

McDis-RCON 0.3.0a

Aplicación pensada para retransmitir la consola de Minecraft a Discord.

Diseñada para funcionar en Ubuntu 22.04.3 LTS (puede funcionar en otras versiones) y para

- Servidores: Vanilla
- Networks: Waterfall, Bungeecord o Velocity

Puede funcionar con otros tipos de servidores, pero el sistema de eventos puede no funcionar correctamente. En dicho caso, McDis te permite desactivar el sistema de eventos por defecto e implementar uno personalizado.

COMANDOS EN CMD:

`mcdis init:`

Crea este documento y la `md_config.yml` para poder inicializar McDis-RCON.

`mcdis run:`

Inicializa McDis-RCON en la dirección que te encuentres si es que hay una config valida.

Una vez abierto McDis, este manejará 'Ctrl + c', tratando de cerrar los procesos usando los 'stop_command' y si demoran mucho (máx 60 segs) los cerrará forzosamente y cerrará el código.

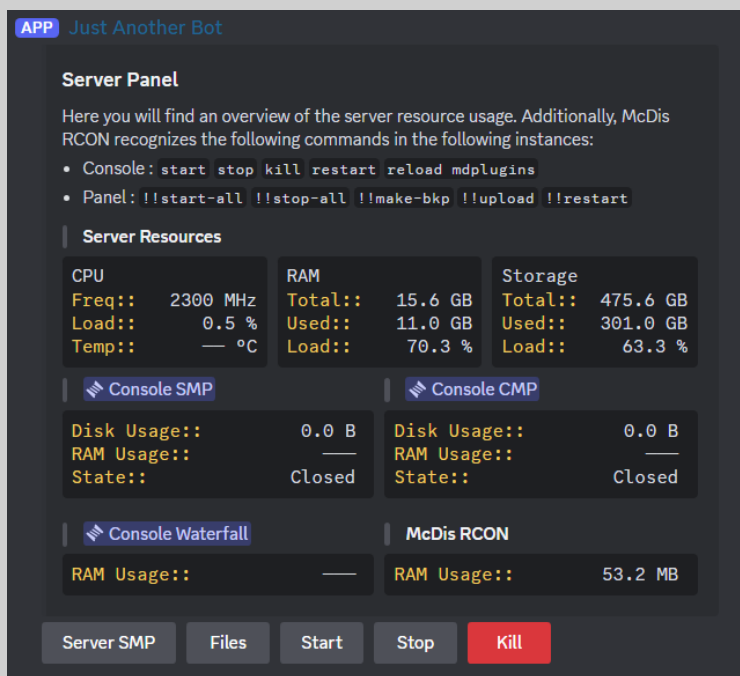
MD_CONFIG.YML:

- Bot Token:

Token del Bot que se usará para retransmitir la consola. Para crear un Bot de Discord existe este tutorial: [\[Link\]](#). De preferencia darle permisos de administrador al bot para que no tenga problemas al crear el panel.

- Panel ID:

ID del canal de Discord donde deseas que se cree el panel, este debe ser un canal de texto sí o sí. Además, el Bot debe tener visibilidad del canal.



```
Bot Token:                               McDis-RCON 0.1.0a
Panel ID:
Language: en
Backups: 3

Booleans:
  omit crash report relay: false

Processes:
  Networks:
    network 1:
      start_command: start
      stop_command: stop
      relay_blacklist:
        - 'example 1'
        - 'example 2'
        - 'example 3'

Servers:
  server 1:
    start_command: start
    mcdreforged: false
    stop_command: stop
    relay_blacklist:
      - 'example 1'
      - 'example 2'
      - 'example 3'

  server 2:
    start_command: start
    mcdreforged: false
    stop_command: stop
    relay_blacklist:
      - 'example 1'
      - 'example 2'
      - 'example 3'
```

- Language:

El idioma en que estará McDis, de momento solo se acepta español 'es' e inglés 'en'.

