Table of Contents of Core Bot Setup and Configuration

Imports and Libraries:

`discord`, `discord.ext.commands`, `requests`, `datetime`, `bs4 (BeautifulSoup)`, `io (BytesIO)`, `matplotlib`, `sharppy`, `cartopy`, `os`, `json`, `logging`, `geopy (Nominatim)`, `astropy`, `timezonefinder`, `ephem`, `Pillow (PIL)`

Configuration (config.py):

`DISCORD_TOKEN` (environment variable)

Data source URLs (NWS, SPC, GOES, etc.)

Default values (region, products, hours before now)

Bot settings (command prefix, status message)

Sounding models (NAM, GFS, HRRR, RAP)

RAP products (fill in from CoD website)

Geocoding settings (provider, API keys)

Other API keys (placeholders)

Map overlay settings (icon paths)

Logging Setup:

Configures logging to `weather_bot.log` file.

Bot Initialization:

Creates the bot instance with the command prefix.

Data Caching:

Initializes caches for METARs, TAFs, and alerts.

`load_cache`, `save_cache` functions for managing cached data in JSON files.

Weather Commands

`$metar <airport_code> [hours_ago]`

Fetches current or historical METARs from the Aviation Weather Center.

Caches recent METARs for faster responses.

Handles optional `hours_ago` argument for past METARs.

Extensive error handling and logging.

`$taf <airport_code>`

EVAN J LANE May 25, 2024 22:02:41

Fetches TAFs from the Aviation Weather Center.

Caches recent TAFs for faster responses.

Error handling for invalid airport codes or data issues.

`$skewt observed <station_code>`

Fetches observed sounding data from the SPC website and generates a Skew-T diagram.

Users can specify the desired model (default: NAM).

Adds a logo to the Skew-T plot.

Error handling for invalid inputs or data issues.

`$skewt fcst <model> <forecast_hour> <station_code>`

Fetches forecast sounding data and generates a Skew-T diagram.

Supports NAM, GFS, HRRR (with forecast hour), and potentially RAP models.

Adds a logo to the plot.

Error handling for invalid input or data issues.

`$sat <region> <product_code>`

Fetches GOES-16 satellite imagery for various regions (CONUS, full disk, mesosectors, tropical Atlantic, tropical Pacific).

Supports different product codes (GeoColor, visible, infrared, water vapor) for each region.

Error handling for invalid inputs or data issues.

Adds a logo to the satellite image.

Additional Commands

`$astro <location>`

Provides sunrise, sunset, and moon phase information for a given location.

Uses geocoding (`geopy`) to get coordinates.

Calculates astronomy data using `PyEphem`.

Error handling for invalid locations or calculation errors.

`$radar <location>`

Fetches radar imagery from the NWS for a given location or the user's location (if set).

Uses geocoding (`geopy`) to get coordinates.

Adds a marker to the map at the specified location.

Includes basic map features (coastlines, state borders).

Error handling for data retrieval or geocoding issues.

`$discussion <office_code>`

Fetches the forecast discussion from the SPC website for a given office code (e.g., "FFC").

Parses the discussion text and sends it in a code block format.

Error handling for invalid office codes or data retrieval issues.

Placeholder Commands (Not Yet Implemented):

```
$alerts <location>
$model <model_name> <parameter> <location>
$lightning <region>
$webcam <location>
```

Helper Functions

`add_map_overlay(ax, lat, lon, icon_path, logo_path, zoom)`

Adds a location marker (if latitude and longitude are provided) and a logo to a map image.

Used in the `$radar` and `$skewt` commands.

`load_cache(cache_type)`

Loads cached data (METARs, TAFs, or alerts) from a JSON file.

`Save_cache(cache_type, data)`

Saves data to a JSON file with indentation for readability.

Help Command (New)

`$help`

Displays a list of available commands and their basic usage.

Provides instructions on how to get more detailed help (e.g., by referring to documentation in a specific channel).

Running the Bot

`bot.run(config.DISCORD_TOKEN)` Starts the bot.

ohh

EVAN J LANE May 25, 2024 22:02:41