
eldonationtracker

Release 6.0.0

Eric Mesa

Feb 28, 2021

CONTENTS:

1	Installation	3
1.1	Via PyPi	3
1.1.1	Virtual Environment (recommended)	3
1.1.2	System Packages	3
1.1.3	User Install	4
1.2	Via Github	4
1.2.1	GUI Windows Executable Users	4
1.2.2	Commandline Users and/or Developers	4
2	Usage	5
2.1	For All Users	5
2.2	GUI Windows Executable users (Microsoft Windows Users)	5
2.2.1	Launching	5
2.2.2	Using	6
2.3	Commandline users (PyPi)	10
2.3.1	GUI	10
2.3.2	Commandline Only (No GUI)	10
2.4	Commandline users (git)	10
2.5	Docker or Podman users	11
3	participant.conf	13
3.1	Example	13
3.2	Locations	14
3.3	Config Values	14
4	donation	15
5	donor	17
6	team	19
7	team_participant	21
8	participant	23
9	call_about	27
10	call_settings	29
11	call_tracker	31
12	gui	33

13	extralife_io	35
14	update_available	41
15	Indices and tables	43
	Python Module Index	45
	Index	47

ELDonation Tracker is both a Python interface to the [Extra Life Charity](#) API and a reference implementation, including a GUI, that can be used to provide on-screen donation information and updates when streaming a gaming video or recording a gaming video in OBS or XSplit. For a video explaining how to use the GUI reference implementation, visit: <http://djotaku.github.io/ELDonationTracker/> . You may also use the API to build your own applications that access the API. The modules are well documented, see the Module Index below.

Note: Starting 20200227 this project will strictly follow Semantic Versioning as laid out at <https://semver.org/>

That means:

Given a version number MAJOR.MINOR.PATCH, increment the:

1. MAJOR version when you make incompatible API changes,
 2. MINOR version when you add functionality in a backwards compatible manner, and
 3. PATCH version when you make backwards compatible bug fixes.
-

INSTALLATION

1.1 Via PyPi

The first thing to decide is whether you want to install ELDonationTracker to your system packages, user packages, or a virtual environment.

1.1.1 Virtual Environment (recommended)

Create the folder you want to work in and cd into it.

```
python3 -m venv .
source ./bin/activate
# when you are done using the program you can type deactivate
# first, make sure you have the latest pip, I've had trouble installing with old
↪versions
pip install --upgrade pip
pip install eldonationtracker
# on Windows you may need to type python -m pip install eldonationtracker
# Grab participant.conf from git repo or create based on documentation
# Place participant.conf in persistent location, see the page in documentation
```

1.1.2 System Packages

Note: This is NOT recommended and can cause issues with your system.

```
sudo pip install --upgrade pip
sudo pip install eldonationtracker
# Grab participant.conf from git repo or create based on documentation
# Place participant.conf in persistent location, see the page in documentation
```

1.1.3 User Install

Note: A user install is sometimes a bit buggier than either using a virtual environment or system packages

```
sudo pip install --upgrade pip
sudo pip install eldonationtracker
# Grab participant.conf from git repo or create based on documentation
# Place participant.conf in persistent location, see the page in documentation
```

1.2 Via Github

Here you have a few options depending on what you want to do.

1.2.1 GUI Windows Executable Users

Go to the latest [release](#) and download one of the files that ends in “For Windows”. You can choose a single executable binary or a zip file. The single executable is simply launched like any Windows application - double-click it. It has a slower startup time than the zip file, but once it’s running, there is no performance penalty. For the zip file: unzip it to the location you want to use and then proceed to [Usage](#).

New in version 4.4.0: Single-binary executable build added.

Changed in version 4.0.1: Pyinstaller images will no longer be made for Linux as there are issues with the version of libC it links to as well as other side effects from the VM used by the Github CI system. Linux users can still use the GUI via PyPi, Source Code download, or git clone.

1.2.2 Commandline Users and/or Developers

Two options:

1. Go to the latest [release](#) and click on “Source Code (zip)” or “Source Code (tar.gz)” depending on whether you’re using on Windows or Linux. Then proceed to [Usage](#).
2. Go to the main Github [page](#) and click on “Clone or Download” and click the button to copy the URL to your clipboard. Then run:

```
git clone https://github.com/djotaku/ELDonationTracker.git
```

And any time you want to get up to the latest version you can just go to that folder and type:

```
git pull
```

The master branch is always equivalent to the latest release (except maybe with more up-to-date documentation) so you should always end up with a working version of ELDonationTracker if you do a git pull. (As long as you’re not changing any files. For that reason you may want to move your participant.conf to the persistent location - see [participant.conf](#) for that location) Then proceed to [Usage](#).

2.1 For All Users

Version 5.0 adds the use of the rich module which uses different font colors on the console output (either the little black window that opens on Windows or the terminal used to launch the program on Linux). The following is the key for the colors used:

- green: something positive for the user. Eg: new donor
- red: something that needs the user's attention and signifies an error.
- magenta: Needs user's attention, but isn't necessarily an error
- bold blue: informative, but does not need the user's attention.

New in version 5.0: Added use of rich module to colorize output

2.2 GUI Windows Executable users (Microsoft Windows Users)

This refers to you if you downloaded a file like “Extra Life Donation Tracker for Windows vX.X.zip” or “eldonation-tracker for Windows vX.X.exe”. You may wish to watch a video on how to use the GUI at <http://djotaku.github.io/ELDonationTracker/>. But if you prefer to read, continue below.