- Backups:

McDis trae un sistema de backups integrados, este crea un .zip basado en la carpeta del proceso, por defecto McDis almacena un máximo de 3 backups, pero puedes disminuirlo hasta 1 o incrementarlo hasta 5.

- Booleans:

- **Omit crash report relay:** Cuando un servidor crashea, el reporte se imprime en la consola, este es largo por lo que puede evitar que termine el releo. Dándole valor **true** a esta variable, se terminará la retransmisión cuando el crash suceda.

- **Upload overwrite:** Si el comando **!!upload** debe sobrescribir archivos.

- **Allow flask:** Discord solo te permite subir archivos de hasta cierto tamaño, usando flask puedes solicitar archivos directamente al servidor, esto elimina el límite al tamaño de las solicitudes.

- Flask:

Si 'allow flask' es true, se necesitará que proveas una ip y un puerto. La ip debe la dirección de tu dedicado y el puerto debe ser uno que tengas abierto. (Más detalles de cómo funciona FLASK al final del pdf).

- Processes:

start_command comando de inicio del proceso

stop_command comando de cierre del proceso.

relay_blacklist lista de términos que de contenerse en log, no se retransmitirán.

- **Networks:** Procesos del tipo Waterfall, Bungeecord o Velocity. Puedes tener más de uno. En mi caso yo uso Waterfall, este es un ejemplo de cómo lo uso.

```
Processes:
  Networks:
    Waterfall:
      start_command: java -Dfile.encoding=UTF-8 -jar launcher.jar nogui
      stop_command: end
      relay_blacklist:
        - '<->'
        - '<-'
        - '->'
```

- **Servers:** Procesos de tipo servidor, ya sean con MCDReforged o no, testeado solo con servidores 1.17.1, 1.20.1, 1.20.4, aunque seguramente funciona con las versiones intermedias. En mi caso yo uso dos servidores, SMP Y CMP, pueden agregar cuantos necesiten.

Si usas MCDReforged, no es necesario que el comando de inicio sea 'mcdreforged', solo edita la variable del mismo nombre y en 'start_command' pon el comando al .jar, McDis modificará la config de MCDR.

```
Servers:
SMP:
  start_command: java -Xms6G -Xmx6G -Dfile.encoding=UTF-8 -jar launcher.jar nogui
  mcdreforged: false
  stop_command: stop
  relay_blacklist:
    - 'example 1'
    - 'example 2'
    - 'example 3'

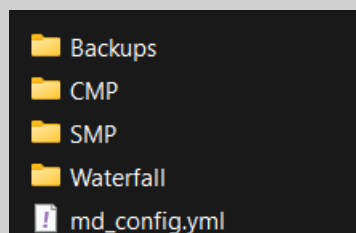
CMP:
  start_command: java -Xms6G -Xmx6G -Dfile.encoding=UTF-8 -jar launcher.jar nogui
  mcdreforged: false
  stop_command: stop
  relay_blacklist:
    - 'example 1'
    - 'example 2'
    - 'example 3'
```

ESTRUCTURA DE ARCHIVOS:

Una vez definida la config, corriendo 'mcdis run' se crearán otras carpetas, en mi caso:

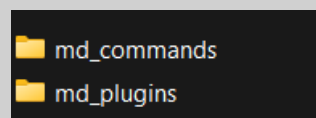
- Backups contendrá los backups que realices con McDis.
- CMP contiene todos los archivos del proceso CMP.
- SMP contiene todos los archivos del proceso SMP.
- Waterfall contiene todos los archivos del proceso Waterfall.

Estas carpetas pueden ser creadas de antemano.



McDis también creará carpetas 'md_commands' y 'md_plugins' en las carpetas de servidores.

- md_commands destinada a contener comandos predefinidos.
- md_plugins destinada a contener los plugins que instales.



RUTAS DE ARCHIVOS:

McDis llamará a la carpeta donde este se ejecute como 'McDis', no la renombrará, solo se referirá a ella de esa forma. En ese sentido, por ejemplo, la carpeta md_plugins en SMP, tendría un mcdis_path: McDis/SMP/md_plugins.

