Gacha For Stream Avatars

made by [@MatsuTheBear](#)

Format of the Tutorial

Requirements

- **Python 3** (version must be above or equal to 3.10, or else the code will not work!)
- **Macro Program** that allows to launch shell programs or Python files with arguments passed through Twitch APIs (**Touch Portal was used in this tutorial**)
- **Stream Avatars** (of course) and a **Twitch Account** (feature **not** tested on Youtube)

First Setup

There is nothing to change for the first setup. The only thing you have to do is go inside the **run-files** folder and modify **windows.bat** if you are using Windows, or **macos.sh** if you are using macOS. Inside those files, replace YOURPATHHERE with the path of the python file.
- On Windows, you can do it by right-clicking the python file, and select **Copy as Path**.
- On macOS, left-click the file and then press Alt. It will show on the bottom of the Finder screen, and you can then press right click on the python name→ **Copy gacha.py as PathName**

Path consists of the directory of the file and the **file itself,** so examples of paths are
- macOS: /Users/Matsu/GachaGithub/**gacha.py**
- Windows: F:/GachaGitHub/**gacha.py**

Info about the Program and more

What does it do?

GachaForStreamAvatars is a Python program that allows people to simulate a gacha, enhancing StreamAvatars. Users are able to pull for avatars and gears, ranking from R to SSS, using channel points, donations or anything that can trigger an event, and then use them during your streams. The streamer has total control of what rarity avatars and gears have and their % of being pulled. If the same avatar or gear is pulled, rerolls are done. After N rerolls (number chosen by the streamer) the program will instead give back coins/points, used by StreamAvatars (you can enable avatars to be bought with points). It's a fun gimmick for members that have a lot of channel points.

Is it working as intended?

Yes! But bugs and glitches can happen. So as a safe measure, the program will register every avatar pulled for each user, so in case there is a problem with I/O speeds, OBS or whatever the reason is, you can always check who pulled what. And for whatever reason you program is not working, you can still

Where can I find you and where can I support you?

If you need to ask me anything, you can do it with a DM on Twitter @MatsuTheBear or on Discord (the link will direct you to my Discord server). I'll be more than happy to help you.

If you want to support me, you can do it by following me on Twitter and Twitch. If you want to support me in other ways, I have a Ko-Fi page where you can also commission me emotes

Remember that this software is **free to use**, you don't have to pay me! And if you paid for it, you got scammed. I'm sorry.

Updates and more

My main goal is to convert everything as a plugin for StreamAvatars. Updates will be done through Github and Twitter, as well on my Discord server. For any bug you can notify me on my socials. Don't be afraid to contact me for any question or to ask for tech help, I will be more than happy to help you! Keep in mind that living in Italy, our timezones may be different, so I might not respond immediately.

How to install Python on Windows and macOS(Option 1)

1) **Download Python** from the official site (https://www.python.org/downloads/). Select the version you need. Remember to check the version shown. Every version above 3.10.0 is good, as long as is not below 3.10 (with 3.9 the code won't work, for example)

2) During the installation process, **be sure to select ADD TO PATH** if the selection is available (or add to environment variables if defined differently). Feel free to select other options based on your preferences.

- If you forgot to select PATH, *you don't need to uninstall Python*. You can still re-launch the installer after the main setup, and select "Modify". You should find the option available to be selected
- For Mac users, there is a high chance that you have to manually add the Python path to the environment variables file. See **Option 2** to see how to do it.

