

# 空投 Bot 使用指南

by\_wazxdec

## 目录

### 0. 前言

### 1. 游戏内需要做的准备工作

### 2. Bot 所需程序的安装与配置

#### a) MCC 的安装与配置

#### b) Python 的安装以及所需程序包的安装

#### c) NapCat 的安装与配置

### 3. Bot 相关配置说明

#### a) 游戏内空投 Bot

#### b) QQ 空投 Bot

### 4. 常见问题 Q&A

该文档完成日期: 2025/11/13

最后一次编辑: 2025/11/13

## 0. 前言

其实在去年的时候我就有想过写一个这样的机器人，但是碍于时间原因再加上原来用的 QQ 框架性能占用过高且近期停止支持，使得我不得不停止了这个机器人的开发。直到我这两天忽然想起了以前没完成的机器人，搜索了几天有关框架的资料，最后搞出了这个机器人。虽然绝大多数代码都是 AI 写的，代码也可能很烂，但是至少能用了（雾）。

在搞这个机器人的时候最难的部分就是与 NapCat 的衔接部分，用于控制 NapCat 发送消息的 Websocket 消息结构复杂，想要实现发送不同的消息非常麻烦。另外，MCC 虽然提供了 Websocket 控制接口，但是并没有给 API，再加上 MCC 给脚本提供的环境并不支持通过 Websocket 进行收发，只能用 TCP 协议和 Bot 联系。

最后也不知道该写啥了，希望各位使用者用的顺利吧（



←瞎画的（（（  
随便看看吧（（（

wazxdec  
2025/11/12

## 1. 游戏内需要做的准备工作

在使用这个机器人前，要先在游戏内做好空投机。只需要能实现刷出来的物品运输到要空投运输矿车中并能自动补充运输矿车即可。接下来用红石把控制空投的按钮（按下即可空投）用红石连接

到一个机器人能够得到的按钮即可，这个按钮的方块坐标要记好，接下来的配置步骤中会用到。

什么??? 你还没有空投机??? 火速去看这篇帖子!!!

[xin 服可使用的自动空投机-仅需附近有玩家挂机即可自动空投 - 教程经验 - 2B2T 中国版 MC](#)

如果你使用了上面的机器，你需要把空投机开关打开，并且将运输矿车走向调整为仓储，仓储满时的设置要调为关闭空投机，并且将仓储旁边的空投按钮（也就是最上面用于空投仓储中的运输矿车的按钮）挖掉，并用红石连到你的机器人能够得到的地方。如果你不想造这么大的机器，并且希望这个机器以存储功能为主，可以使用随着这个机器人附带的末地机器人专供版空投机，仅保留了除蛋、发车功能，这个投影中所有的淡蓝色告示牌在实际建造中均无需放置。需要注意的是，机器人需要控制的按钮旁边最好不要有太多方块，否则会影响空投的触发。

投影 mod 怎么用请自己上网查，这里不提供投影 mod 的使用教程。

## 2. Bot 所需程序的安装与配置

这里提供运行这个 Bot 所需软件的安装与配置。此处仅提供 Windows 系统下的安装与配置，其它系统的配置请自行查询资料。

如果您只想使用游戏内空投 Bot，仅需看 MCC 的安装与配置部分即可。

## a)MCC 的安装与配置

MCC 官网: [主页 | Minecraft 命令行客户端](#)

MCC 下载页面: [Releases · MCCTeam/Minecraft-Console-Client](#)

下载 MCC 之前, 需要先下载.NET 7.0。

下载地址: [下载 .NET 7.0 \(Linux、macOS 和 Windows\) | .NET](#)

下载时请下载右侧的“运行应用 - 运行时”中最下面的“.NET 运行时 7.0.20” (当然其他的也可以, 只不过 MCC 仅需要.NET 运行时 7.0.20 就可以运行), 找到 Windows 一行中安装程序那一列, 点击下载 x64 版本, 绝大多数电脑都需要使用这个版本。如果您的电脑仍然在使用已经过时的 32 位处理器或者正在使用 32 位系统, 请下载 x86 版本。如果希望这个机器人运行在处理器架构为 ARM 的电脑上 (例如使用 Windows 系统的树莓派), 请根据您的系统具体情况进行下载, 32 位系统请选择 arm32 版本, 64 位系统请选择 arm64 版本。此外, 如果您使用的是 Windows 10 ARM, 请下载 arm32 版本, 即使您的处理器支持 64 位指令集。

之后打开 MCC 的下载页面, 找到最新的版本 (在最上面), 根据文件后缀下载与您下载的.NET 7.0 架构相同的版本, 如果是 arm32 请下载后缀为 arm 的版本。另外, 如果下载过慢可以尝试一些 GitHub 镜像加速站, 可能能加快您的下载速度。

下载完成后，双击运行该程序，在该程序同目录下会自动生成一个文件名为 MinecraftClient.ini 的文件。之后关闭 MCC。对于 MCC 的配置文件需要做如下改动：