2.2.1 Launching

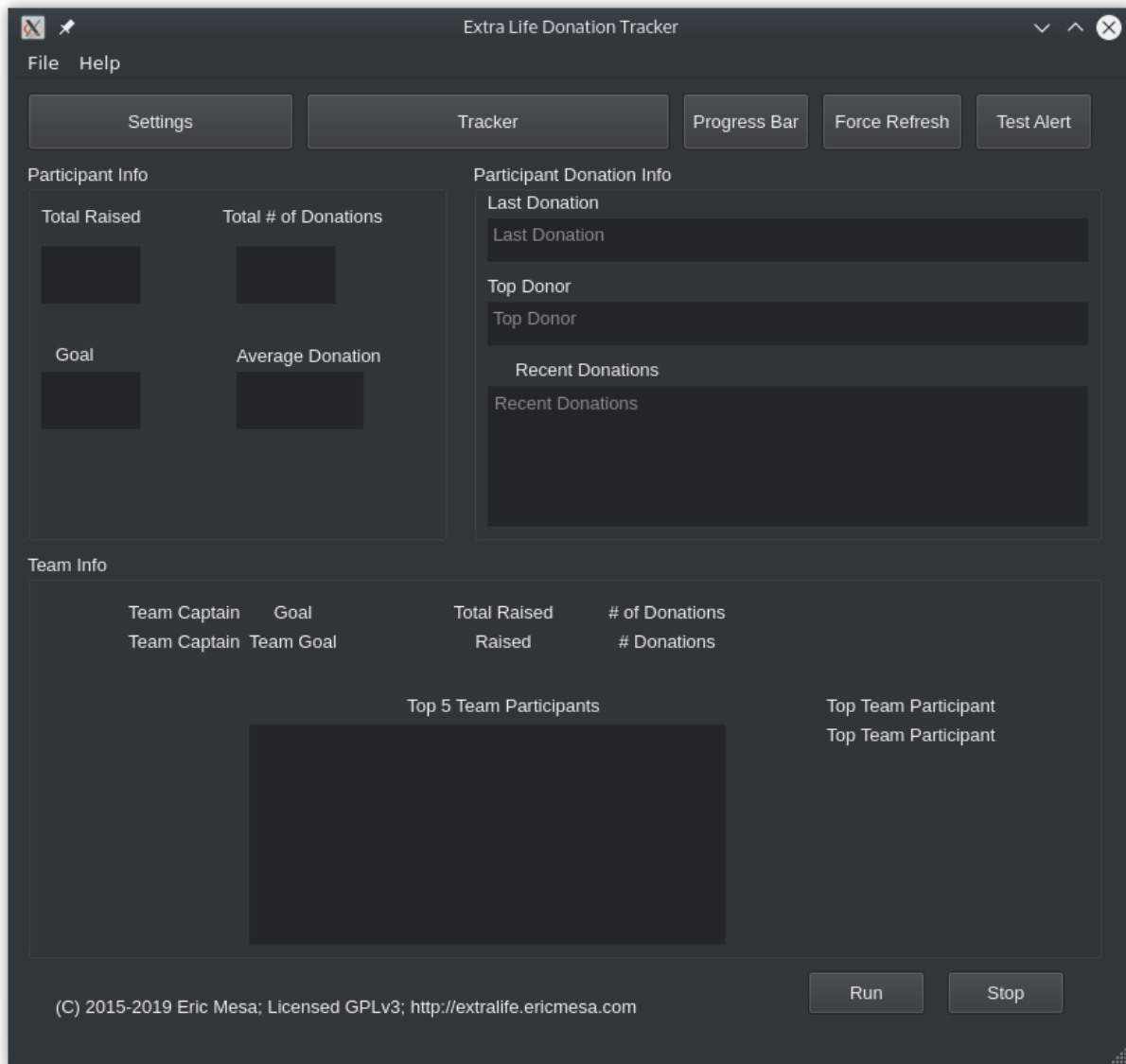
For the zip folder: Go into the folder you unzipped. Find the file called gui.exe and double-click it. If Windows or your anti-virus software throws a warning, click through to allow it to run.

For the single binary: Double-click eldonationtracker.exe to launch it. If Windows or your anti-virus software throws a warning, click through to allow it to run.

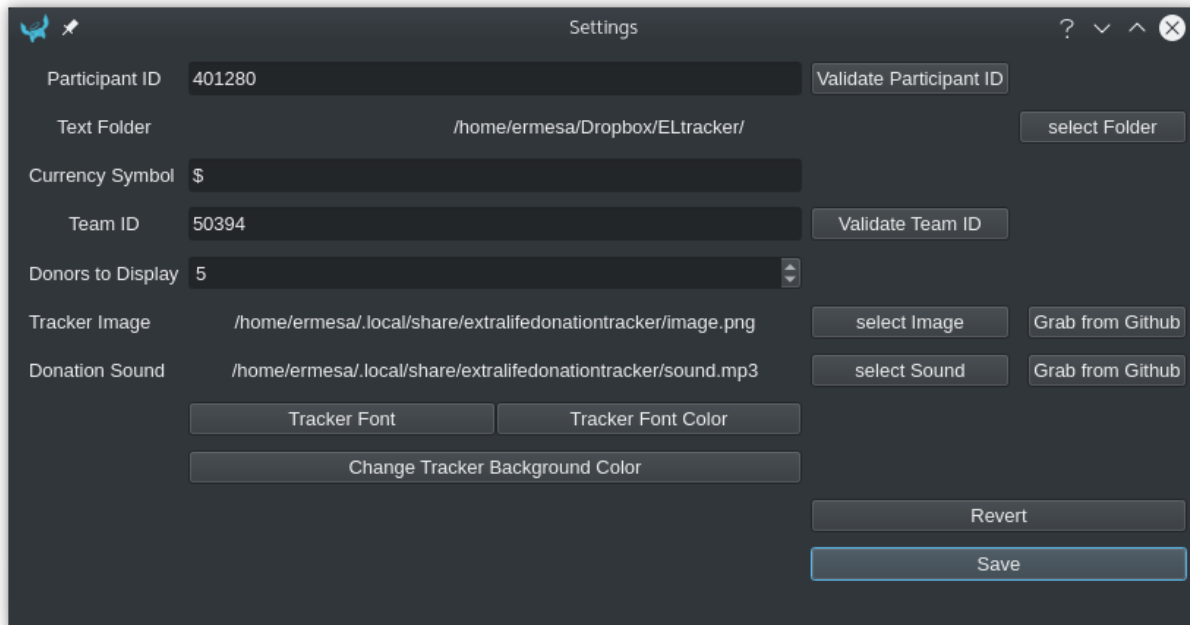
New in version 4.4.0: Single-binary executable.

2.2.2 Using

When you launch the GUI, it will look like this (colors will vary by your OS's color scheme):



If this is not the first time you've used the GUI, it will eventually populate with data (I'll show that at the end). If you've never launched the GUI before and you haven't updated the *participant.conf* during a CLI run, the GUI won't know where to get the data or where to put it once it grabs it from the CLI. So the first thing you want to do is click on the settings window:



The first thing to do here is to enter your participant ID. Where do you find that? Well, it's at the end of your extralife website URL. For example, for 2020, the URL to donate to my campaign is:

<https://www.extra-life.org/index.cfm?fuseaction=donorDrive.participant&participantID=401280>

As you can see, right at the end it says "participantID=401280". If you look back at that settings window image, you'll see that's what I have in there. If you want to check that you've typed or copied the participant ID correctly, you can click on *Validate Participant ID* and it will attempt to connect to that API endpoint for that participant ID. If it is successful, you have the right participant ID (or coincidentally mistyped someone else's). If it's unsuccessful, the most likely reason is that you mistyped it, but you would also get that outcome if the API is unavailable.

New in version 4.3.0: Validate Participant ID

- The next thing to change is the text folder. This is where eldonationtracker will create text files that you will use as inputs in OBS or XSplit. Every time something changes - you get a donation or the team (if you're part of one) gets a donation, those text files will change (as long as eldonationtracker is running) and so they'll change in real time on your screen in OBS or XSplit. (In OBS or XSplit you set a text source linked to one of those files and it will change as the file contents change)
- Now let's edit the Team ID. If you don't have a team, just make this blank. (No spaces!) Otherwise find your Team ID in a similar way as you found your participant ID. Go to your team page and the ID will be at the end of that URL. If you want to check that you've typed or copied the team ID correctly, you can click on *Validate Team ID* and it will attempt to connect to that API endpoint for that team ID. If it is successful, you have the right team ID (or coincidentally mistyped someone else's). If it's unsuccessful, the most likely reason is that you mistyped it, but you would also get that outcome if the API is unavailable.