3) In order to test if Python was installed correctly, **open Command** on Windows or **Terminal** on MacOS, **type "python"** (without quotation marks) and **press enter**. If everything works, you should see something like this:

```
Python 3.10.7 (v3.10.7:6cc6b13308, Sep  5 2022, 14:02:52) [Clang 13.0.0 (clang-1
300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

If you have multiple Python interpreters installed or it does not work, try using **python3** instead

How to install Python on macOS(Option 2)

Is it true that Python 2.7 was already installed on older versions of MacOS, but thanks to Apple (why Apple, why), on Macs with M1/M2 launching Python directly from Terminal will (likely) not work. At least, for me it did not. There are two solutions, but they are kinda scary for non-programmers or people that don't usually stream on a Mac, so if you don't know what to do, be free to DM me on Twitter @MatsuTheBear
1) After installing Python normally, add Python to /etc/paths
2) Install brew/homebrew, and then install python with brew (if everything doesn't work!)

1) Python to /etc/paths (Do this!)

- Go to **Launchpad**, select **Python Launcher** and copy the path shown. If you are not sure where to go, you can also copy this path: **/usr/local/bin/python3**
- Open the **Terminal** and enter the following command: **sudo nano /etc/paths**
- Enter your password.
- Press Down-Key until you reach a new row (The indicator is on an empty row). Then Right-Click -> Paste.
- Be sure that everything is correct. Press **Control + X** to quit, and then press **Y** to confirm your changes. Press again Enter

- If you did a mistake, you can still press **Control + X** and then **N**
- If it does not work, try adding the other two paths shown in Python Launcher:
  - /opt/local/bin/python3
  - /sw/bin/python3

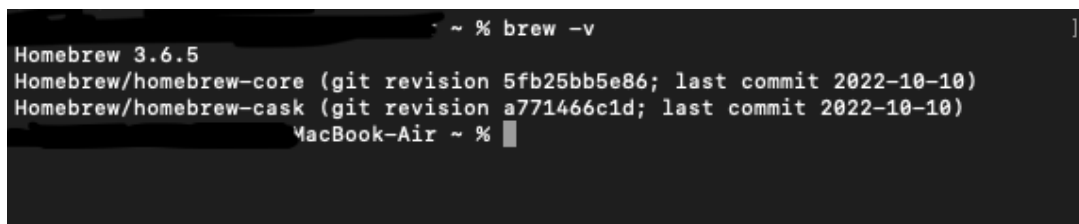## 2) Install Python with Brew (ONLY AS LAST CHOICE!!!!)

Brew is an amazing software. It allows you to simulate "apt-get" on MacOS, and install packages like Python, directly on the Terminal. For programmers, it's really useful. Unfortunately, if you are reading this, it means that everything shown did not work. But worry not! Everything is documented enough.

Install the command line developer tools
- If it's your first time using Terminal for coding, well you have to bear this step too. On Terminal, enter the command "**xcode-select –install"** (without quotation marks). If you already installed the developer tools, you should get an error message and a prompt to update.

Install Homebrew
- Go to https://brew.sh/ and copy the link under "Install Homebrew"
- If you are unsure if the link is the right one, copy this one:
  /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
- Enter your password and wait for the package to be downloaded
- To verify if brew is installed, type **"brew -v".** The result should be like this.