12~16 行，用户账户配置部分：

```
12 Account = { Login = "", Password = "" }  
    # Login 请填写邮箱或玩家名称。若要以离线模式登录请使用  
    # "-"作为密码。若留空则使用交互式登录。  
13 Server = { Host = "127.0.0.1", Port = 25565 }  
    # 游戏服务器的地址和端口，可填入域名或 IP 地址。（可删  
    # 除端口字段，会自动解析 SRV 记录）  
14 AccountType = "microsoft"  
    # 帐户类型：mojang 或是 microsoft 或是 yggdrasil。此项  
    # 设置也会影响交互式登录。  
15 Method = "mcc"  
    # 微软账户的登录方式：mcc 或是 browser（手动在网页上  
    # 登录）。  
16 AuthServer = { Host = "", Port = 443 }  
    # Yggdrasil API 认证服务器的域名与端口。
```

您需要在第 10 行 Login 后面的引号中填入您机器人的用户名，并在 Password 后面的引号中填入“-”。如果您使用了正版账户作为您的机器人，您需要在 Login 后面的引号中填入您的正版账户绑定

的微软账户的登陆邮箱，并将第 14 行 AccountType 后面引号中的内容改为 microsoft，第 15 行后面引号中的内容改为 browser。在您下次启动 MCC 的时候就会自动打开浏览器并进行微软账户的登录，按照指示登录并将登录后获得的代码粘贴在 MCC 中即可完成登录。如果您选择了正版登录，在登陆完成后会在 MCC 同目录下生成两个文件，分别为 ProfileKeyCache.ini 和 SessionCache.ini，请不要把这两个文件发送给任何人，否则可能会导致您的账户被盗。

您还需要在第 13 行 Host 后面的引号中填入目标 MC 服务器的 IP 地址（例如 2b2t.xin），Port 后面通常留空即可。如果您的目标 MC 服务器 IP 地址格式为 xxx.xxx:12345，请将冒号前的内容填到上述 IP 地址的位置中，冒号后的数字填到 Port 后面。

21~55 行，MCC 功能配置部分：

```
21 Language = "zh_cn"  
    # 请使用 Minecraft 语言代码填写，详见  
    https://mccteam.github.io/r/l-code.html  
22 LoadMccTranslation = true  
    # 在可用时加载应用于 MCC 的翻译，关闭则仅使用英语。
```

这两行通常不会出问题，如果您打开 MCC 后发现语言不是中文，请按照上面的内容修改。

```
27 MinecraftVersion = "1.20.1"
```

```
# 游戏版本, 可使用 "auto"(自动) 或类似 "1.X.X" 的值。设定具体版本将跳过从服务器解析的过程。
```

请将 MinecraftVersion 改为 1.20.1, 使用 auto 会导致版本解析失败导致无法进入服务器, 1.20.1 以上的版本 MCC 的地形处理无法正常运行, 会因数据包错误而被踢出服务器。

```
33 ShowXPBarMessages = true
```

```
# 显示经验条上方的消息, 如果被此类消息刷屏请禁用此选项。
```

在您的机器人排队时控制台可能会被排队人数刷屏, 如果您反感这些消息请设置为 false, 但是同时也将无法显示排队人数。

```
36 TerrainAndMovements = true
```

```
# 开启地形处理将消耗更多的内存、CPU 和网络带宽, 但这允许你进行移动以及和方块交互。
```

```
37 MoveHeadWhileWalking = true
```

```
# 在移动时转向头部。
```

```
38 MovementSpeed = 1
```

```
# 高于 2 的移动速度可能会被检测为作弊。
```

```
39 TemporaryFixBadpacket = true
```

```
# 暂时修复一些服务器上的坏数据包问题。需要先启用 "TerrainAndMovements"。
```

```
40 InventoryHandling = true
```

```
# 启用库存处理（可操作背包、箱子等容器）。
```

请将第 36 行至第 40 行按照上面设置，否则无法与游戏中的方块交互，甚至无法从登录服务器进入主服务器。

一定要将 `TemporaryFixBadpacket` 设置为 `true`，并将 `MovementSpeed` 置为 1，否则都会因为数据包错误而被踢出服务器。

```
55 IgnoreInvalidPlayerName = false
```

```
# 忽略无效的玩家名
```

该项默认为 `true`，旨在过滤掉玩家列表中一些用于展示某些内容的“玩家”。但是在 `xin` 服开启此项会导致无法读取使用中文 ID 的玩家名称，也无法收到使用中文 ID 玩家的消息，因此在该机器人在 `xin` 服使用时请将此项设置为 `false`。

100~102 行，控制台显示部分：

```
100 ConsoleColorMode = "vt100_24bit"
```

```
# 使用“disable”、“legacy_4bit”、“vt100_4bit”、“vt100_8bit”  
或“vt100_24bit”。如果终端上出现“←[0m”等乱码，您可以尝  
试切换到“legacy_4bit”模式，或者直接禁用它。
```

如后面注释所述，出现“←[0m”等乱码时请切换为 `legacy_4bit`。这种情况通常只出现在 Windows 10 以前系统的控制台或者 Windows Server 2016 以前系统的控制台。如果禁用它您将无法在控制台看到某些带有颜色的消息所附带的颜色。

