Contest Log Analytics - Installation Guide

Version: 1.0.0-alpha.1 Date: 2026-01-13

--- Revision History ---

[1.0.0-alpha.1] - 2026-01-13

Changed

- Updated version to match project release versioning.

- Enhanced Docker installation section with detailed information about:

* Data files location and requirements

* Directory structure expectations

* Automatic environment variable configuration

* Note about included sample data files

[0.126.0-Beta] - 2025-12-18

Changed

- Promoted the Docker installation workflow to the primary "Installation Steps".

- Demoted the existing Conda/CLI instructions to "Advanced / Developer Setup".

[0.94.1-Beta] - 2025-12-07

Changed

- Added `python-kaleido` to the required libraries in Step 3 to support

static image export for Plotly reports.

[0.94.0-Beta] - 2025-12-06

Changed

- Added `plotly` to the required libraries in Step 3 to support the

Phase 2 visualization engine upgrade.

[0.91.15-Beta] - 2025-10-11

Fixed

- Corrected data file dependencies to state that

For Web Dashboard (Preferred)

- **Docker Desktop**: The application runs inside a container, requiring no local Python or library installation.

For CLI / Development (Advanced)

- **Git:** For cloning the source code repository.
- **Miniforge:** This is the recommended way to install Python and manage the project's libraries in an isolated environment. Miniforge is a minimal installer for the Conda package manager.

---

2. Installation Steps

Method A: Web Dashboard (Docker)

This is the easiest way to get started. Docker automatically handles all dependencies and environment configuration.

1. **Clone the Repository:**

   ```
   git clone https://github.com/user/Contest-Log-Analyzer.git
   cd Contest-Log-Analyzer
   ```

2. **Verify Data Files:** The repository includes sample data files in `CONTEST_LOGS_REPORTS/data/`. These are sufficient for initial testing. The required data files include:

   - `cty.dat`: Required for all contests.
   - `CQ160mults.dat`: Required for the CQ 160-Meter contest.
   - `arrl_10_mults.dat`: Required for the ARRL 10 Meter contest.
   - `ARRLDXmults.dat`: Required for the ARRL DX contest.
   - `NAQPmults.dat`: Required for NAQP contests.
   - `SweepstakesSections.dat`: Required for ARRL Sweepstakes and ARRL Field Day.
   - `band_allocations.dat`: Required for all contests to perform frequency validation.
   - `iaru_officials.dat`: Required for the IARU HF World Championship contest.

   Note: If you need to add your own data files, place them in `CONTEST_LOGS_REPORTS/data/` before starting Docker.

3. **Start the Application:**

   ```
   docker-compose up --build
   ```

   Docker automatically configures the environment variables:

   - `CONTEST_INPUT_DIR=/app/CONTEST_LOGS_REPORTS`

- `CONTEST_REPORTS_DIR=/app/CONTEST_LOGS_REPORTS`

The application expects:

- Data files in: `CONTEST_LOGS_REPORTS/data/`
- Log files in: `CONTEST_LOGS_REPORTS/Logs/`
- Reports generated in: `CONTEST_LOGS_REPORTS/reports/`

4. **Access the Dashboard:** Open your web browser and navigate to `http://localhost:8000`.

Method B: Advanced / Developer Setup (CLI)

Use this method if you are developing the code or prefer the command line.

Step 1: Clone the Repository Open a terminal or command prompt, navigate to the directory where you want to store the project, and clone the remote Git repository. **CODE_BLOCK** git clone https://github.com/user/Contest-Log-Analyzer.git cd Contest-Log-Analyzer **CODE_BLOCK** This will create the project directory (`Contest-Log-Analyzer`) on your local machine.

Step 2: Create and Activate the Conda Environment It is a best practice to create an isolated environment for the project's dependencies. This prevents conflicts with other Python projects on your system. **CODE_BLOCK**

Create an environment named ”cla” with Python 3.11

conda create --name cla python=3.11

Activate the new environment

conda activate cla **CODE_BLOCK**

Step 3: Install Libraries with Conda With the `cla` environment active, use the following single command to install all required libraries from the recommended `conda-forge` channel. This includes `plotly` for interactive charts. **CODE_BLOCK** conda install -c conda-forge pandas numpy matplotlib seaborn imageio prettytable tabulate requests beautifulsoup4 plotly **CODE_BLOCK**

Step 4: Set Up the Input and Output Directories The application requires separate directories for its input files (logs, data) and its output files (reports). This separation is critical to prevent file-locking issues with cloud sync services.

1. Create the Input Directory: This folder will contain your log files and required data files. It can be located anywhere, including inside a cloud-synced folder like OneDrive. Example: `C:\Users\YourUser\OneDrive\Desktop\CLA_Inputs` Inside this folder, you must create the following subdirectories: **CODE_BLOCK** CLA_Inputs/ | +-- data/ | +-- Logs/ **CODE_BLOCK**

2. Create the Output Directory: This folder is where the analyzer will save all generated reports. **This directory must be on a local, non-synced path.** A recommended location is in your user profile directory. Example: `%USERPROFILE%\HamRadio\CLA` (which translates to `C:\Users\YourUser\HamRadio\CLA`)

Step 5: Set the Environment Variables You must set two system environment variables that point to the directories you created in the previous step.

- `CONTEST_INPUT_DIR`: Points to your main input directory (e.g., `C:\Users\YourUser\OneDrive\Desktop\CLA_Inputs`).
- `CONTEST_REPORTS_DIR`: Points to your main output directory (e.g., `C:\Users\YourUser\HamRadio\CLA`). **For Windows:**

1. Open the Start Menu and search for "Edit the system environment variables."
2. In the System Properties window, click the "Environment Variables..." button.
3. In the "User variables" section, click "New..." and create both variables.
4. Click OK to close all windows. You must **restart** your terminal or command prompt for the changes to take effect.

Step 6: Obtain and Place Data Files The analyzer relies on several external data files. Download the following files and place them inside the `data/` subdirectory within your **Input Directory** (`CONTEST_INPUT_DIR`).

- `cty.dat`: Required for all contests.
- `CQ160mults.dat`: Required for the CQ 160-Meter contest.
- `arrl_10_mults.dat`: Required for the ARRL 10 Meter contest.
- `ARRLDXmults.dat`: Required for the ARRL DX contest.
- `NAQPmults.dat`: Required for NAQP contests.
- `SweepstakesSections.dat`: Required for ARRL Sweepstakes and ARRL Field Day.
- `band_allocations.dat`: Required for all contests to perform frequency validation.

- `iaru_officials.dat`: Required for the IARU HF World Championship contest.

---

3. Running the Analyzer

To verify the installation, run the program from the project's source code directory. Ensure your `cla` conda environment is active.

CODE_BLOCK

Make sure your conda environment is active

conda activate cla

Run the script from the main project directory, providing a relative path

to a log file inside your CONTEST_INPUT_DIR

(cla) C:..._Analyzer>python main_cli.py --report score_report 2025/NAQP-CW/aug/k3aj.log **CODE_BLOCK**

If the installation is successful, you will see an output message indicating that the report was saved, and you will find a new `.txt` file in a `reports` subdirectory inside your `CONTEST_REPORTS_DIR`.