New in version 4.3.0: Validate Team ID

- Edit the Donors to Display field. Some of the text files produced by eldonationtracker (lastNDonations, topN-donors, etc) use this number to determine how many donors or donations to write to the file. I usually put it at about five, but I also don't get a lot of donations most years.

The final settings all deal with the tracker window. If you scroll just a little lower in this tutorial you'll see a rectangle with a green background and an image and some text appearing and disappearing. That window is what will appear on your screen when someone donates if you've properly set up OBS or XSplit to capture those windows.

- You should select an image with a transparent background (most likely a gif or png) that will appear when someone donates. I'll show what it'll look like in a minute. If you would like to use the default image, click on *Grab from Github* next to the *select Image* button and it will grab it from GitHub and place it in the XDG location for your system. Make sure to hit *Persist Settings* afterward to save it to your settings.

New in version 4.3.0: The ability to grab the default image from GitHub.

- Also select an MP3 file you would like to play when you get a donation. Keep it shorter than 15 seconds. If you would like to use the default mp3 (my daughter saying "you got a donation!"), click on *Grab from Github* next to the *select Sound* button and it will grab it from GitHub and place it in the XDG location for your system. Make sure to hit *Persist Settings* afterward to save it to your settings.

New in version 4.3.0: The ability to grab the default sound from GitHub.

- With the last 3 buttons you can change the Font type and size, the font color, and the background color. I recommend a size around 48-50 (you may need to type it in yourself if it's not a selectable number). For the background color, it's probably best to stick with the chromakey green I've selected because that makes it incredibly easy for OBS or XSplit to make the background disappear so that on your screen you just see the image and text (not the green background). But if the image you want to use has a lot of green in it, you may need to choose bluescreen blue or some other color that will also work well with OBS or XSplit's chromakey filters.

New in version 4.2.0: Ability to change the font type, size, and color as well as the tracker background color.

Warning: Because of the way QT color chooser dialogue windows work, if you pick a color and hit cancel, it will still change the color in the Tracker window. (whereas you have to click "ok" in the Font chooser window to change the font) If you go back in and pick one of the colors from the palette on the left, you can get it working again. Or you can slide the right-most slider from black to white. Finally, if you can't remember what color you had quitting out of everything without saving should bring back the last color you saved (or the default).

- Finally, it's time to save your settings. Hit *Save*. Your settings will be saved to a special location on your computer so that even as you upgrade (either grab new zip files from Github or update via PyPi or git pull) you won't have to keep inputting your settings. If you have not hit *Save* yet, Revert will reload whatever configuration information was in the file when you hit the *Settings* button.

Changed in version 5.2.0: Changed to just have a Save setting instead of Save and Persistent Settings

OK, now it's time to test that things are working with your settings. Close the settings window and click on *Tracker*. Then hit test alert. If everything was correctly set up in the settings, you should see something like:

And hear the sound you picked. What the text says will depend on whether you've ever run this program before either in GUI or on the commandline. If you've never run it, you'll get a test message. If you have run it and the settings are correctly configured, it should show whatever is in your file called `LastDonationNameAmnt.txt`.

OK, now it's time to hit *Run* and hopefully if all the directions have been followed and I haven't introduced any bugs, it should start grabbing data from the API. You should look at the commandline window for information. Whether you launched the GUI from `gui.exe`, used PyPi, or `python gui`, you should have a commandline window showing messages related to what's going on. It should look something roughly like this:

```
Looking for persistent settings at (this path will depend on your system)
Persistent settings found.
Participant.conf version check!
Version is correct
run button
Starting the participant run. But first, reloading config file.
Looking for persistent settings at (this path will depend on your system)
```

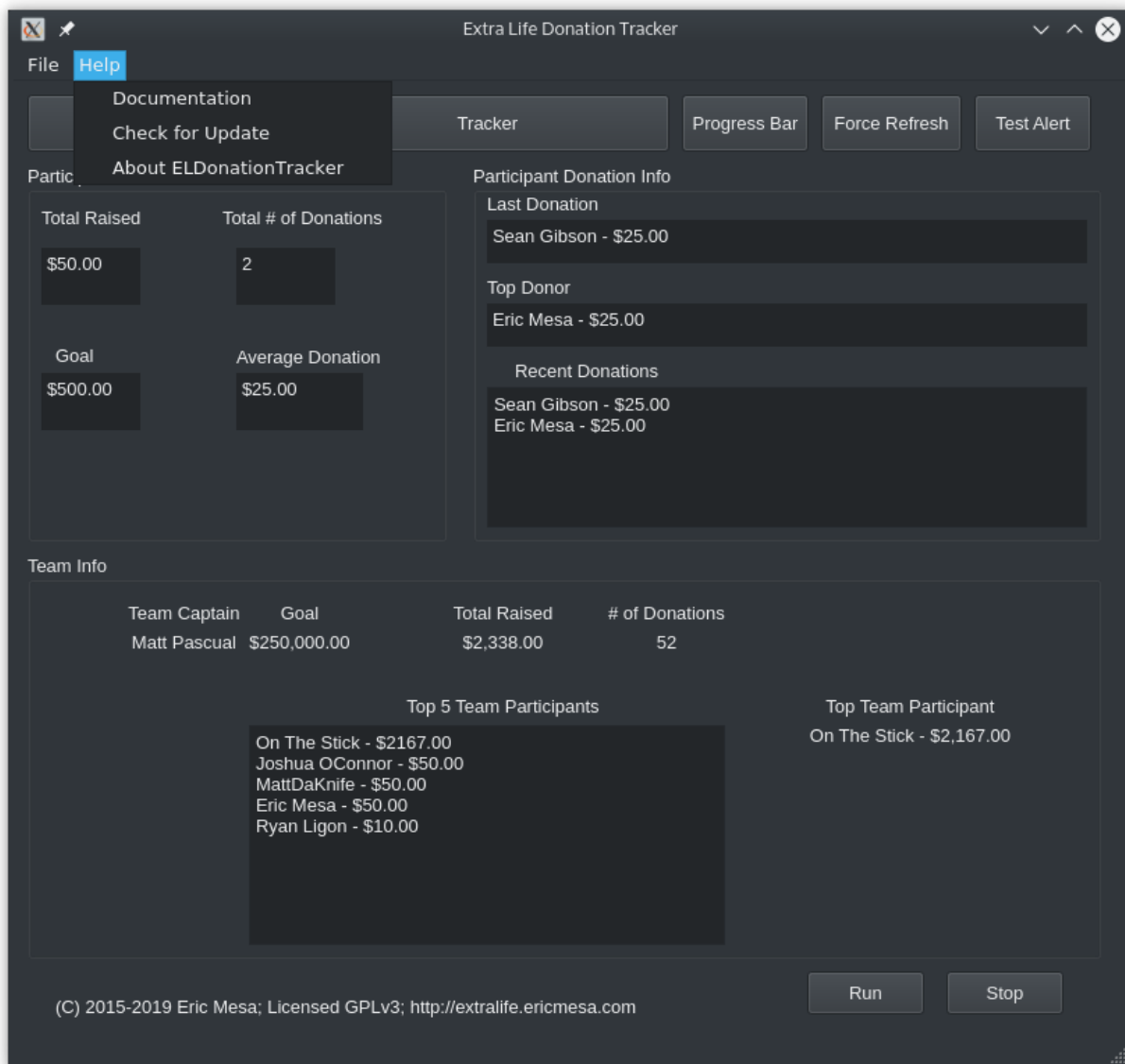
(continues on next page)