313~318 行，内置 ChatBot 中 AutoReLog 的配置部分

```
313 Enabled = true
314 Delay = { min = 4.0, max = 5.0 }
    # 重新加入到服务器前的延迟时间。(单位：秒)
315 Retries = -1
    # 重新登录服务器失败时的重试次数，使用-1表示无限重试。
316 Ignore_Kick_Message = true
    # 当设置为 true 时，将不考虑踢出的信息直接重连。
317 # 如果踢出信息与这其中的任何一个字符串匹配，那么将
    触发自动重连。
318 Kick_Messages = [ "connection has been lost", "server is
    restarting", "server is full", "too many people", ]
```

这部分请按照上面的内容配置，能保证机器人在因网络问题掉线时自动重连。

452~468 行，内置 ChatBot 中 ScriptScheduler 的配置部分

```
452 Enabled = false
```

这一行控制着这个功能的开关，默认为 false，当该值为 true 时开启。这个功能可以进行一些简单操作，例如在加入服务器后自动完成从登录服进入主服务器的操作，或者定时重载特定的脚本，但是需要您将操作步骤写在一个脚本当中。如果您想使用这个功能，

可以查看 MCC 官方提供的帮助文档: [创建简单脚本 | Minecraft 命](#)

## 命令行客户端

下面是一个 xin 服自动进入主服务器的脚本示例:

```
1 send /l 您的密码
2 wait 10
3 changeslot 3
4 wait 10
5 useitem
6 wait 10
7 inventory 1 click 4
8 inventory 2 click 4
9 inventory 3 click 4
```

实际密码需要根据您在 xin 服注册时设置的密码更改, 如果您使用的是正版账号请删掉这行。将上述内容保存到名为“login.txt”的文本文件中, 并将这个文本文件放在在存储 MCC 程序的文件夹中, 并且将配置文件的 455~460 行如下配置:

```
455 Task_Name = "login"
456 Trigger_On_First_Login = true
457 Trigger_On_Login = true
458 Trigger_On_Times = { Enable = false, Times = [ 14:00:00, ] }
```

```
459 Trigger_On_Interval = { Enable = true, MinTime = 3600.0,  
    MaxTime = 7200.0 }  
460 Action = "script login.txt"
```

这样在每次加入服务器时都能触发这个脚本以进入主服务器。  
另外每 1~2 小时就会触发一次这个脚本，以防意外情况导致机器人留在登录服。

另外，如果您想设置多个触发内容，也可以按照配置文件中的格式添加更多的[[ChatBot.ScriptScheduler.TaskList]], 以实现更多功能。

## b)Python 的安装以及所需程序包的安装

Python 官网: [Welcome to Python.org](https://www.python.org/)

Python 下载页面: [Download Python | Python.org](https://www.python.org/downloads/)

Python 是一种易于学习与编程，功能多样的编程语言，本机器人与 QQ 的衔接部分便使用 Python 编写。

下载 Python 请选择最新的稳定版本。截止到 2025/11/13，最新的稳定版本为 3.14.0。请根据您的系统下载对应的版本，具体该下哪个版本请参考前面的.NET 7.0 的下载。

如果您仍然在使用 Windows 10 以前的系统或者 Windows Server 2016 以前的系统，请下载 Python 3.8.10，在这之后的版本将不再支持这些老旧的系统。如果您仍然在使用 Windows XP 或者

Windows Server 2003 甚至更早的系统，请更新您的系统，Python 3 不再支持这些老旧的系统。（不会都 2025 年了还有人用 XP 吧……）

注意：Python 3.8.10 是一个停止支持的版本，可能会导致程序不稳定甚至可能出现某些漏洞，如果可能请尽量使用更新的系统并使用更新版本的 Python。如果您真的需要 Python 3.8.10，请在下面的链接中下载：[Python Release Python 3.8.10 | Python.org](#)

下载并安装好后，还需要安装 Websocket 相关的程序包以支持程序的运行。打开命令提示符（Win+R 打开运行后输入 cmd 并回车）或者打开 PowerShell（Win+R 打开运行后输入 powershell 并回车），分别输入以下两条命令：

```
1 pip install websockets
2 pip install websocket-client
```

当分别出现提示“Successfully installed websockets-15.0.1”和“Successfully installed websocket-client-1.9.0”时，代表安装成功。

如果您仍然不确定安装是否成功，可以输入以下命令查询已安装的程序包：

```
1 pip list
```

如果在返回的结果中找到了 websockets 和 websocket-client，代表您已经成功安装了这两个程序包。

## c) NapCat 的安装与配置

NapCat 官网: [NapCatQQ | 现代化的基于 NTQQ 的 Bot 协议](#)

## 端实现

NapCat 下载页面: [Releases · NapNeko/NapCatQQ](#)

NapCat 有多种安装方式, 这里仅介绍最简单的安装方式。

其他方式可以查看官网: [NapCat | NapCatQQ](#)

