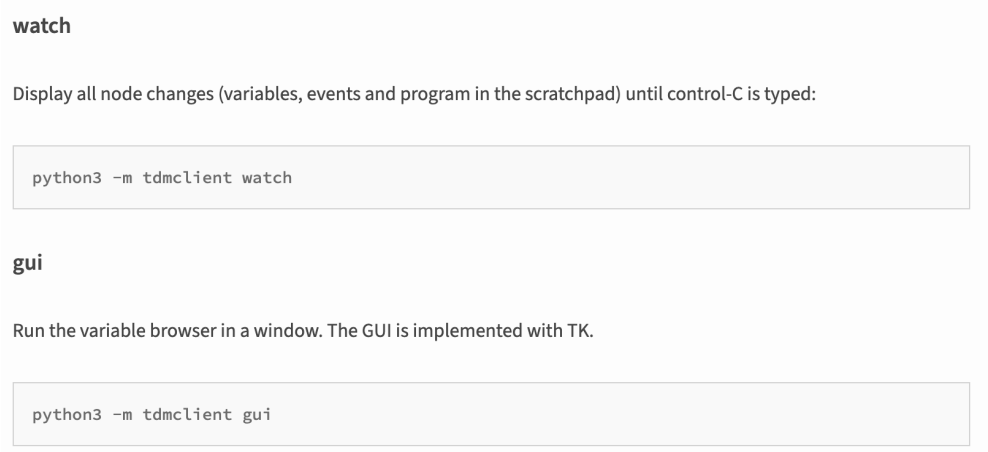
Understanding tdmclient: The ultimate Guide

tdmclient is a powerful tool that allows interaction with Thymio robots in various ways. Here, we explore **3 main methods** to leverage its capabilities:

1. Command Line Interface (CLI)

Environment: Terminal
Use Cases: Primarily for monitoring Thymio's status or uploading code.
Relevance: This method may not be of high interest for our purposes since it's less hands-on for programming education.

Examples of CLI usage:



watch

Display all node changes (variables, events and program in the scratchpad) until control-C is typed:

```
python3 -m tdmclient watch
```

gui

Run the variable browser in a window. The GUI is implemented with TK.

```
python3 -m tdmclient gui
```

2. Python-to-Aseba Transpiler

(https://github.com/epfl-mobots/tdm-python/blob/main/doc/transpiler.md)

Environment: Jupyter Notebook using `%%run_python` or `%%run_aseba` magic commands.
→ The code actually runs on the robot (meaning : it is executed by the microcontroller of the Thymio)

Key Points:

- **Limitations:** The transpiled Python code is limited to **basic functions.(** “Imagine the person that has to write the translator. Obviously he did not translate all libraries, and only coded the basic features. Also aseba is a language that can only handle ints, no floats for example. So we are quite limited here”
- **Autonomy**: After flashing, the robot operates independently without a connected PC.
- **Only** the code in the specific Jupyter cell is translated.
- Expect restrictions similar to coding in Aseba itself.

Functions and Features: Refer to the Python-to-Aseba feature comparison for details on what's available.

3. ClientAsync Module

(https://github.com/epfl-mobots/tdm-python/blob/main/doc/lowlevel.md)

Environment: Python script with the ClientAsync module.

Key Points:

- **Connection**: A node must be locked at the start to secure communication with Thymio.
- **Operation**: The code runs on the computer and interacts from time to time with the Thymio → continuous connection with the robot needed !
- **Asynchronous Communication**: Need for async functions when communicating (called with `await`) because the code is essentially waiting for the Thymio to return/update its values
- **Higher-level programming:** it is possible with access to extensive libraries (e.g., numpy, pandas, opencv). → Greater flexibility and complexity in coding due to PC-based execution.

For an in-depth look at the asynchronous client, check out the ClientAsync documentation.