(continued from previous page)

```
Persistent settings found.
19:19:10
```

When you're done, be sure to hit stop. When you exit out, it will take a few seconds until it's done before the GUI will disappear. If you Go *File* → *Quit*, that will also trigger it to stop. Again, it'll take a few seconds before it's all cleaned up and ready to disappear from your screen.

Finally, let's quickly go over the help menu items at the top of the GUI.



- *Documentation* will take you to the latest version of this very documentation you're reading now
- *Check for Update* will check if you have the latest version. It will then pop up a window to let you know.
- *About ELDonationTracker* will bring up a window with some URLs and copyright data. Eventually if we start getting more contributors, those would be listed there, too.

2.3 Commandline users (PyPi)

Go to the folder you created in *Installation*. If you don't have the virtual environment activated, start with that:

```
source ./bin/activate
# to check for upgrades
pip install --upgrade eldonationtracker
```

2.3.1 GUI

Make sure you have the *participant.conf* in the persistent location. You can grab the one in the GitHub repo or create your own by looking at the example there. Once the GUI has actually started, you can easily modify the config file via the GUI. To start the GUI:

```
python -m eldonationtracker.gui
```

That should work just fine. Keep an eye on the commandline for any errors or messages from eldonationtracker. The benefit you get from using the GUI is that once the GUI comes up you can click “tracker” to get a window that will display an image and text when a donation is registered. For text instructions on how to use the GUI, go to *Using* or watch the video at <http://djotaku.github.io/ELDonationTracker/>

eg:

You can also edit the *participant.conf* settings in a GUI rather than on the commandline and those settings will persist to commandline-only usage.

2.3.2 Commandline Only (No GUI)

Make sure you have the *participant.conf* in the persistent location. You can grab the one in the Github repo or create your own by looking at the example there. To start the commandline only version:

```
python -m eldonationtracker.cli
```

Of course, you can import the modules into your own scripts and modify how you use the code I've written. In that case, you may be interested in the module index to get a good look at the API available to your program.

Changed in version 6.0.0: Command changed from `python -m eldonationtracker.participant` to `python -m eldonationtracker.cli`

Changed in version 5.0.0: Command changed from `python -m eldonationtracker.extralifedonations` to `python -m eldonationtracker.participant`

2.4 Commandline users (git)

If you downloaded a zip or tar file, unzip it first, then `cd` into that directory. If you did a git clone, `cd` in to that directory. Afterwards, follow along below to create a virtual environment (so as not to mess with your Python installation), grab the required packages, and run the program. (For information on what you should put into *participant.conf*, see *participant.conf*.)

```
python3 -m venv .
source ./bin/activate
# when you are done using the program you can type deactivate
pip install --upgrade pip
pip install -r requirements.txt
# on Windows you may need to type python -m pip install -r requirements.txt
# edit participant.conf
cd eldonationtracker
# for the GUI:
python gui.py
# for the commandline only
python cli.py
```

The benefit you get from using the GUI is that once the GUI comes up you can click “tracker” to get a window that will display an image and text when a donation is registered. For text instructions on how to use the GUI, go to [Using](#) or watch the video at <http://djotaku.github.io/ELDonationTracker/>

eg:

You can also edit the settings in a GUI rather than on the commandline. Once the settings are configured, hit the run button. You should get the same output on the commandline as you would if you weren’t running the GUI. Check there for any errors or messages from the program.

Changed in version 6.0.0: Command changed from from python participant.py to python cli.py

2.5 Docker or Podman users

If you would like to run the donation tracker in Docker or Podman, you can now do so. This would give you access to the commandline output without having to make any changes to the underlying system.

The container can be found at the Docker hub at: <https://hub.docker.com/repository/docker/djotaku/eldonationtracker>

As you already know, you will need a *participant.conf* file with your configuration. For the Docker container, the important thing is that your `text_folder` field needs to have “/root/output/” as the output. (Obviously fill in the other fields - like your participant ID - as necessary)

Let’s say that you have a folder called `extralifedonationtracker` where you have and a folder called `testoutput` where you want the OBS/XSplit output to go. Then you would run:

```
docker run -it -v ./extralifedonationtracker:/root/.config/extralifedonationtracker -v ./testoutput:/root/output djotaku/eldonationtracker:5.3
```

if using docker or

```
podman run -it -v ./extralifedonationtracker:/root/.config/extralifedonationtracker:Z -v ./testoutput:/root/output:Z djotaku/eldonationtracker:5.3
```

if using Podman

New in version 5.3.0: Docker / Podman container

PARTICIPANT.CONF

This is the configuration file for the program.

3.1 Example

Here's an example from my config file in 2020:

```
{
"Version" : "2.0",
"extralife_id" : "401280",
"text_folder" : "/home/ermesa/Dropbox/ELtracker/",
"currency_symbol" : "$",
"team_id" : "50394",
"tracker_image": "Engineer.png",
"donation_sound": "/home/ermesa/Programming Projects/python/extralife/Donation.mp3",
"donors_to_display": "5"
"font_family": "Liberation Sans",
"font_size": 52,
"font_italic": true,
"font_bold": 75,
"font_color": [255, 255, 255, 255],
"tracker_background_color": [0, 255, 0, 255]
}
```

If you are using the GUI, you can edit the file with the GUI and it should always turn out OK. If you are editing it in a text editor, the important thing to remember is the quotation marks.

3.2 Locations

The program will look in 3 locations.

First, it will look in the XDG official locations. If you use “save persistent settings” in the GUI, it will save to this location. On Linux this will be in `$HOME/.config/extralifedonationtracker`. On Windows it will be `C:\Users\username\AppData\Roaming\config\extralifedonationtracker` where you replace username with your Windows username.

Second, it will look up two directory levels. If you cloned from Github or grabbed a Source Code.zip or .tar.gz, this would be in the first directory you cd into - the one that has requirements.txt, README.md, etc.