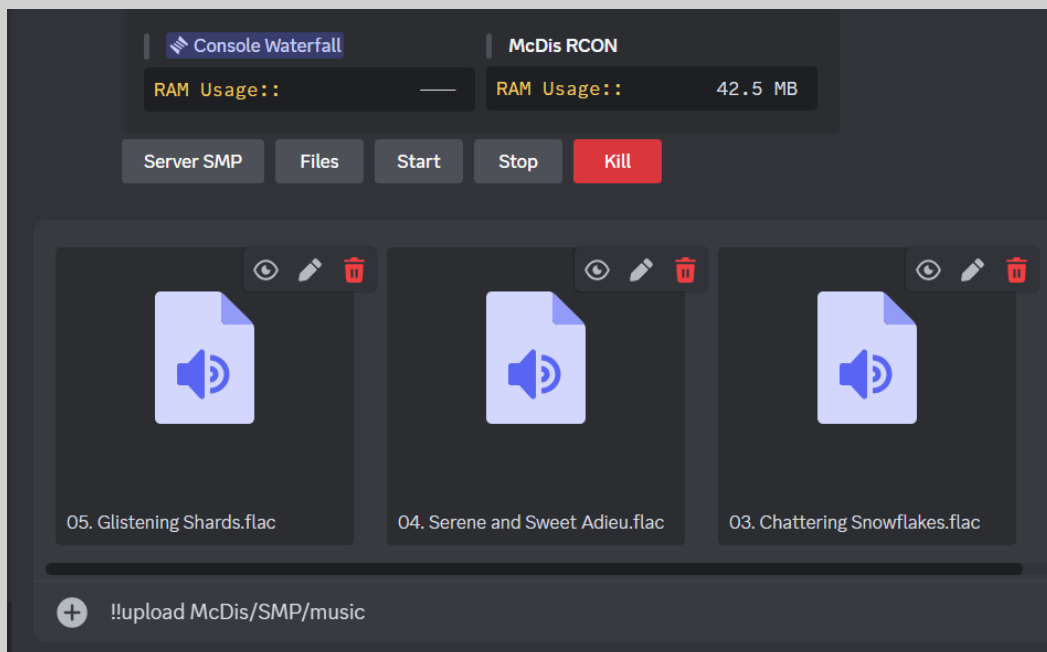
PANEL:

McDis reconoce comandos en diferentes instancias

- Console: `start stop kill restart reload mdplugins`
- Panel: `!!start-all !!stop-all !!make-bkp !!upload !!restart`

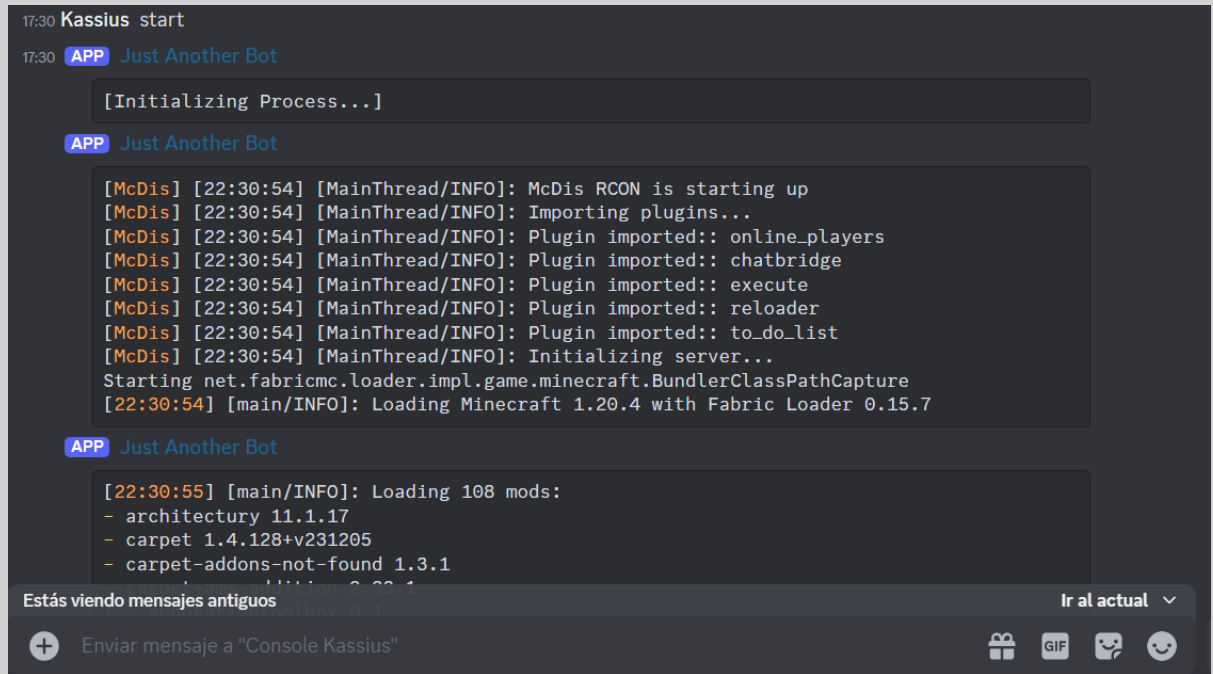
- Panel:

- `!!restart`:
Reinicia McDis, esto cerrará por completo el código y lo volverá a abrir, o sea que los procesos que tenga abiertos McDis, también se cerrarán. McDis tratará de cerrarlos con sus respectivos 'stop_commands' y si demoran demasiado tiempo en cerrar los cerrará forzosamente, (máx 60 segs).
- `!!start-all`:
Inicializa todos los procesos que tengas definidos.
- `!!stop-all`:
Ejecuta el stop_command en todas las consolas.
- `!!make-bkp <name>`:
Crea un backup del proceso especificado, <name>.
Ej.: `!!make-bkp SMP`
- `!!upload <mcdis_path>`:
Sube archivos que envíes adjuntos al mensaje a una dirección relativa, <mcdis_path>.



- Consolas:

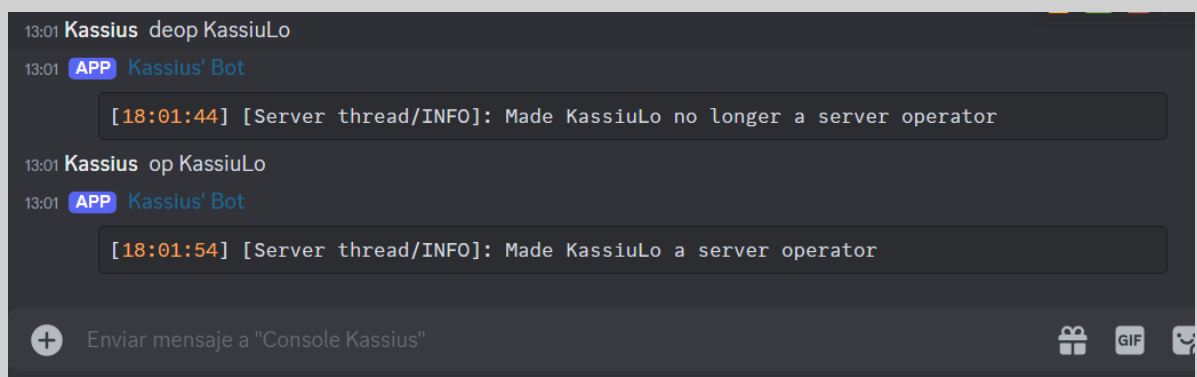
- start:
Ejecuta el 'start_command' en la consola.
- stop:
Ejecuta el 'stop_command' en la consola.
- kill:
Cierra forzosamente el proceso.
- restart:
Reinicia el servidor usando el 'stop_command' y una vez cerrado lo vuelve a abrir.
- reload_mdplugins:
Lo que hará es volver a importar los plugins en md_plugins.



