OBD Collection and Display

Tools to collect vehicle onboard diagnostic data and graphically display the data

Collection

- OBD Collection Tools – GITHUB repositiory https://github.com/mbroihier/obd-collection-tools
  - obd_logger.py – python3 module that opens a session with a ELM 327 OBD device and collects all mode 1 data available in the vehicle
    - Logger connects to the interface
    - Queries the vehicle to obtain a list of readings that are available
    - Cycles through the list and creates a “CSV” file of the readings

Collection (continued)

- obd_log_to_json.py
  - Converts obd_logger.py CSV files into json files
  - Converts obd_logger.py CSV files into JavaScript files that can be imported by the display server

Display

- OBD Display Tools - GITHUB repository https://github.com/mbroihier/obd-display-server
- Node.js, express, Heroku based server that displays files produced by the obd_logger and obd_log_to_json tools
- Server displays JavaScript plot files produced by obd_log_to_json tool
- Server displays graphs of collected data
  - Lines can be removed/restored
  - Lines can be put on an alternate axis

Equipment

- OBD logging is performed with a Raspberry PI 3 using a bluetooth connection to a Foseal OBDII ELM327 interface.
- I’ve programmed the Raspberry PI to run Node Red and to automatically start the Python 3 obd_logger.py utility mentioned above.
- My Node Red scripts are also monitoring the status of the logger and displays a green LED on a Blinkt LED display if the logger is running. It is red if it fails.
- My Node Red scripts are also monitoring a shutdown button that is on my prototype board. When this button is pressed, the scripts shutdown the logger and power off the Raspberry PI allowing me to run “headless”