Third, it will look in the directory where it is being run. If you are using the GUI single executable, this means the same folder as gui.exe (on Windows) or gui (on Linux).

As long as it's in one of those directories, it will be fine. Otherwise it will attempt to grab a copy from Github.

3.3 Config Values

- Version: this should be left alone. It's to know if a persistent config needs updating
- extralife_id: this should be your extralife id, the end of your campaign URL
- text_folder: this is where the text files generated by the program will be placed
- currency_symbol: the prefix for your money
- team_id: if you're on a team, grab from team page URL. If not, put null without quotation marks
- tracker_image: if you're using the GUI, what you want to appear when you get a donation
- donation_sound: if you're using GUI, what you want to hear when there's a donation
- donors_to_display: controls how many donors/donations appear in files that have more than one

The values corresponding to the tracker (everything after donors_to_display) is best configured in the GUI to ensure you're getting values that make sense on your system.

DONATION

A class to hold the Donation attributes and methods.

```
class eldonationtracker.api.donation.Donation(json)
```

Bases: object

Donation Attributes.

Class exists to provide attributes for a donation based on what comes in from the JSON so that it doesn't have to be traversed each time a donor action needs to be taken.

property amount

the amount of the donation. If they blocked it from showing it is set to 0.

property avatar_url

the URL of the avatar associated with this donation.

property donation_date

the date of the donation.

property donation_id

The donor drive ID of this donation.

property donor_id

the donor drive ID of the donor who made this donation

static json_to_attributes(*json*)

Convert API JSON values to Donation attributes.

Parameters *json* (*json*) – JSON attributes from the API

property message

the message associated with the donation.

property name

the name of the donor for this donation. If the donor wished to stay anonymous, the variable is set to/ 'Anonymous'

DONOR

A donor.

```
class eldonationtracker.api.donor.Donor(json)
```

Bases: object

Donor Attributes.

Class exists to provide attributes for a donor based on what comes in from the JSON so that it doesn't have to be traversed each time a donor action needs to be taken.

property amount

The sum of all donations the donor has made this campaign

property donor_id

The ID assigned by the API (currently not used).

property image_url

The URL for the donor's avatar (currently not used)

static json_to_attributes (*json_values*)

Convert API JSON values to Donor attributes.

Parameters *json_values* (*json*) – JSON attributes from the API

property name

Donor's name if provided, else Anonymous.

property number_of_donations

The number of donations the donor has made this campaign

TEAM

Contains classes pertaining to teams.

```
class eldonationtracker.api.team.Team(team_id: str, output_folder: str, currency_symbol: str, donors_to_display: str)
```

Bases: object

Hold Team API Data.

property currency_symbol

The currency symbol used in the output.

donation_run() → None

Get and calculate donation information.

property donors_to_display

The number of donors to write out to the output file.

property num_donations

The number of donations to the team.

property output_folder

The folder for the output text files.

participant_run() → None

Get and calculate team participant info.

team_api_info() → None

Get team info from API.

property team_avatar_image

The team's avatar image.

property team_captain

The name of the team captain.

property team_donation_url

URL to the JSON api for donations to the team.

property team_goal

The fundraising goal of the team.

property team_id

The team's ID in the API.

property team_participant_url

URL to the JSON api for participants in the team.

team_run() → None

A public method to update and output team and team participant info.

property team_url

URL to the team JSON API.

property team_url_base

The donor drive endpoint for the teams.

property total_raised

The total amount raised by the team.

write_text_files (*dictionary: dict*) → None

Write info to text files.

Parameters dictionary – The dictionary containing the values to write out to text files.

TEAM_PARTICIPANT

class eldonationtracker.api.team_participant.**TeamParticipant** (*json*)

Bases: *eldonationtracker.api.donor.Donor*

Participant Attributes.

Inherits from the donor class, but over-rides the `json_to_attributes` function.

API variables:

Parameters

- **self.name** – participant’s name
- **self.amount** – the sum of all donations by this participant
- **self.number_of_donations** – number of all donations by this participant
- **self.image_url** – the url of the participant’s avatar image (not used)

PARTICIPANT

Grabs Participant JSON data and outputs to files.

```
class eldonationtracker.api.participant.Participant (config)
```

Bases: object

Owns all the attributes under the participant API.

Also owns the results of any calculated data.

Donor Drive API api info at <https://github.com/DonorDrive/PublicAPI>

property avatar_image_url

The user's avatar image on the website.

property average_donation

The average amount of donations received by this participant

property created_date_utc

The date the participant created their campaign.

property currency_symbol

The currency symbol eg '\$' added to the output text files.

property display_name

Return the participant's display name.

property donation_link_url

The link for making donations to the participant.

property donation_url

API endpoint for the Participant's donations.

property donors_to_display

For output text files that display multiple donors (or donations), the number of them that should be written to the text file.

property event_name

The event name as obtained from the API.

property extra_life_page_url

The URL to the participant's ExtraLife page.

property extralife_id

The participant's extra life ID

property goal

The goal for the amount of the money the participant wishes to raise.

property is_team_captain

True if the participant is the team captain.

property my_team

An instantiation of a team class for the participant's team.

property new_donation

True if the participant receives a new donation.

property number_of_donations

The number of donations received by the participant.

output_donation_data () → None

Write out text files for donation data.

If there have been donations, format the data (eg horizontally, vertically, etc) and output to text files. If there have not yet been donations, write default data to the files.

output_donor_data () → None

Write out text files for donor data.

If there have been donations, format the data (eg horizontally, vertically, etc) and output to text files. If there have not yet been donations, write default data to the files.

output_participant_data () → None

Format participant data and write to text files for use by OBS or XSplit.

A public method to do the above. Also called from the main loop.

property participant_donor_url

API Endpoint for the Participant's donors.

property participant_url

API endpoint for the participant

run () → None

Run to get participant, donation, donor, and team data and output to text files.

set_config_values () → None

Set participant values, create URLs, and create Team.

property stream_is_live

True if the participant is actively streaming.

property stream_url

The URL to the participant's stream on Twitch.

property sum_pledges

Number of pledges for the participant.

property team_id

The Team ID if the participant is part of a team.

property team_name

If the participant is part of a team, this is the team name.

property text_folder

Where the output text files will be written on disk

property total_raised

Total amount raised by the participant.

update_donation_data () → None

Update donation data.

update_donor_data () → None

Update donor data.

update_participant_attributes () → None

Update participant attributes.

A public method that will update the Participant object with data from self.participant_url.

Also called from the main loop.

write_text_files (*dictionary: dict*) → None

Write OBS/XSplit display info to text files.

It uses the helper function extralife_IO.write_text_files to handle the task.

Parameters dictionary (*dict*) – Dictionary containing values to write to text files . The key will become the filename. The value will be written to the file.