NapCat 通过入侵电脑版 QQ 的程序, 更改 QQ 的源代码以实现外露 QQ 的内部接口, 同时自身提供多种多样的 API 接口以供用户为其编写插件, 在一众想尽一切办法兼容 QQ 登录协议的 QQ Bot 框架中脱颖而出。它易于使用, 无需复杂的登录协议以及各种协议的 Q-Sign 服务器, 也无需各种人机验证, token 抓取, 你所需要做的一切仅仅是拿起手机扫描控制台上二维码。NapCat 兼容 OneBot11 协议, 自身的网络接口使得它的插件可以是任何语言编写的任何程序, 且无需制约于 NapCat 的运行环境。

安装 NapCat 前您需要先安装最新版 QQ: [QQ-轻松做自己](#)

前往 NapCat 的下载页面, 找到最新版本 (在最上面), 找到 NapCat.Shell.Windows.OneKey.zip, 点击并下载, 下载后解压全部文件并运行其中的 NapCatInstaller.exe, 该程序将自动下载必须文件并入侵 QQ 主程序, 注意: NapCat 会先将 QQ 所有程序复制到您解压后的文件夹中, 然后再入侵程序, 并不影响您正常使用 QQ。

待 NapCat 安装完成后, 安装程序会自动关闭。在该文件夹中找到 NapCat.XXXX.Shell, 进入该文件夹后启动 napcat.bat 即可启动

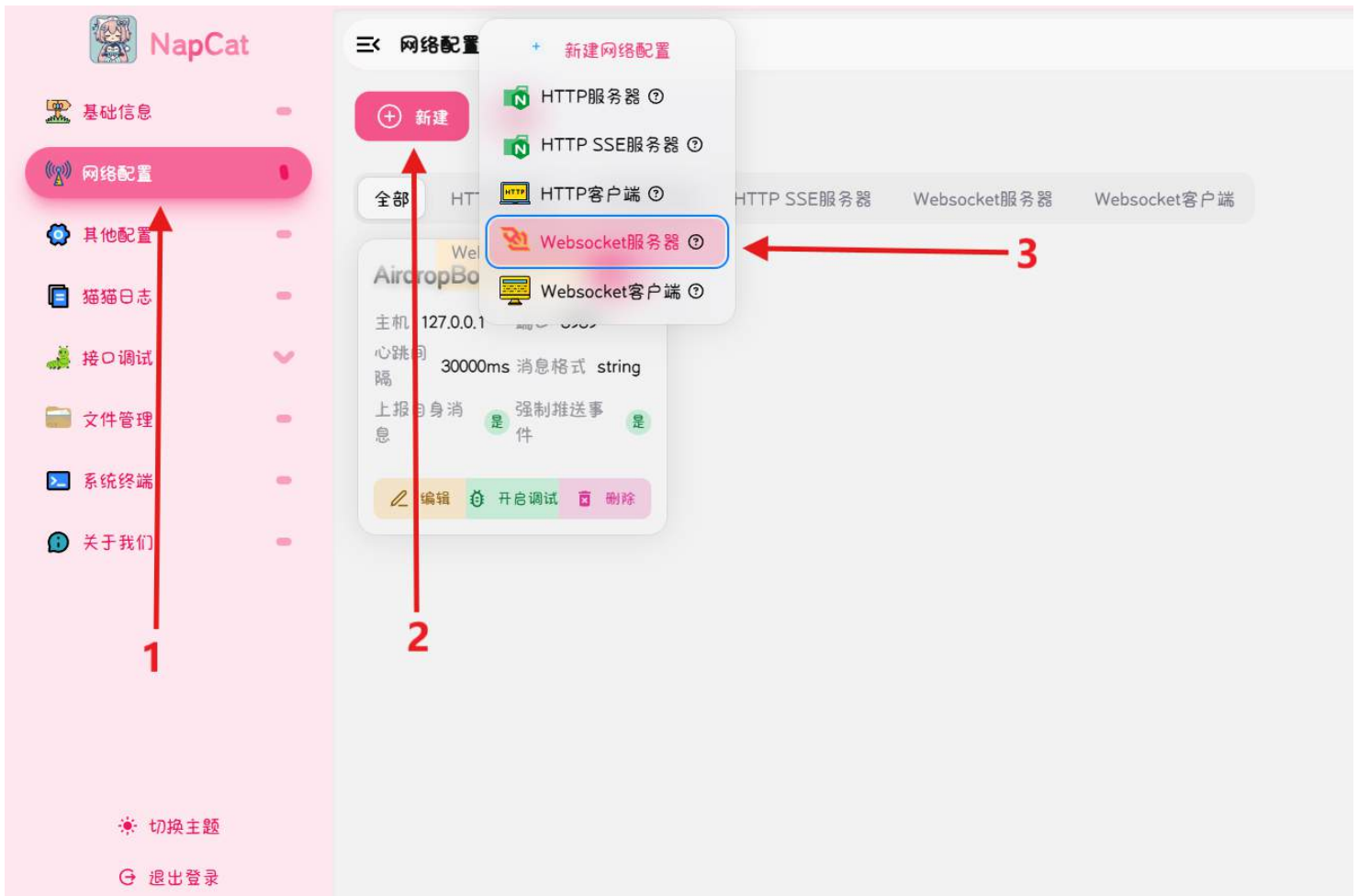
NapCat。启动 NapCat 后控制台会显示一个二维码，如果没有或者无法扫描也可以在路径 `NapCat.XXXX.Shell\versions\X.X.XX-XXXX\resources\app\napcat\cache\qrcode.png` 找到图片形式的二维码。扫码并在手机上确认后即可登录 QQ。