The screenshot shows a Discord chat window with a dark theme. At the top, a message from 'Kassius' at 17:30 says 'start'. Below it, a message from 'Just Another Bot' (labeled 'APP') at 17:30 says '[Initializing Process...]'. This is followed by a large code block containing a log from 'McDis' at 22:30:54. The log shows the server starting up, importing plugins (online_players, chatbridge, execute, reloader, to_do_list), and initializing the server. It also shows the loading of 108 mods, including 'architecture 11.1.17', 'carpet 1.4.128+v231205', and 'carpet-addons-not-found 1.3.1'. At the bottom of the chat, there is a message from 'Kassius' at 17:30 saying 'stop'. The chat interface includes a search bar, a message input field with a plus icon, and a 'Enviar mensaje a "Console Kassius"' button. There are also icons for GIFs, emojis, and a 'Ir al actual' dropdown menu.

```
17:30 Kassius start
17:30 APP Just Another Bot
[Initializing Process...]
APP Just Another Bot
[McDis] [22:30:54] [MainThread/INFO]: McDis RCON is starting up
[McDis] [22:30:54] [MainThread/INFO]: Importing plugins...
[McDis] [22:30:54] [MainThread/INFO]: Plugin imported:: online_players
[McDis] [22:30:54] [MainThread/INFO]: Plugin imported:: chatbridge
[McDis] [22:30:54] [MainThread/INFO]: Plugin imported:: execute
[McDis] [22:30:54] [MainThread/INFO]: Plugin imported:: reloader
[McDis] [22:30:54] [MainThread/INFO]: Plugin imported:: to_do_list
[McDis] [22:30:54] [MainThread/INFO]: Initializing server...
Starting net.fabricmc.loader.impl.game.minecraft.BundlerClassPathCapture
[22:30:54] [main/INFO]: Loading Minecraft 1.20.4 with Fabric Loader 0.15.7
APP Just Another Bot
[22:30:55] [main/INFO]: Loading 108 mods:
- architecture 11.1.17
- carpet 1.4.128+v231205
- carpet-addons-not-found 1.3.1
17:30 Kassius stop
```

De ahí en más, siempre que el servidor esté abierto, el hilo de la consola funciona como la consola misma.