CALL_ABOUT

Contains programming logic for the about window in the GUI.

```
class eldonationtracker.ui.call_about.AboutProgram(*args: Any, **kwargs: Any)
    Bases: PyQt5.QtWidgets.

    ok_clicked()

eldonationtracker.ui.call_about.main()
```


CALL_SETTINGS

Contains the programming logic for the settings window in the GUI.

```
class eldonationtracker.ui.call_settings.MyForm(participant_conf, tracker)
```

```
    Bases: PyQt5.QtWidgets.QDialog
```

```
    Class for the settings Window.
```

```
    persistent_save()
```

```
        Use xdg_config, saves a persistent config to the XDG spot.
```

```
    reload_config()
```

```
        Reload the values from the config file.
```

```
        Called by ui.py before loading the settings window.
```

```
    revert()
```

```
        Revert the values in the settings window.
```

```
        If the user made mistakes while editing the config this will take them back to the values since they opened the window.
```

```
eldonationtracker.ui.call_settings.main(participant_conf)
```

```
    Launch the window.
```


CALL_TRACKER

A window that displays the last donation. Useful during streaming.

```
class eldonationtracker.ui.call_tracker.MyForm (participant_conf, participant)
```

```
    Bases: PyQt5.QtWidgets.QDialog
```

```
    The class to hold the tracker window.
```

```
    load_and_unload_test ()
```

```
        Trigger the tracker functionality.
```

```
        This causes the image and sound to load so that the user can test to see how it's going to look on their OBS or XSplit screen as well as to make sure they can hear the sound. Called by ui.py.
```

```
    set_background_color (color)
```

```
    set_font (font)
```

```
    set_font_color (font_color)
```

```
eldonationtracker.ui.call_tracker.main (participant_conf)
```

```
    Launch the window.
```


GUI

The main GUI window.

```
class eldonationtracker.ui.call_main_gui.ELDonationGUI
    Bases:      PyQt5.QtWidgets.QMainWindow,      eldonationtracker.ui.main_gui.
               Ui_MainWindow
    The main ui Window.
    call_settings_button()
    call_tracker_button()
    check_for_update()
    static dead_button()
    get_some_text()
    load_documentation()
    quit()
        Quit the application.
    static read_files(folders, files)
    run_button()
    show_about()
    stop_button()
    test_alert()
    version_check()
eldonationtracker.ui.call_main_gui.main()
```


EXTRALIFE_IO

Holds all the file and internet input and output.

class eldonationtracker.utils.extralife_io.ParticipantConf

Bases: object

Holds Participant Configuration info.

Parameters

- **cls.participant_conf_version** – version of participant.conf
- **cls.version_mismatch** – Initialized to False.If true, the user has a different version of the participant.conf.
- **cls.fields** – A dictionary initialed to None for all fields.
- **self.xdg** – By using the PedanticPackage, the directory will be created if it doesn't already exist.
- **self.participantconf** – holds a dictionary of the user's config file.

fields: dict = {'currency_symbol': None, 'donation_sound': None, 'donors_to_display

get_cli_values() → Tuple[Any, Any, Any, Any, Any]

Return data required for a CLI-only run.

Returns A tuple of strings with config values needed if only running on the commandline.

get_font_info()

Return values needed to change the font for the tracker.

Returns A tuple of strings and a list for the color.

get_github_config()

get_gui_values() → Tuple[Any, Any, Any, Any, Any, Any, Any, Any, Any, Any, Any, Any, Any]

Return values needed for the GUI.

Returns A tuple of strings with config values needed if only running the GUI. The two background colors are lists.

get_if_in_team() → bool

Return True if participant is in a team.

Returns True if the participant is in a team.

get_text_folder_only() → str

Return text folder data.

Returns A string with the text folder location.

get_tracker_assets (*asset: str*)

get_tracker_background_color ()

Return value needed to change the tracker background color

Returns A list representing the RGB value for the background

get_tracker_image () → str

Return the tracker image location on disk.

Returns Location of tracker image on disk.

get_tracker_sound () → str

Return the donation sound image location on disk.

Returns location of the donation sound on disk.

get_version ()

Return version.

get_version_mismatch () → bool

Return bool of whether there is a version mismatch.

Returns True if the version the user is running is not the same as what this program has as its version.

load_json () → dict

Load in the config file.

Checks in a variety of locations for the participant.conf file. First, it checks in the persistent settings location (XDG_CONFIG_HOME) . Then it checks the current directory and one level up. Otherwise it raises an exception and the program stops.

Returns A dictionary representing the JSON config file.

Raises FileNotFoundError

participant_conf_version: str = '2.0'

reload_json ()

Reload JSON and update the fields.

update_fields ()

Update fields variable with data from JSON.

version_mismatch: bool = False

write_config (*config: dict, default: bool*)

Write config to file.

Only called from GUI. Commandline user is expected to edit participant.conf manually. Afterward it triggers self.reloadJSON to have the program run with the updated config.

Parameters

- **config** – A dictionary holding the config values.
- **default** – If True, will save the file in the current directory.

`eldonationtracker.utils.extralife_io.format_information_for_output` (*donation_list:*
list, cur-
rency_symbol:
str,
donors_to_display:
str, team:
bool,
dona-
tion=True)
 → dict

Format the donation attributes for the output files.

Parameters

- **donation** –
- **donation_list** (*list*) – A list of donors or donations to format for output.
- **currency_symbol** – The currency symbol for the output.
- **donors_to_display** – How many donors or donations to display
- **team** – If true, this is creating output for a team. Otherwise, for the participant.

Returns A dictionary with the output text formatted correctly.

`eldonationtracker.utils.extralife_io.get_donations` (*donations_or_donors:* *list,*
api_url: *str, is_donation=True,*
largest_first=False) → list

Get the donations from the JSON and create the donation objects.

If the API can't be reached, the same list is returned. Only new donations are added to the list at the end.

Parameters

- **is_donation** – True if we are getting data for donations. False if we are getting data for donors.
- **largest_first** – True if we want to sort by largest. False if sort by latest donors or donations.
- **donations_or_donors** – A list consisting of donation.Donation or donor.Donor objects.
- **api_url** – The URL to go to for donations.

Returns A list of donation.Donation objects.

`eldonationtracker.utils.extralife_io.get_json` (*url:* *str, order_by_donations:* *bool =*
False, order_by_amount: *bool = False)*
 → dict

Grab JSON from server.

Connects to server and grabs JSON data from the specified URL. The API server should return JSON with the donation data.

Parameters

- **url** – API URL for the specific json API point.
- **order_by_donations** – If true, the url param has data appended that will cause the API to return the data in descending order of the sum of donations.
- **order_by_amount** – If true, the url param has data appended that will cause the API to return the data in descending order of the sum of amounts.

Returns JSON as dictionary with API data.

Raises HTTPError, URLError

`eldonationtracker.utils.extralife_io.multiple_format(donors, message: bool, horizontal: bool, currency_symbol: str, how_many: int) → str`

Format string for output to text file.

This function is for when there is are multiple donors or donations. For example, for the text file holding the Top 5 donors.