在启动 NapCat 后，可以通过 NapCat 的网页 UI 来配置 NapCat。在路径 `NapCat.XXXX.Shell\versions\X.X.XX-XXXX\resources\app\napcat\config\webui.json` 可以找到 NapCat 网页 UI 的配置文件。需要注意 2~6 行：

```
2     "host": "0.0.0.0",
3     "port": 6099,
4     "token": "" ,
5     "loginRate": 10,
6     "autoLoginAccount": "",
```

port 后面代表网页 UI 的开放端口，默认 6099，以端口为 6099 为例，在浏览器上（注意是运行 NapCat 电脑上的浏览器）输入 `http://localhost:6099/webui?token=xxx` 即可访问 NapCat 网页 UI 并进行配置，后面的 xxx 需要替换为实际配置的 Token。需要注意的是，如果上述配置文件 token 部分留空，则 NapCat 会随机生成 Token，具体 Token 需要在控制台中查看。而 autoLoginAccount 部分可以配置快速登录的账户，后面输入 QQ 号即可，但要求这个 QQ 号在 NapCat 上登录过。

成功访问网页 UI 后即可开始配置 NapCat 的 Websocket 服务器。如下图，在 NapCat 的网页 UI 创建一个 Websocket 服务器：



按照上图的顺序点击，打开后将会是 Websocket 服务器的配置界面。按照右图配置 Websocket 服务器：

Port 部分应当在后续的配置中与机器人配置文件中要连接到的 Websocket 服务器端口保持一致。机器人默认的连接端口为 8989，因此这里以 8989 为例。如果端口不一致会导致机器人与 NapCat 无法通信。



另外，由于机器人并未添加 Token 验证功能，因此 Token 部分要留空，以保证在没有 Token 的情况下也能正常连接。

### 3. Bot 相关配置说明

在做好上述准备工作后，就可以开始配置 Bot 了。Bot 分为两个部分，一个是游戏内的空投 Bot，一个是 QQ 空投 Bot，两个 Bot 之间实际上没有什么关联。如果您只想用游戏内的空投 Bot，仅需阅读有关游戏内空投 Bot 的配置就行了。

#### a) 游戏内空投 Bot

游戏内空投 Bot 的程序是 AirdropBot.cs 这个文件，它并不能直接运行，而是作为 MCC 的插件依赖 MCC 运行。

配置部分为 14~28 行：

```
14 // 配置区域
15 private readonly string TARGET_MESSAGE_PATTERN =
    @"!airdrop"; // 要检测的特定内容
16 private readonly Vector3 TARGET_BLOCK_COORD = new
    Vector3(0, 0, 0); // 触发空投的按钮坐标
17
18 // 时间设置（分钟）
19 private readonly int SUCCESS_PLAYER_COOLDOWN = 30; //
    请求空投玩家的冷却时间
```

```
20 private readonly int OTHER_PLAYERS_COOLDOWN = 15; //  
    其他玩家冷却时间  
21 private readonly int MESSAGE_INTERVAL_MIN = 20; // 不定  
    时消息最小间隔  
22 private readonly int MESSAGE_INTERVAL_MAX = 40; // 不定  
    时消息最大间隔  
23  
24 // 运行时间设置 (24 小时制)  
25 private readonly int START_HOUR = 9; // 开始时间 (小  
    时)  
26 private readonly int START_MINUTE = 0; // 开始时间 (分  
    钟)  
27 private readonly int END_HOUR = 1; // 结束时间 (小时)  
28 private readonly int END_MINUTE = 30; // 结束时间 (分  
    钟)
```

第 15 行@后引号的内容为触发空投 Bot 的内容，当玩家聊天内容（除了自己）含有触发空投 Bot 的内容时就会进行空投，随后进入一段时间的冷却时间，所有玩家都不能触发。触发空投 bot 的玩家会有额外的冷却时间，触发空投 Bot 的玩家冷却时间配置为第 19 行，单位分钟，而其他玩家的冷却时间为第 20 行，单位分钟。第 16 行括号中要填入前面游戏内的准备工作中能触发空投的按钮的方

块坐标。21, 22 行可以配置发送不定时消息的间隔, 21 行后面的数字为最小间隔时间, 22 行后面的数字为最大间隔时间, 单位均为分钟, 这个消息可以填写一些说明消息, 例如使用!airdrop 来获得空投。此外这个机器人还有运行时间功能, 25、26 行能配置开始提供空投的时间, 例如从 9:00 开始空投就把 25 行后面的数字改为 9, 26 行后面的数字改为 0; 27、28 行能配置结束提供空投的时间, 例如 1:30 结束空投就把 27 行后面的数字改为 1, 26 行后面的数字改为 30。

