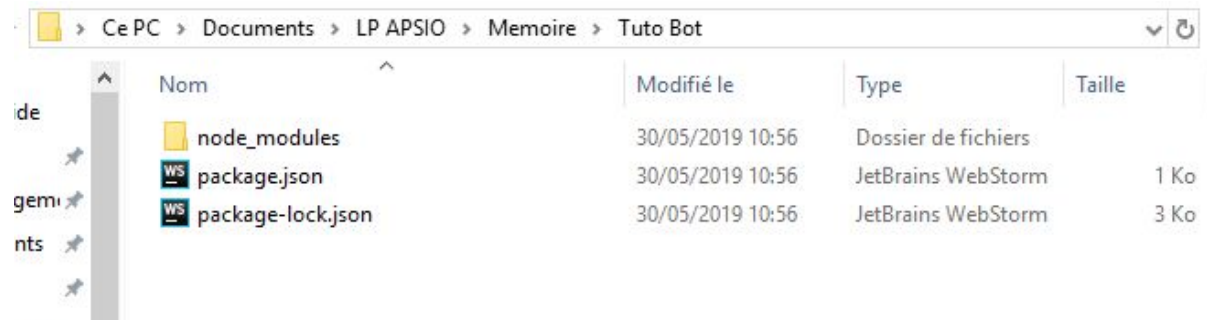


Nous pouvons dès à présent installer Discord.js. Pour ce faire dans l'invite de commande tapez la commande "npm install discord.js"

```
C:\Users\skyze\Documents\LP APSIO\Memoire\Tuto Bot>npm install discord.js
npm WARN deprecated snekfetch@3.6.4: use node-fetch instead
npm WARN deprecated created a lockfile as package-lock.json. You should commit this file.
npm WARN discord.js@11.5.1 requires a peer of bufferutil@^4.0.0 but none is installed. You must install
s yourself.
npm WARN discord.js@11.5.1 requires a peer of erlpack@discordapp/erlpack but none is installed. You
dependencies yourself.
npm WARN discord.js@11.5.1 requires a peer of libsodium-wrappers@^0.7.3 but none is installed. You
dependencies yourself.
npm WARN discord.js@11.5.1 requires a peer of node-opus@^0.2.7 but none is installed. You must inst
yourself.
npm WARN discord.js@11.5.1 requires a peer of opusscript@^0.0.6 but none is installed. You must ins
s yourself.
npm WARN discord.js@11.5.1 requires a peer of sodium@^2.0.3 but none is installed. You must install
yourself.
npm WARN discord.js@11.5.1 requires a peer of @discordjs/opus@^10.149.0 but none is installed. You m
dependencies yourself.
npm WARN tuto-bot@1.0.0 No description
npm WARN tuto-bot@1.0.0 No repository field.

+ discord.js@11.5.1
added 7 packages from 6 contributors and audited 7 packages in 10.322s
found 0 vulnerabilities
```

Nous pouvons constater que dans notre dossier, nous avons maintenant un nouveau fichier "package-lock.json" et un dossier node\_modules :



Notre environnement est maintenant prêt, nous pouvons commencer la partie programmation.

### 1.3.3°] Ajout de fonctionnalités au Bot

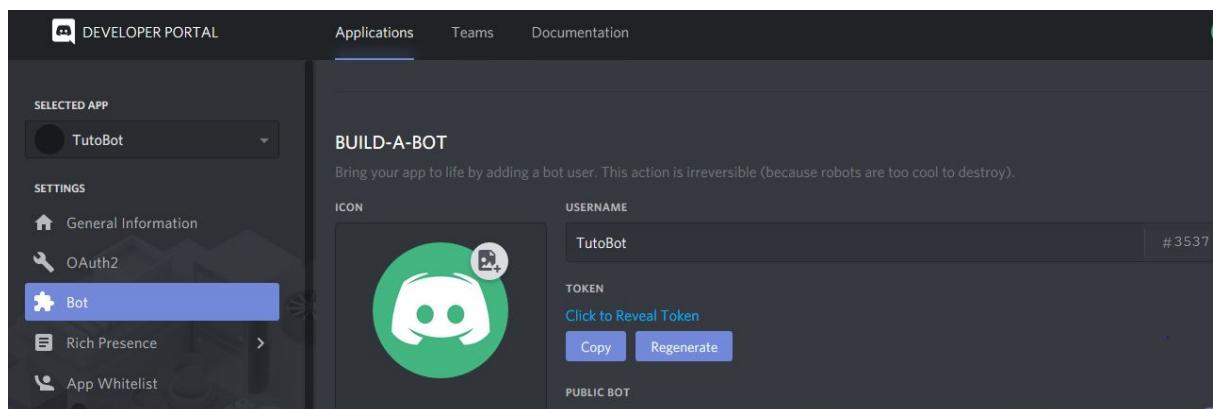
A partir de l'éditeur de votre choix, créer un nouveau fichier dans le dossier, que nous avons créé précédemment. Nommez le "index.js", C'est dans ce fichier que nous coderons les fonctionnalités du bot.