Parameters

- **donors** – An iterable of Donors or Donations class objects.
- **message** – If True, the message (if any) from the donor object :param horizontal: If True, format the message horizontally. Else vertically.
- **currency_symbol** – The currency symbol to append to the return string.
- **how_many** – A number for how many donors/donations to append to the string.

Returns A string containing the inputs formatted.

Horizontal example:

John Doe - \$100.00 - Thanks for raising money for the kids. | Jane Doe - \$25.00 - Hurray!

Vertical example:

John Doe - \$200.00 - a message

Jane Doe - \$75.00 - another message

`eldonationtracker.utils.extralife_io.single_format(donor, message: bool, currency_symbol: str) → str`

Format string for output to text file.

This function is for when there is only one donor or donation. For example, for the text file holding the most recent donation.

Parameters

- **donor** – Donor or Donation class object.
- **message** – If True, the message (if any) from the donor object :param currency_symbol: The currency symbol to append to the return string.

Returns A string containing the inputs formatted. For example:

John Doe - \$100.00 - Thanks for raising money for the kids.

`eldonationtracker.utils.extralife_io.validate_url(url: str)`

`eldonationtracker.utils.extralife_io.write_html_files(data: str, filename: str, text_folder: str)`

Write data to an HTML file.

Parameters

- **data** – The data to write to an HTML file.
- **filename** – The filename for the HTML file.
- **text_folder** – The directory to write the HTML files to.

eldonationtracker.utils.extralife_io.**write_text_files** (*dictionary: dict, text_folder: str*)

Write info to text files.

The dictionary key will become the filename and the values will be the content of the files.

Parameters

- **dictionary** – The dictionary with items to output.
- **text_folder** – The directory to write the text files.

UPDATE_AVAILABLE

Return true if there is an update available.

An update is available if the version on PyPi is higher than the version the user has on their machine.

`eldonationtracker.utils.update_available.get_pypi_version(url: str) → str`
Use PyPi JSON API to get latest version.

Returns A string with the version number.

`eldonationtracker.utils.update_available.main() → bool`
Get the latest version on PyPi and compare to current version.

Made the decision to return false if the URL couldn't be reached rather than make the user look for an update that might not be there.

Returns True if there's an update available. False if up to date.

`eldonationtracker.utils.update_available.update_available(pypi_version: str, current_version: str) → bool`

Use semver module to calculate whether there is a newer version on PyPi.

Returns True if the PyPi version is higher than the version being run. Returns false if the version being compared to PyPi is equal or greater than the PyPi version.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

e

- `eldonationtracker.api.donation`, [15](#)
- `eldonationtracker.api.donor`, [17](#)
- `eldonationtracker.api.participant`, [23](#)
- `eldonationtracker.api.team`, [19](#)
- `eldonationtracker.api.team_participant`,
[21](#)
- `eldonationtracker.ui.call_about`, [27](#)
- `eldonationtracker.ui.call_main_gui`, [33](#)
- `eldonationtracker.ui.call_settings`, [29](#)
- `eldonationtracker.ui.call_tracker`, [31](#)
- `eldonationtracker.utils.extralife_io`,
[35](#)
- `eldonationtracker.utils.update_available`,
[41](#)

A

AboutProgram (class in eldonation-tracker.ui.call_about), 27
 amount() (eldonationtracker.api.donation.Donation property), 15
 amount() (eldonationtracker.api.donor.Donor property), 17
 avatar_image_url() (eldonation-tracker.api.participant.Participant property), 23
 avatar_url() (eldonation-tracker.api.donation.Donation property), 15
 average_donation() (eldonation-tracker.api.participant.Participant property), 23

C

call_settings_button() (eldonation-tracker.ui.call_main_gui.ELDonationGUI method), 33
 call_tracker_button() (eldonation-tracker.ui.call_main_gui.ELDonationGUI method), 33
 check_for_update() (eldonation-tracker.ui.call_main_gui.ELDonationGUI method), 33
 created_date_utc() (eldonation-tracker.api.participant.Participant property), 23
 currency_symbol() (eldonation-tracker.api.participant.Participant property), 23
 currency_symbol() (eldonation-tracker.api.team.Team property), 19

D

dead_button() (eldonation-tracker.ui.call_main_gui.ELDonationGUI static method), 33
 display_name() (eldonation-tracker.api.participant.Participant property),

23

Donation (class in eldonationtracker.api.donation), 15
 donation_date() (eldonation-tracker.api.donation.Donation property), 15
 donation_id() (eldonation-tracker.api.donation.Donation property), 15
 donation_link_url() (eldonation-tracker.api.participant.Participant property), 23
 donation_run() (eldonationtracker.api.team.Team method), 19
 donation_url() (eldonation-tracker.api.participant.Participant property), 23
 Donor (class in eldonationtracker.api.donor), 17
 donor_id() (eldonationtracker.api.donation.Donation property), 15
 donor_id() (eldonationtracker.api.donor.Donor property), 17
 donors_to_display() (eldonation-tracker.api.participant.Participant property), 23
 donors_to_display() (eldonation-tracker.api.team.Team property), 19

E

ELDonationGUI (class in eldonation-tracker.ui.call_main_gui), 33
 eldonationtracker.api.donation module, 15
 eldonationtracker.api.donor module, 17
 eldonationtracker.api.participant module, 23
 eldonationtracker.api.team module, 19
 eldonationtracker.api.team_participant module, 21
 eldonationtracker.ui.call_about module, 27

eldonationtracker.ui.call_main_gui
module, 33

eldonationtracker.ui.call_settings
module, 29

eldonationtracker.ui.call_tracker
module, 31

eldonationtracker.utils.extralife_io
module, 35

eldonationtracker.utils.update_available
module, 41

event_name() (eldonation-
tracker.api.participant.Participant
property),
23

extra_life_page_url() (eldonation-
tracker.api.participant.Participant
property),
23

extralife_id() (eldonation-
tracker.api.participant.Participant
property),
23

F

fields (eldonationtracker.utils.extralife_io.ParticipantConf
attribute), 35

format_information_for_output() (in mod-
ule eldonationtracker.utils.extralife_io), 36