下面是一些消息内容的配置, 注意: 所有消息中都含有 {GenerateRandomSuffix()}这个变量, 这个变量是用来给消息末尾加上三位随机字母或者数字以应对反刷屏用的, 请在更改消息时不要删除这一部分。

第 63 行可以配置定时消息:

```
63 SendDelayedMessage($"在游戏中发送!airdrop 可获得免费末地空投! {GenerateRandomSuffix()}");
```

第 148 行可以配置非服务时间的回复消息:

```
147 // 非服务时间只发送非服务时间消息
148 SendDelayedMessage($"非服务时间! 请 9:00 至次日 1:30 再试! {GenerateRandomSuffix()}");
```

第 170 行可以配置触发空投者冷却时间的回复内容:

```
170 SendDelayedMessage("${playerId}冷却中，请  
    {(int)Math.Ceiling(remainingTime.TotalMinutes)}分钟后再试  
    {GenerateRandomSuffix()}");
```

第 182 行可以配置其他玩家冷却时间的回复内容：

```
182 SendDelayedMessage("${playerId}冷却中，请  
    {(int)Math.Ceiling(globalRemainingTime.TotalMinutes)}分钟后再试  
    {GenerateRandomSuffix()}");
```

这两行代码中有{playerId},  
{(int)Math.Ceiling(remainingTime.TotalMinutes)},  
{(int)Math.Ceiling(globalRemainingTime.TotalMinutes)}三个变量，分  
别代表发送消息的玩家 ID，触发空投玩家的冷却剩余时间，其他  
玩家的冷却剩余时间。编辑消息时可以适当调整三个变量的顺序以  
改变消息内容。

配置完成后将这个脚本放到存储 MCC 程序的目录下，启动  
MCC 后在控制台输入：

```
1 /script AirdropBot.cs
```

即可加载这个插件。如果想卸载这个插件，可在控制台输入：

```
1 /bots unload AirdropBot
```

即可卸载这个插件。

## b)QQ 空投 Bot

在启动前要把 config.json 和 NapCatHandler.py 放到同一个文件夹中。在启动 NapCatHandler.py 这个程序前要配置 config.json。

config.json:

```
01 {
02   "websocket_url": "ws://127.0.0.1:8989",
03   "tcp_server": {
04     "host": "127.0.0.1",
05     "port": 8990
06   },
07   "trigger_message": "!给我空投",
08   "tcp_broadcast_content": "Airdrop",
09   "user_cooling_time": 1800,
10   "global_cooling_time": 900,
11   "non_service_time_start": "01:30",
12   "non_service_time_end": "09:00",
13   "replies": {
14     "normal_reply": "{nickname}已空投，请在末地出生点查收您的空投",
15     "user_cooling_reply": "{nickname}空投正在冷却中，请{remaining_time}分钟后再试！",
```

```
16     "global_cooling_reply": "{nickname}空投正在冷却中，请
    {remaining_time}分钟后再试！ ",
17     "non_service_reply": "非服务时间，请 9:00 后至次日 1:30 前
    再试！ "
18 },
19     "mode": "blacklist",
20     "whitelist": {
21         "users": [ 114514, 114514 ],
22         "groups": [ 114514,114514 ]
23     },
24     "blacklist": {
25         "users": [ 114514,114514 ],
26         "groups": [ 114514,114514 ]
27     },
28     "exceptions": {
29         "users": [ 114514 ],
30         "groups": [ 114514 ]
31     }
32 }
```

配置内容说明：

websocket\_url: 要连接到的 Websocket 服务器, 应当与 NapCat 的 Websocket 地址保持一致。

警告: 如果你不了解这是什么, 请不要随意修改!

tcp\_server: 用于连接 MCC 桥接插件时的 TCP 广播 IP 与端口。这个 IP, 端口应当与 TCPConnectionBot.cs 第 16、17 行相同。

```
16 private string serverHost = "127.0.0.1";  
17 private int serverPort = 8990;
```

警告: 如果你不了解这是什么, 请不要随意修改!

由于 MCC 官方的 Websocket 接口未给出 API, 我只能自己手搓了一个“API”出来, 由于 MCC 内部环境不支持搭建 Websocket 服务器或者客户端, 只能使用 TCP 进行连接。在 MCC 加载了 TCPConnectionBot.cs 这个插件后, 能连接指定的 TCP 服务器, 当 TCP 服务器广播的消息含有特定内容时, 会使用特定方块, 这样就实现了 QQ 空投 Bot 与 MCC 的单向交互。

trigger\_message: 触发空投的 QQ 消息内容, 当检测到特定 QQ 消息时就会触发空投。

tcp\_broadcast\_content: 触发空投时在 TCP 广播的内容, 这个内容应当与 TCPConnectionBot.cs 的第 18 行后面的内容相同。

```
18 private string targetMessage = "Airdrop";
```

