

Hosting on Glitch.com

VIEWER REQUEST #02: GLITCH.COM HOSTING





It is strongly advised you actually shell out a couple dollars a month and get an actual real VPS, due to the free nature of Glitch it has led to a high number of Glitch based IP's getting API banned by Discord due to abuse.

HOLY FUCKING HELL GUYS I'VE FOUND IT

I've finally found it. I found a free hosting that you guys can enjoy and put your bots on, that's not like the most complicated 4D puzzle in the world to get functional, and that *actually* supports not only the most recent version of node, but also saving files locally so you can have a simple bot with persistence without so much hassle you'd be better off waiting until you had a job to get hosting.

So OH MY GOD, let's get to it right now because I'm so excited. I mean there's *tiny* little things you have to be weary of, but it's, like, **Almost Paradise**. Let's knock on heaven's door.

Getting Code Running

Ok like, have you ever seriously looked into getting a bot up on heroku? God it's a mess of setups and configurations and *holy crap what the hell is a worker and a procfile?????*

Glitch.com has **none** of this. Getting code up and running is as simple as:

- Open your browser to [Glitch.com](https://glitch.com)

- **Completely ignore the childish drawings of a 3 year old on the front page** (trust me on this one, don't look)
- Click on **Start a New Project**, then **Create a Node App**
- You're... well technically you're done.

That's right. You can start coding right away, technically. The default project is an express.js website.

Don't delete that yet, we'll still need some of that code later on. Because, as simple as this might be and, well, really is actually, there's still a few things that we need to set up, including making sure the project stays online, and securing it against prying eyes.

Create your account

In order to not ever lose access to your code, the first thing you need to do is to create an account.

- Click **Sign In** at the top-right of the page
- Choose either Github or Facebook to login.
- Yeah ok the project is now yours.

Toldja this shit was simple.

Configure the Project

So here's a few things about the project that we need to configure, for a few reasons.

First off, **Secure the project**. By default, anyone with your project's name can access your code directly. They can't *edit* it but they can snoop in and look at your code. And, btw, I haven't found a way to un-share someone that's viewed your project yet (I'm talking to Glitch to get that fixed).

- Click on the **Share** Button at the top of the file list, besides your name.
- Click on **Make Private**.
- You can still invite people to view and collaborate later, with the link provided.

⚠ If you make your project private, people can snoop if you accidentally give out the project name and see the tokens in your env file, if you leave it public they can view your code, but cannot view any tokens inside the env file.

The next thing is, **Name the project**. Now, projects work through express.js whether you really want to or not. Later on you can learn to make a dashboard but for now, we just need to set it up to keep it alive.

- Click on the project name at the top-left of the screen (mine was `best-glue`, these guys know what I sniff I tell ya!)
- In the pop-up click on the name at the top (*slightly* counter-intuitive, but yeah that's how you rename it)
- The name you choose (it must be unique and not taken by anyone else) will be your "site's" subdomain address.
- While you're at it, give it a description if you really want to.

Finally, we need to **Disable some auto-save features**. Glitch automatically saves the file, quite literally, on every keypress you make. And restarts it. This is not only slightly visually annoying, but also damaging to

bots - the Discord API will reset your bot's token if you login 1000 times in a day. That means, if you type 1000 characters in your code, as it is. QUITE an issue.

- Create a new file in the project, and call it `watch.json`
- Paste in the following code in it:

```
1  {
2    "install": {
3      "include": [
4        "^package\\.json$",
5        "^\\.env$"
6      ]
7    },
8    "restart": {
9      "exclude": [
10       "^public/",
11       "^dist/"
12     ],
13     "include": [
14       "\\\.js$",
15       "\\\.json"
16     ]
17   },
18   "throttle": 900000
19 }
```

This number, `900000`, means that every 15 minutes, if any files have changed, the bot will restart. Now there is a caveat here, which is that this also means any change you do in the bot will not take effect (will not reboot) until, up to, 15 minutes. But hey. It's free, let's not look a gift horse in the mouth!


