The script

host_monitor_by_user.py

is a Telegram bot designed to monitor remote hosts by periodically pinging them. It leverages the Telegram Bot Framework and includes several functionalities to manage and monitor hosts on a per-user basis. Below is a detailed analysis of the script:

Overview

- **Purpose**: The bot monitors remote hosts by sending ping commands and reports their status to users.
- **Version**: `0.2.1`
- **Dependencies**: `os`, `platform`, `time`, `asyncio`, `dotenv`, `telegram` modules.

Key Components

Class `HostMonitorBot`

This class inherits from `TlgBotFwk` and extends its functionality to include host monitoring features.

- **Attributes**:
  - `jobs`: A dictionary to store jobs for each user.
  - `external_post_init`: A method to load all user data after initialization.
- **Methods**:
  - `__init__(self, show_success=False, token=None, *args, **kwargs)`: Initializes the bot, sets up attributes, and loads user data.
  - `async def get_user_data(self, user_id: int, user_item_name: str, default_value=None)`: Retrieves user-specific data from persistence.
  - `async def load_all_user_data(self)`: Loads all user data from persistence.
  - `async def job_event_handler(self, callback_context: CallbackContext)`: Handles job events, pings the host, and sends status messages.
  - `def ping_host(self, ip_address, show_success=True, user_id=None)`: Executes the ping command and sends the result to the user.
  - `async def add_job(self, update: Update, context: CallbackContext)`: Adds a new ping job for a user.
  - `async def delete_job(self, update: Update, context: CallbackContext)`: Deletes an existing job.
  - `async def list_jobs(self, update: Update, context: CallbackContext)`: Lists all jobs for the current user.
  - `async def list_all_jobs(self, update: Update, context: CallbackContext)`: Lists all jobs in the job queue.
  - `async def toggle_success(self, update: Update, context: CallbackContext)`: Toggles the success message flag.
  - `def run(self)`: Starts the bot's main loop.

Script Execution

1. **Environment Setup**:
   - Loads environment variables from

my.env

using `dotenv`.

- Retrieves the bot token from the environment variables.

2. **Bot Initialization**:

- Creates an instance of `HostMonitorBot` with the retrieved token.

3. **Bot Execution**:
   - Starts the bot's main loop by calling `bot.run()`.

Example Usage

- **Adding a Job**:
  - Users can add a job using the `/addjob <ip_address> <interval_in_seconds>` command.
  - The bot will periodically ping the specified IP address at the given interval and report the status.
- **Deleting a Job**:
  - Users can delete a job using the `/deletejob <job_name>` command.
- **Listing Jobs**:
  - Users can list their jobs using the `/listjobs` command.
  - Admins can list all jobs using the `/listalljobs` command.

Error Handling

- The bot includes error handling in various methods to log errors and send error messages to the bot owner.

Conclusion

The

host_monitor_by_user.py

script is a comprehensive Telegram bot for monitoring remote hosts via ping commands. It supports multiple users, allowing each user to manage their own set of monitored hosts. The bot provides commands to add, delete, and list jobs, making it a useful tool for network monitoring and management.