user\_cooling\_time: 触发空投的用户的冷却时间, 单位秒。

global\_cooling\_time: 其他用户的冷却时间, 单位秒。

non\_service\_time\_start: 非服务时间开始时间。

non\_service\_time\_end: 非服务时间结束时间。

user\_cooling\_reply: 触发空投的用户冷却时间内的回复内容。

global\_cooling\_reply: 其他用户冷却时间内的回复内容。

注: user\_cooling\_reply, global\_cooling\_reply 中{nickname}为用户昵称, {remaining\_time}为剩余冷却时间, 可以调整变量的位置以调整消息内容。

non\_service\_reply: 非服务时间内的回复消息。

mode: 可选择 blacklist (黑名单), whitelist (白名单) 模式, 下面的 users 和 groups 分别对应黑/白名单内的用户/群聊, 如果有多个用户/群聊可以用半角逗号隔开。

exceptions: 特例用户/群聊。特例中的用户/群聊不会受到冷却时间, 服务时间约束, 在他们发送特定消息时总能触发空投。

最后, 还需要配置 TCPConnectionBot.cs 中 19~21 行的按钮坐标:

```
19 private int blockX = 0;  
20 private int blockY = 0;  
21 private int blockZ = 0;
```

在三行中分别填入按钮的方块的 x 坐标, y 坐标与 z 坐标。

在配置完成后, 需先启动 NapCat, 然后打开 NapCatHandler.py, 最后再从 MCC 加载 TCPConnectionBot.cs。

## 4. 常见问题 Q&A

Q: QQ 总是进入离线模式 (QQ 自动退出登录), 如何解决?

A: 由于 QQ 对于非官方机器人的打击越来越严重, 许多被判定为机器人的账号都会被强制下线甚至封号。同时登录协议也越来越复杂, 这也是为什么很多 QQ 框架在 QQ 更新 QQNT 架构后宣布停止维护。如果有能力开发 QQ 官方机器人尽可能还是去开发官方机器人。所以这个问题暂时没有解决方法, 只能每天登录一遍了。

Q: 有时候 QQ 机器人只会回复消息, 但是不会空投, 如何解决?

A: 可以尝试重载一下 TCPConnectionBot.cs 这个插件, 如果还是不行请检查 QQ Bot 与 MCC 之间的连接以及 QQ Bot 与 NapCat 之间的连接, 启动时如果连接成功会 QQ Bot 会在控制台提示连接成功。

Q: 我用于挂载 NapCat 的电脑性能过差 (主要是一些没有显卡的云电脑), 无法加载 NapCat 的网页 UI, 该如何配置 NapCat?

A: 如果无法加载网页 UI, 可以尝试修改配置文件等其他方式来配置 NapCat。具体可以查看官方文档: [👉 NapCat の WebUI 配置指南👈 | NapCatQQ](#)。

如果看不懂, 也可以尝试内网穿透, 让网页 UI 的渲染工作由性能更好的电脑完成。

以下是几个推荐使用的内网穿透及其使用方法:

## Radmin VPN: [Radmin VPN - Download Free VPN for](#)

### [Windows Now](#)

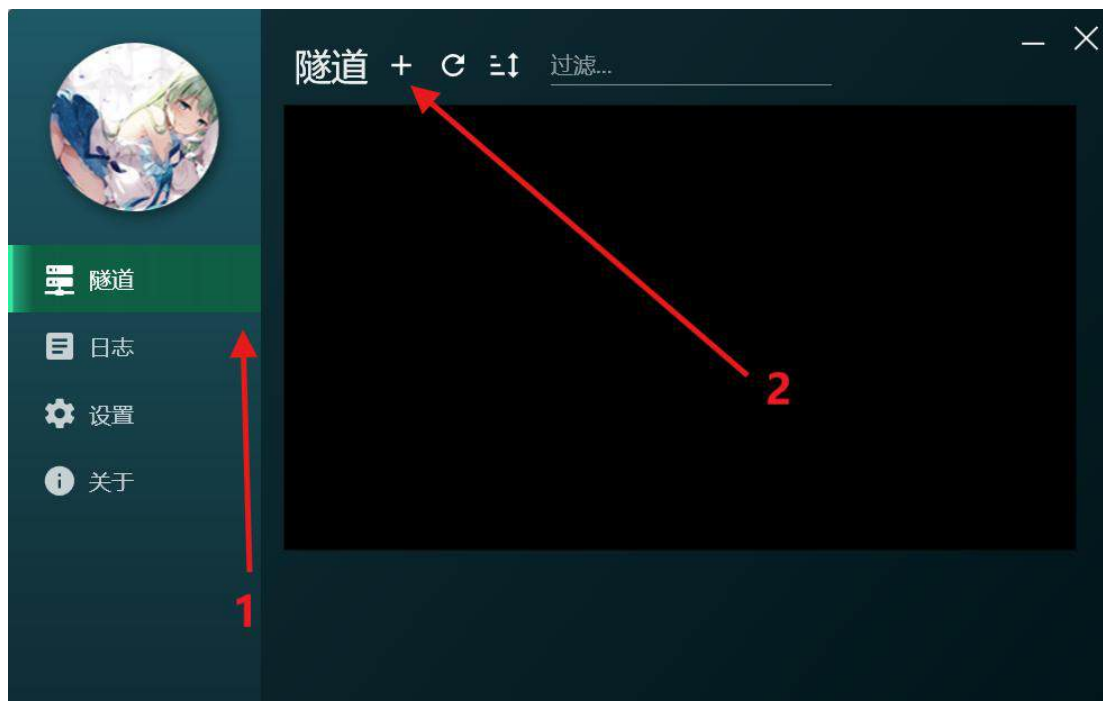
使用方法：首先在挂载 NapCat 的电脑上和性能更好的电脑上都下载并安装 Radmin VPN，在挂载 NapCat 的电脑上打开 Radmin VPN，按照右图的步骤操作：