Oh one last thing for you crazy people with light-sensitive eyes (aka dark theme users) : click on your avatar at the top right, then click on "Change Theme". **I know, right? You're welcome!**

Keeping the project "alive"

Alright so, like, Glitch is made to be a *web* hosting really, and will "sleep" after 5 minutes if it receives no HTTP request. However, there is a very convenient way to keep it alive, which is actually provided by the app itself - the `express.js` module is pre-installed, and all you need to do is to "ping" it every 5 minutes to make sure it doesn't sleep. These lines of code in your project (either the main file or any module you call on bootup) should do the trick for now:

```
1  const http = require('http');
2  const express = require('express');
3  const app = express();
4  app.get("/", (request, response) => {
5    console.log(Date.now() + " Ping Received");
6    response.sendStatus(200);
7  });
8  app.listen(process.env.PORT);
9  setInterval(() => {
10   http.get(`http://${process.env.PROJECT_DOMAIN}.glitch.me/`);
11 }, 280000);
```

What does this do? Keeps an express.js server alive, which does not really affect your project in and of itself, and pings itself every 5 minutes, so it never shuts off. Awesome.

 For best results, have an outside source pinging the project address as well, glitch.com suggests using [Uptime Robot](#).

Package.json

There are 2 things that you much change in the project's package.json file in order to ensure that your project will actually work.

First, you must provide for a *node.js version* if your project requires a higher version of node (for instance, 8.4.0). This is done with the `engines` key, as such: `"engines": { "node": "8.4.0" }`

Second, you must provide for the `start` script. A lot of us just generally configure the `main: index.js` key and this is not sufficient. You must provide for a start script:

```
1  "scripts": {  
2    "test": "echo \"Error: no test specified\" && exit 1",  
3    "start": "node index.js"  
4  },
```


I show the `test` script here because this is by default in any project where `npm init` was used, so it's a good point of reference. Here's a full package.json, this one is guidebot's modified version:

```
1  {
2    "name": "guidebot",
3    "version": "2.0.3",
4    "description": "A boilerplate example bot with command handler and reloadable com
5    "main": "node index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1",
8      "start": "node index.js"
9    },
10   "engines": { "node": "8.4.0" },
11   "repository": {
12     "type": "git",
13     "url": "git+https://github.com/AnIdiotsGuide/guidebot.git"
14   },
15   "author": "The Idiot's Guide Community",
16   "license": "MIT",
17   "bugs": {
18     "url": "https://github.com/AnIdiotsGuide/guidebot/issues"
19   },
20   "homepage": "https://github.com/AnIdiotsGuide/guidebot#readme",
21   "dependencies": {
22     "discord.js": "^11.2.1",
23     "enmap": "^0.3.2",
24     "moment": "^2.18.1",
25     "moment-duration-format": "^1.3.0",
26     "express": "^4.15.5"
27   }
28 }
```

Another change is that your `config.json` file or `config.js` file is **insecure** if you share your project.

The easiest way to fix this is to use environment variables. Open the `.env` file, and add the following line:


```
TOKEN=MzUzOTUxODYwOTA3OTY2NDY0.DI3K3w.VN1Gvs17CSh2IYIELJDJAFejH4w
```

 Obviously use your real token, duh!

You can then access this from anywhere using `process.env.TOKEN`, so again with the guidebot example, you would do the following in `config.js`:

```
1 // Your Bot's Token. Available on https://discordapp.com/developers/applications/  
2 "token": process.env.TOKEN,
```

F.A.Q.

- X was compiled against a different Node.js version

That error means you forgot to set your engines in your package file, make sure you have added `"engines": { "node": "8.4.0" }` to your package, you can also use the following commands in the console (you can access the console by going to your Project Name > Advanced Options > Open Console), `nvm use NODE VERSION`, followed by `npm rebuild`, that should eliminate the error.

- "My bot goes down after X"

Glitch puts projects to sleep after 5 minutes of inactivity, with the code you used near the start of the guide you should have at least one step towards keeping your bot online 24/7, how ever there is another tool you can use, and this is promoted by the Glitch team, and that tool is [Uptime Robot](#).

First you will need to go to the web address of your project (`https://project-name.glitch.me`). Then login to Uptime Robot and click on `+ Add new monitor` , set **monitor type** as `HTTP(S)` , give the monitor a name and in the **URL** field paste in your glitch project URL. Don't forget to set the **monitoring interval** to `Every 5 minutes (default)` , click save and job's done!

Getting Help

Here are a few Glitch resources:

- [Glitch Discourse](#) (their Q&A/Forum place)
- [Glitch FAQ](#) (some pertinent technical details)