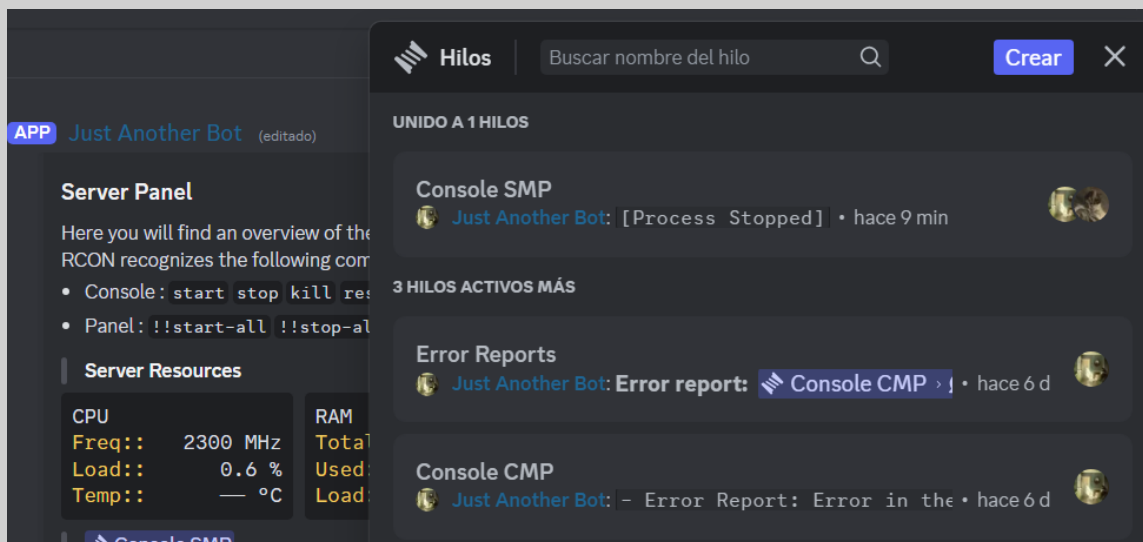


The screenshot shows a Discord chat window with a dark theme. At the top, a message from 'Kassius' at 13:01 says 'deop KassiuLo'. Below it, a message from 'Kassius' Bot (labeled 'APP') at 13:01 says '[18:01:44] [Server thread/INFO]: Made KassiuLo no longer a server operator'. This is followed by a message from 'Kassius' at 13:01 saying 'op KassiuLo'. Below that, another message from 'Kassius' Bot (labeled 'APP') at 13:01 says '[18:01:54] [Server thread/INFO]: Made KassiuLo a server operator'. At the bottom of the chat, there is a message from 'Kassius' at 13:01 saying 'stop'. The chat interface includes a search bar, a message input field with a plus icon, and a 'Enviar mensaje a "Console Kassius"' button. There are also icons for GIFs, emojis, and a 'Ir al actual' dropdown menu.

```
13:01 Kassius deop KassiuLo
13:01 APP Kassius' Bot
[18:01:44] [Server thread/INFO]: Made KassiuLo no longer a server operator
13:01 Kassius op KassiuLo
13:01 APP Kassius' Bot
[18:01:54] [Server thread/INFO]: Made KassiuLo a server operator
13:01 Kassius stop
```

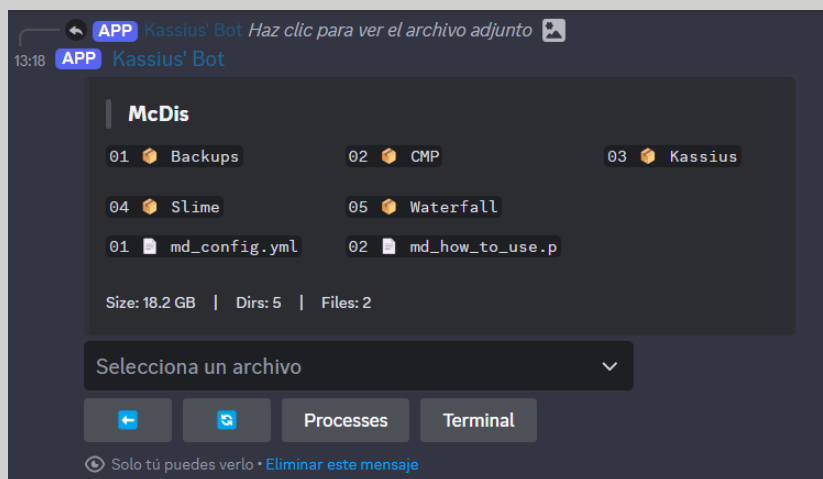
ERRORES:

Por lo general, si hay algún error mientras se manejan eventos o funciones, McDis creará un reporte en la misma consola y enviará una referencia al hilo ‘Error reports’ del canal del panel.



FILES MANAGER:

McDis también incluye un gestor de archivos que navega en rutas relativas a McDis. Al darle a ‘Files’ en el panel se desplegará una interfaz como esta



Este gestor de archivos te permite navegar entre carpetas, editar archivos, renombrarlos, copiarlos, moverlos, crear nuevas carpetas, borrarlos, comprimir carpetas, etc (Pero todo dentro de McDis). Al darle al botón Terminal puedes ejecutar comandos para ello de manera similar a una consola:

Ej: con los archivos mostrados en la imagen anterior

Renombrar el archivo 'md_config.yml'

```
-> rename file:01 new_name.yml
```

Renombrar la carpeta 'Waterfall'

```
-> rename dir:05 new_name
```

También si entras a un archivo, si este es de texto, McDis te permite editarlo si las limitaciones de discord no se lo impiden.

Terminal

Este formulario se enviará a Kassius' Bot. No compartas contraseñas ni ningún tipo de información confidencial.