打开后会提示输入用户名与密码，设置好用户名与密码后再从性能更好的电脑上点击加入网络，输入刚刚设置的用户名与密码，加入成功后会在下面显示这个网络下的其他设备（正常情况下仅会有用于挂载 NapCat 的电脑），记住后面的 IP（这里假设 IP 为 12.34.56.78），在浏览器中输入 `http://12.34.56.78:6099/webui?token=xxx` 并访问，跟上面一样，Token 与后面的端口要根据具体情况更改。



## 樱花穿透 Sakura Frp: [Sakura Frp | 樱花内网穿透](#)

使用方法：在挂载 NapCat 的电脑上下载并安装樱花内网穿透，注册账号并通过登录密钥在 SakuraFrp 启动器上登录。登录完成后按下图操作：



之后找一个距离自己地理位置较近的站点，按下图填写：



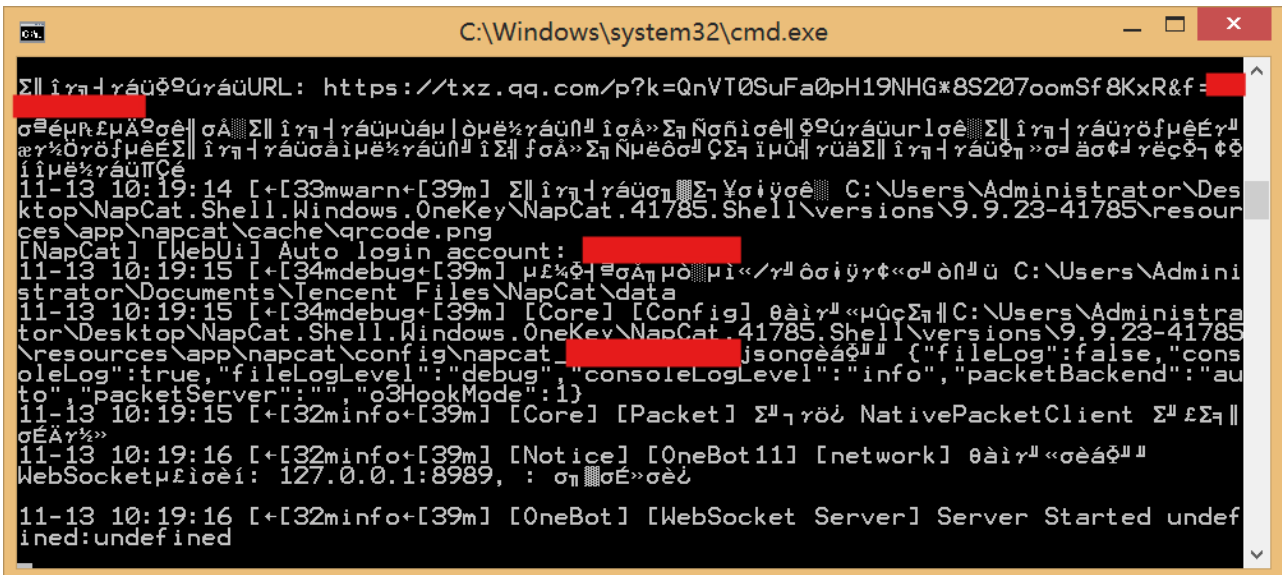
本地端口同样要按照实际情况改变。注意：自动 HTTPS 一定要开，否则无法访问。创建并启用并打开后在日志中查看 IP，之后用

性能较好的电脑访问 `https://frp-xxx:12345/webui?token=xxx`，即可查看 NapCat 的网页 UI。

Q: 如果我想空投混装 kit 空投，有没有什么方法吗？

A: 如果您能读懂 MCC 的简单脚本如何写，可以模拟玩家处理背包的操作来实现 kit 的装填，也可以在游戏中使用投掷器装填，具体实现方法可以自行研究。

Q: 无论怎么更改控制台编码格式都会出现下图乱码怎么办？



A: 如果您遇到了这种乱码，请尝试使用 PowerShell 来运行 NapCat。这种情况常出现在 Windows 10 或者 Windows Server 2016 以下的系统上。如果尝试运行时出现了：“.\NapCatWinBootMain.exe' is not recognized as an internal or external command,operable program or batch file.”，请尝试使用 `cd` 命令将 PowerShell 的运行目录引导到 `NapCat.XXXX.Shell` 文件夹下再尝试。