G

get_cli_values() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_donations() (in module eldonation-
tracker.utils.extralife_io), 37

get_font_info() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_github_config() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_gui_values() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_if_in_team() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_json() (in module eldonation-
tracker.utils.extralife_io), 37

get_pypi_version() (in module eldonation-
tracker.utils.update_available), 41

get_some_text() (eldonation-
tracker.ui.call_main_gui.ELDonationGUI
method), 33

get_text_folder_only() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_tracker_assets() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 35

get_tracker_background_color() (eldona-
tiontracker.utils.extralife_io.ParticipantConf
method), 36

get_tracker_image() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 36

get_tracker_sound() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 36

get_version() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 36

get_version_mismatch() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 36

goal() (eldonationtracker.api.participant.Participant
property), 23

image_url() (eldonationtracker.api.donor.Donor
property), 17

is_team_captain() (eldonation-
tracker.api.participant.Participant
property),
23

J

json_to_attributes() (eldonation-
tracker.api.donation.Donation static method),
15

json_to_attributes() (eldonation-
tracker.api.donor.Donor static method),
17

L

load_and_unload_test() (eldonation-
tracker.ui.call_tracker.MyForm method),
31

load_documentation() (eldonation-
tracker.ui.call_main_gui.ELDonationGUI
method), 33

load_json() (eldonation-
tracker.utils.extralife_io.ParticipantConf
method), 36

M

main() (in module eldonationtracker.ui.call_about), 27

main() (in module eldonationtracker.ui.call_main_gui),
33

main() (in module eldonationtracker.ui.call_settings),
29

main() (in module eldonationtracker.ui.call_tracker), 31
 main() (in module eldonation-tracker.utils.update_available), 41
 message() (eldonationtracker.api.donation.Donation property), 15
 module
 eldonationtracker.api.donation, 15
 eldonationtracker.api.donor, 17
 eldonationtracker.api.participant, 23
 eldonationtracker.api.team, 19
 eldonationtracker.api.team_participant, 21
 eldonationtracker.ui.call_about, 27
 eldonationtracker.ui.call_main_gui, 33
 eldonationtracker.ui.call_settings, 29
 eldonationtracker.ui.call_tracker, 31
 eldonationtracker.utils.extralife_io, 35
 eldonationtracker.utils.update_available, 41
 multiple_format() (in module eldonation-tracker.utils.extralife_io), 38
 my_team() (eldonation-tracker.api.participant.Participant property), 24
 MyForm (class in eldonationtracker.ui.call_settings), 29
 MyForm (class in eldonationtracker.ui.call_tracker), 31

N

name() (eldonationtracker.api.donation.Donation property), 15
 name() (eldonationtracker.api.donor.Donor property), 17
 new_donation() (eldonation-tracker.api.participant.Participant property), 24
 num_donations() (eldonationtracker.api.team.Team property), 19
 number_of_donations() (eldonation-tracker.api.donor.Donor property), 17
 number_of_donations() (eldonation-tracker.api.participant.Participant property), 24

O

ok_clicked() (eldonation-tracker.ui.call_about.AboutProgram method), 27

output_donation_data() (eldonation-tracker.api.participant.Participant method), 24
 output_donor_data() (eldonation-tracker.api.participant.Participant method), 24
 output_folder() (eldonationtracker.api.team.Team property), 19
 output_participant_data() (eldonation-tracker.api.participant.Participant method), 24

P

Participant (class in eldonation-tracker.api.participant), 23
 participant_conf_version (eldonation-tracker.utils.extralife_io.ParticipantConf attribute), 36
 participant_donor_url() (eldonation-tracker.api.participant.Participant property), 24
 participant_run() (eldonation-tracker.api.team.Team method), 19
 participant_url() (eldonation-tracker.api.participant.Participant property), 24
 ParticipantConf (class in eldonation-tracker.utils.extralife_io), 35
 persistent_save() (eldonation-tracker.ui.call_settings.MyForm method), 29

Q

quit() (eldonationtracker.ui.call_main_gui.ELDonationGUI method), 33

R

read_files() (eldonation-tracker.ui.call_main_gui.ELDonationGUI static method), 33
 reload_config() (eldonation-tracker.ui.call_settings.MyForm method), 29
 reload_json() (eldonation-tracker.utils.extralife_io.ParticipantConf method), 36
 revert() (eldonationtracker.ui.call_settings.MyForm method), 29
 run() (eldonationtracker.api.participant.Participant method), 24
 run_button() (eldonation-tracker.ui.call_main_gui.ELDonationGUI method), 33

S

`set_background_color()` (eldonationtracker.ui.call_tracker.MyForm method), 31
`set_config_values()` (eldonationtracker.api.participant.Participant method), 24
`set_font()` (eldonationtracker.ui.call_tracker.MyForm method), 31
`set_font_color()` (eldonationtracker.ui.call_tracker.MyForm method), 31
`show_about()` (eldonationtracker.ui.call_main_gui.ELDonationGUI method), 33
`single_format()` (in module eldonationtracker.utils.extralife_io), 38
`stop_button()` (eldonationtracker.ui.call_main_gui.ELDonationGUI method), 33
`stream_is_live()` (eldonationtracker.api.participant.Participant property), 24
`stream_url()` (eldonationtracker.api.participant.Participant property), 24
`sum_pledges()` (eldonationtracker.api.participant.Participant property), 24

T

`Team` (class in eldonationtracker.api.team), 19
`team_api_info()` (eldonationtracker.api.team.Team method), 19
`team_avatar_image()` (eldonationtracker.api.team.Team property), 19
`team_captain()` (eldonationtracker.api.team.Team property), 19
`team_donation_url()` (eldonationtracker.api.team.Team property), 19
`team_goal()` (eldonationtracker.api.team.Team property), 19
`team_id()` (eldonationtracker.api.participant.Participant property), 24
`team_id()` (eldonationtracker.api.team.Team property), 19
`team_name()` (eldonationtracker.api.participant.Participant property), 24
`team_participant_url()` (eldonationtracker.api.team.Team property), 19

`team_run()` (eldonationtracker.api.team.Team method), 19
`team_url()` (eldonationtracker.api.team.Team property), 19
`team_url_base()` (eldonationtracker.api.team.Team property), 20
`TeamParticipant` (class in eldonationtracker.api.team_participant), 21
`test_alert()` (eldonationtracker.ui.call_main_gui.ELDonationGUI method), 33
`text_folder()` (eldonationtracker.api.participant.Participant property), 24
`total_raised()` (eldonationtracker.api.participant.Participant property), 24
`total_raised()` (eldonationtracker.api.team.Team property), 20

U

`update_available()` (in module eldonationtracker.utils.update_available), 41
`update_donation_data()` (eldonationtracker.api.participant.Participant method), 24
`update_donor_data()` (eldonationtracker.api.participant.Participant method), 24
`update_fields()` (eldonationtracker.utils.extralife_io.ParticipantConf method), 36
`update_participant_attributes()` (eldonationtracker.api.participant.Participant method), 25

V

`validate_url()` (in module eldonationtracker.utils.extralife_io), 38
`version_check()` (eldonationtracker.ui.call_main_gui.ELDonationGUI method), 33
`version_mismatch` (eldonationtracker.utils.extralife_io.ParticipantConf attribute), 36

W

`write_config()` (eldonationtracker.utils.extralife_io.ParticipantConf method), 36
`write_html_files()` (in module eldonationtracker.utils.extralife_io), 38

`write_text_files()` (*eldonation-tracker.api.participant.Participant* method), [25](#)
`write_text_files()` (*eldonation-tracker.api.team.Team* method), [20](#)
`write_text_files()` (in module *eldonation-tracker.utils.extralife_io*), [38](#)