```
~ % brew -v
Homebrew 3.6.5
Homebrew/homebrew-core (git revision 5fb25bb5e86; last commit 2022-10-10)
Homebrew/homebrew-cask (git revision a771466c1d; last commit 2022-10-10)
MacBook-Air ~ %
```

- After checking brew is installed correctly, type "**brew install python"** and wait it to be installed correctly.
- You should now be ready to go! Check again with entering **python3.10** (or the last version downloaded) on Terminal to be absolutely sure everything is working.
- Replace python3 with python3.10 in **macos.sh**
- If for some reason, the env is not recognized, do the same step as before for adding Python to the env variables:
  - Open Terminal, type **sudo nano /etc/paths/**
  - Copy this path: **/usr/local/bin/python3.X** (with X the last version downloaded, mine is 3.10)
  - Down-Key until you are on a empty row, then paste on the empty row
  - **Command + X** and then **Y** to save
- For more info about the Python environment, go here: Homebrew and Python

How to setup TouchPortal

This setup can work with any macro software that allows people to get data from Twitch APIs, after an event, and can connect of course to OBS. If you are interested in how everything works, this is a "simplified" version of the process:

User redeemes the channel reward → OBS + Macro Software get the trigger → Macro Software sends the username of who redeemed the reward to the python file → Python file returns data based on the input → Macro Software sends a message on OBS

Ok, introduction done, let's get to the actual setup. Here I'm using Touch Portal, but again, you can use whatever software you prefer that has these features.

1) Download Touch Portal, and select the version based on your OS. Complete all the steps to install it correctly. Highly suggested to restart your PC.

2) Open Touch Portal, go to **Settings** (Settings Icon top right → Settings) and select **Twitch.** Complete the setup.

3) To be absolutely sure that Touch Portal is communicating correctly with OBS, download Palakis Websocket for OBS. If you have OBS 28 and above, you *don't* have to download it, because it's already installed.

4) If you have the free version of TouchPortal, you have to do this step on a button. If not, you can do it in Global Events. Here is my setup.
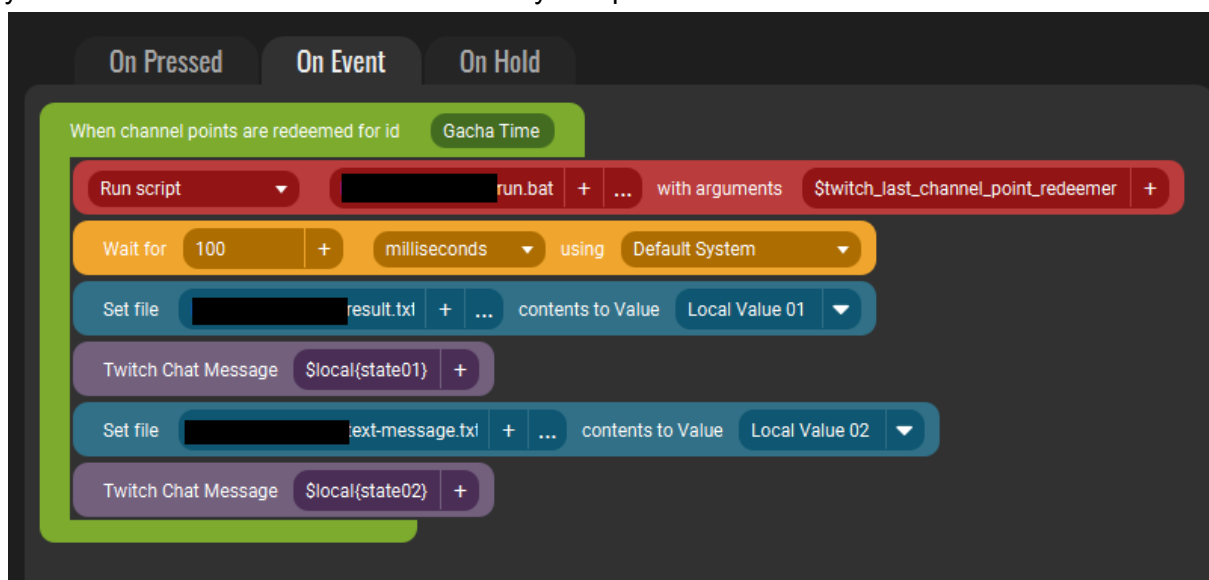
If you don't know how to do that, here are listed all the things you have to search on the searchbar and the values you need to include

On Channel Points Redeemed
- Write the **name of the channel point redeem** (in my case "Gacha Time"). It must be the same, it's CamelCase sensitive

Run Batch File
- If you are using Windows
  - Select **windows.bat** file inside the folder **run-files**
- If you are using macOS
  - Select **macos.sh** file inside the folder **run-files**
  - Just to be sure (you never know with Apple), open **Terminal**, locate the macos.sh file and type **"chmod u+x macos.sh".** This allows TouchPortal to launch the file with all the permissions granted.
- For the Twitch argument, select +, then Twitch → Channel Points → Twitch Last Channel Point Redeemer. You should have $twitch_last_channel_point_redeemer for now.
- Next, hit space and write **avatar** or **gear** based on what you are setting up the event for. So basically
  - **Avatar Gacha:** $twitch_last_channel_point_redeemer avatar
  - **Gear Gacha:** $twitch_last_channel_point_redeemer gear

Wait for Timer
