Developing Software to Support the Carbon Dioxide Model

Thomas Roller

The CO_2 Model

- Beginning of the term
  - Simulates how CO_2 spreads in an enclosed space
  - Restricted to 2 dimensions
  - No compatible way to graph the concentration of CO_2 for a cell over time
  - Simulations may not match real world data
- Goals throughout the term
  - Permit the model to represent 3-dimensional spaces
    - Improved accuracy
  - Provide ways to visualize the information from Cadmium
  - Tweak model parameters to match actual data

Approaches for Creating Scenarios

- Allow the CO_2 model to support 3-dimensional simulations and scenarios
  - C++ code already creates N-dimensional neighbourhoods
  - Neighbourhood dimension is matched to the given scenario
  - No changes to the C++ code was required
- Creating scenarios
  - Convert existing scenarios of the format “X,Y=concentration,type,counter” into their 3D, JSON equivalents
  - Provide a new way to create models

Creating 3D Scenarios - Preliminary Scripts

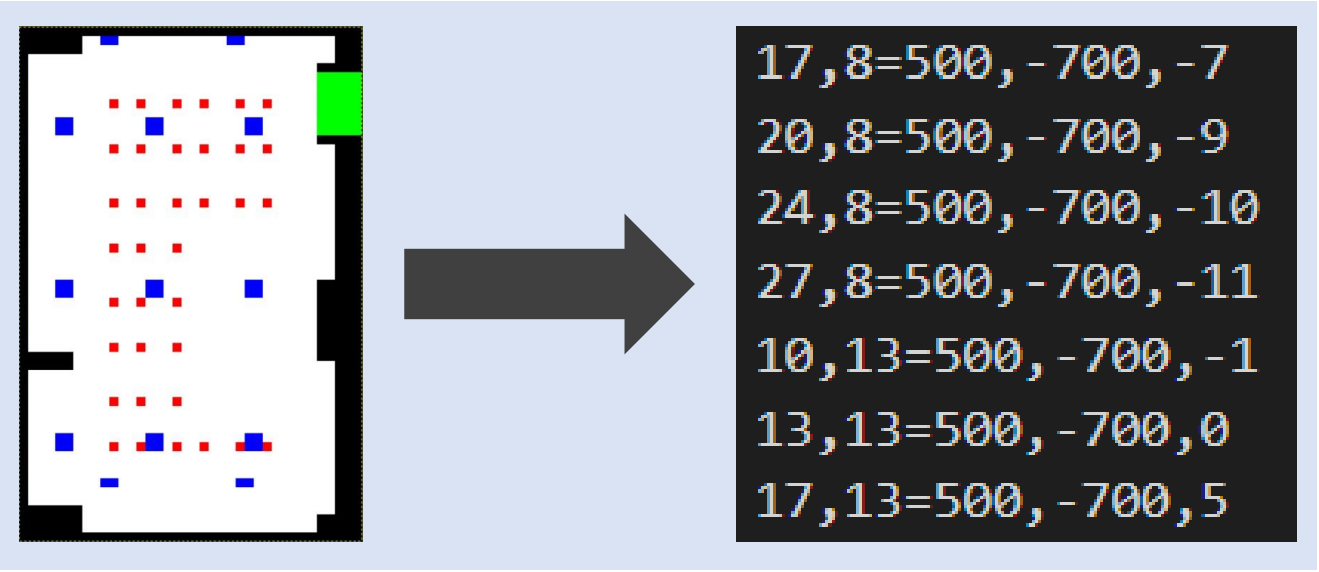
- Converting existing TXT scenarios to JSON scenarios
  - “X,Y=concentration,type,counter” into the more recent JSON format
  - This change allowed older scenarios to be brought into the newer format
  - The old format could be used to represent scenarios with the results being converted

```
17,8=500,-700,-7
20,8=500,-700,-9
24,8=500,-700,-10
27,8=500,-700,-11
10,13=500,-700,-1
13,13=500,-700,0
17,13=500,-700,5
```

```
"cells": [
    {
        "cell_id": [
            0,
            0
        ],
        "state": {
            "concentration": 0,
            "type": -300,
            "counter": -1
        }
    },
```

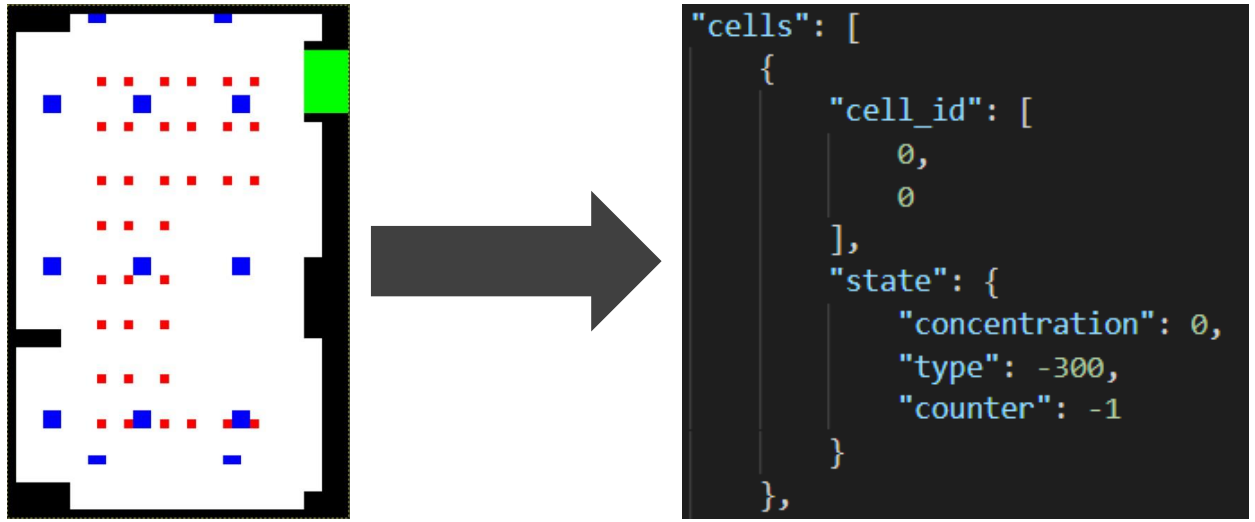
Creating 3D Scenarios - Preliminary Scripts