Déclarons 2 constantes, une afin d'appeler l'api Discord.js et la seconde pour définir notre bot à partir de la première constante. Puis connectons notre bot à l'aide de la méthode login qui prend en paramètre le token du bot précédemment créé.

```
1 const Discord = require('discord.js');
2 const myBot = new Discord.Client();
3 myBot.login("***TOKEN***");
4
```

Vous trouverez votre TOKEN dans la partie développeur de l'application Discord dans le menu Bot.

Lien : <https://discordapp.com/developers/applications/>



Faite Copy, puis coller ce token dans le paramètre de la méthode login à la place de "\*\*\*TOKEN\*\*\*" sur la capture précédente.



Points de Vigilances : Attention ce token est confidentiel, il est impératif que vous soyez le seul à le connaître, vous ne devez en aucun le partager à une autre personne. En effet c'est à partir de ce token que nous pouvons prendre le contrôle du bot.

A partir de là, si vous exécutez votre programme, votre bot se connectera.

Nous allons maintenant permettre au bot d'envoyer un message selon ce qu'on lui envoie. Pour cela on utilise la méthode de gestion des événements de l'objet myBot, que nous avons instancié précédemment.

En détectant les événements de type message pour cela taper le code suivant :

```
myBot.on('message', function(data) {  
  
});
```

C'est dans cette méthode que nous allons interagir avec le bot. Nous pouvons par exemple dire que si le bot détecte le message "Bonjour", il nous répond "Salut".

Code :

```
myBot.on('message', function(data) {  
    if(data.content == "Bonjour"){  
        data.channel.send("Salut")  
    }  
  
});
```

Ce morceau de code écrit, vous pouvez tester la première fonctionnalité de votre bot. Ouvrez un invite de commande dans le répertoire où se trouve le fichier "index.js", puis tapez la commande "node index.js". Cette commande va exécuter le programme que nous venons d'écrire.

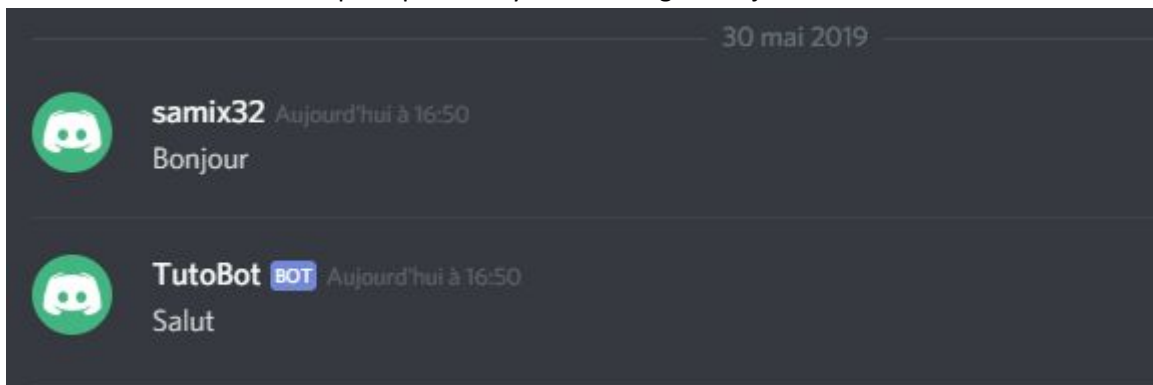
```
C:\Windows\System32\cmd.exe - node index.js
Microsoft Windows [version 10.0.17134.765]
(c) 2018 Microsoft Corporation. Tous droits réservés.

C:\Users\skyze\Documents\LP APSIO\Memoire\Tuto Bot>node index.js
```

Rendez-vous sur Discord. Vous apercevrez votre bot connecté :



Maintenant il ne nous reste plus qu'à envoyer le message "Bonjour".



Nous pouvons voir que le Bot nous a bien répondu.

Vous pouvez maintenant créer une suite de conditions comme celle présentée dans notre exemple pour créer votre propre ChatBot Discord. N'hésitez pas à vous aider de la documentation officielle : <https://discord.js.org/#/>