- It's just a safe option for I/O speeds. You can put whatever you want (or remove it)

Files Contents to Value
- Select **result.txt** and **Local Value 01**

Send Chat Message
- Select **+**, then **Local States** → **Local Values** → **Local Value 01**

Do the same for Value02 if you want the "Congrats" message

How to setup StreamElements

Before i start explaining how you have to setup StreamElements, I highly suggest to check out the StreamAvatars documentation to learn how to import and set up characters, set up gears for different avatars and more. I'm assuming you already have avatars available and verified these requirements:
- At least one avatar per rank, and at least one free avatar (so min 4 avatars in total)
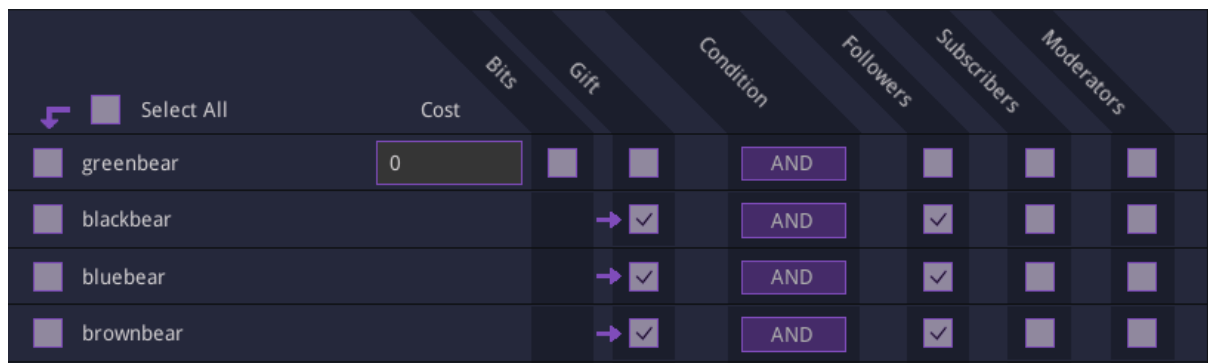- If you are implementing also gears, at least one gear per rank

After these requirements, we can move forward.

0.5) Add avatars. This, of course, it's a must. You can make custom ones or import them through Steam Workshop.

1) Go to **Shop Editing**, and for each character you want to be included in the gacha select **Gift**. You can make them also available only for followers or subscribers if you like.
Remember to leave at least one character free to use. StreamAvatars doesn't allow having no free characters to choose, and will always choose one randomly to make the free one.

In my case, I have (at the moment of writing the document) 25 "bears", aka avatars. Two of them are free to use.

2) Go to the folder of avatars or gear, select the rank file you like the avatar to be and write it down on a new row. So if you want an avatar to be R rank, go to **avatar-tiers/r.txt** and write **exactly** the name of the avatar. CamelCase is ignored, but you have to write the exact name of the avatar, or else it will not work.
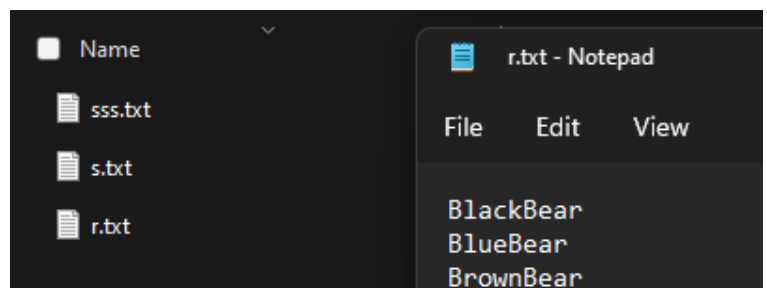
As you can see in the images, I have (example) **bluebear** available on StreamAvatars, and I chose it to be a Rank R bear. So after setting it up correctly on StreamAvatars, i wrote it down on a new row on **r.txt**, text file inside **avatars-tiers.** This is needed because the program will read the data from these files. If no avatar is available (if the file is just empty), you will get an error.

Values you can change inside gacha.py

That's it! You have completed the setup, but you want to change the chances or how many coins a user will get… worry not! The file is structured to be modified easily if you want to change these values! Just as a quick recap, here are the default values (just in case you want to reset).

RANK R %	75
RANK S %	25
RANK SSS %	5
COINS RANK R	500
COINS RANK S	2500
COINS RANK SSS	10000
NUMBER OF REROLLS	5

In order to do that, you can use any text editor (even notepad works just fine!). Personally I use Visual Studio Code. Reach the "Constants" section, and you should see this:

```
39    ####################################    MODIFY HERE    ####################################################
40    # Percentage of X rank characters (example: rank R characters have a 70 percent chance of being pulled)
41    RARITY_R = 70
42    RARITY_S = 25
43    RARITY_SSS = 5
44
45    #Number of coins/points the user will receive if the pull is unsuccessful
46    COIN_R = 500
47    COIN_S = 2500
48    COIN_SSS = 10000
49
50    REROLLS = 5
51    ####################################################################################################
```

This is the **only part of the code** you need to modify.