> MCDIS *

4000

COMANDOS *

mkdir <name>
cd <dir:index | file:index>
del <dir:index | file:index>
copy <dir:index | file:index> <mcdis_path>
move <dir:index | file:index> <mcdis_path>
rename <dir:index | file:index> <new_name>

3802

Cancelar

Enviar

Edita el archivo

Este formulario se enviará a Kassius' Bot. No compartas contraseñas ni ningún tipo de información confidencial.

ARCHIVO *

md_config.yml

MD_CONFIG.YML *

=====
McDis-RCON Configuration
=====

Designed for
- Servers : Vanilla, Fabric
- Networks : Waterfall, Bungeecord, or Velocity

start_command : Start command for the process, example: java -Xms6G -Xmx6G -Dfile.encoding=UTF-8 -jar launcher.jar nogui
stop_command : Command to stop the process; if left blank, the process will be terminated abruptly
relay_blacklist : List of terms; if contained in the log, it will not be sent to Discord

Even with MCDReformed, set the .jar start command: McDis-Boon will edit the MCDReformed

Cancelar

Enviar

COMMANDS:

En cada proceso del tipo de servidor se da la opción de tener almacenados secuencias de comandos predefinidos, para eso navega hasta md_commands de un servidor y se te mostrará una interfaz, ahí podrás crear comandos nuevos, o revisar comandos ya creados.

McDis\SMP\md_commands

In the dropdown below, you will find various predefined commands. Select the one you wish to use.

Dropdown:

Select a command

[New Command]

example_command

example_command

Description:


Your description here.


Action 1:


command 1

command 2

command 3

Use  to iterate over the options.





Execute

Edit

Delete

Solo tú puedes verlo

• [Eliminar este mensaje](#)

BACKUPS:

Los backups creados con !!make-bkp se guardan en la carpeta McDis/Backups, si navegas hasta ahí se te mostrará una interfaz, ahí podrás seleccionar un backup y decidir si cargarlo o borrarlo.

APP Just Another Bot (editado)

McDis\Backups\CMP

1. CMP 1.zip

Disk Usage::

334.0 B

Date::

2024-10-21 13:34:18 (UTC -05:00)

2. CMP 2.zip

Disk Usage::

334.0 B

Date::

2024-10-21 13:34:18 (UTC -05:00)

3. CMP 3.zip

Disk Usage::


334.0 B


Date::


2024-10-21 13:34:18 (UTC -05:00)

If you want to load or delete a backup, select it from the dropdown below.

Select a backup







Solo tú puedes verlo

• [Eliminar este mensaje](#)

PLUGINS:

McDis maneja un sistema de eventos lo que permite que si dentro de un .py hay funciones con ciertos nombres específicos estas sean llamadas cuando ocurren dichos eventos. Además McDis permite la implementación de eventos personalizados e incluso desactivar el sistema de eventos por defecto. Sobre esto daré más explicaciones en un futuro.

Por ejemplo:

```
response = lambda player, message: f'tellraw {player} [{"text": "{message}","color":"gray"}]'\n\nasync def help(self, player: str, message: str):\n    self.execute(response(player, "$6!!$freload mdplugins"))\n    self.execute(response(player, "  ↳ Recargar los mdplugins."))\n\nasync def on_player_command(self, player: str, message: str):\n\n    if message.startswith(f'!!reload mdplugins'):\n        self.unload_plugins()\n        self.load_plugins(reload = True)\n\n        self.execute(response(player, "✓ mdplugins recargados."))
```

FLASK:

Es un pequeño servidor que está incorporado en el código para poder realizar descargas de archivos con tamaños mayores a 5 MB, para poder utilizar flask debes abrir un puerto para permitir el tráfico a través del mismo. Para poder descargar un archivo debes navegar hasta dicho archivo usando McDis y pulsar 'Request', si 'allow flask' es true en la config, se generará un link para poder descargar el archivo. Este link durará 60 segundos y es de uso único, o sea, que si abres el link y empieza la descarga ya no podrás volver a utilizarlo, y si no abres el link cuando lo solicitas, este dejará de ser válido pasados 60 segundos.