- Generating scenarios (old format) from an image
  - Served as a stepping stone for creating JSON scenarios from images
  - Now images could be converted to a text-based representation of the scenario



Scenario Conversion Program

- Taking elements from each of the preliminary scripts, the middle step of the older format could be dropped

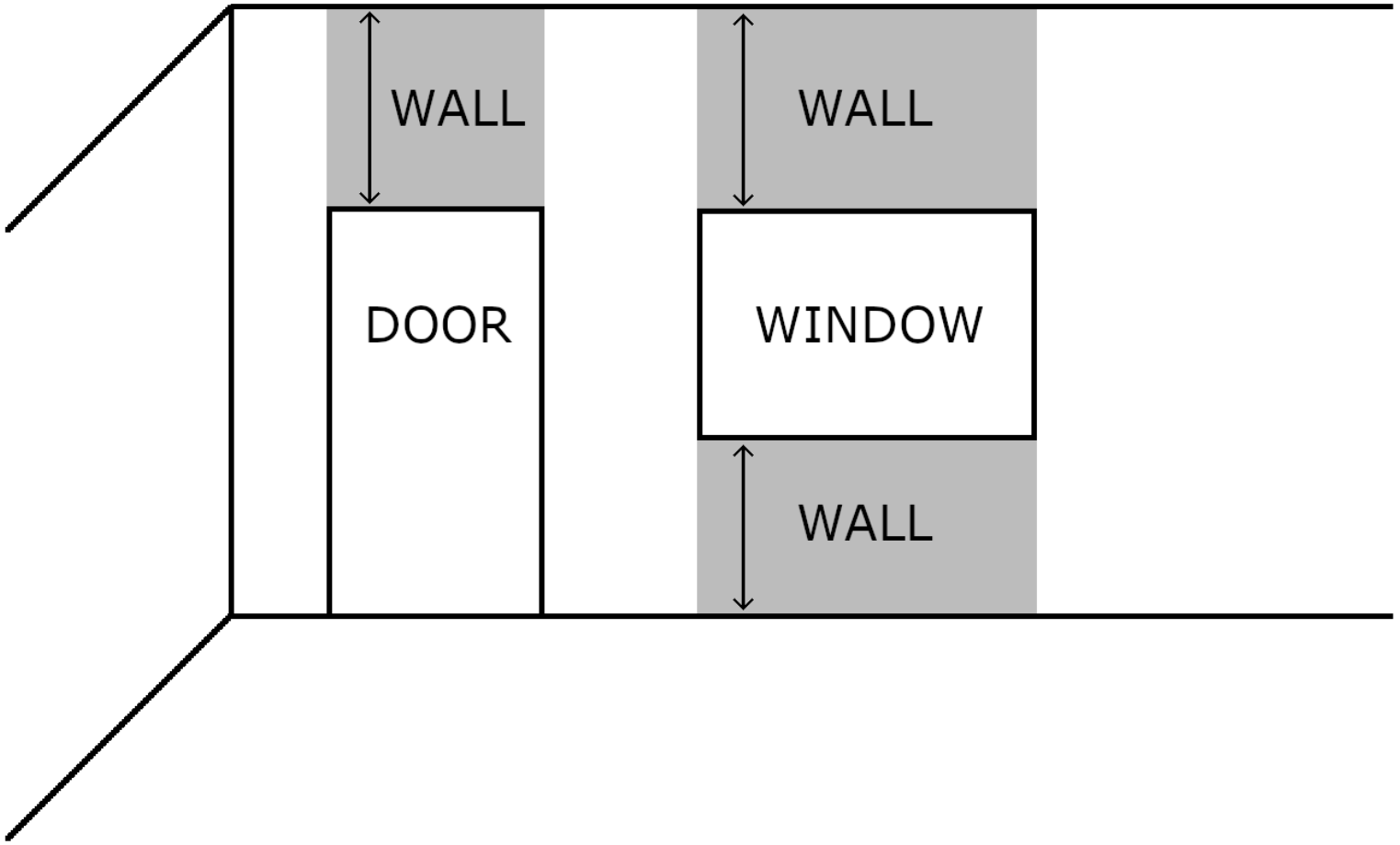


- The program is also capable of converting 2D JSON scenarios to 3D JSON scenarios

Scenario Conversion Program - 3D Scenarios

- Images needed to be turned into 3-dimensional models, not just 2D
  - 2D scenario created first – then “extended” into the 3rd dimension
  - If a 2D scenario is desired, the program will not extend the scenario
- Certain parameters needed to be defined in a configuration file
- Assumptions were made regarding how some elements of a room should be placed
  - There are walls above and below windows
  - Doors start from the ground and have walls above them

Scenario Conversion Program - 3D Scenarios



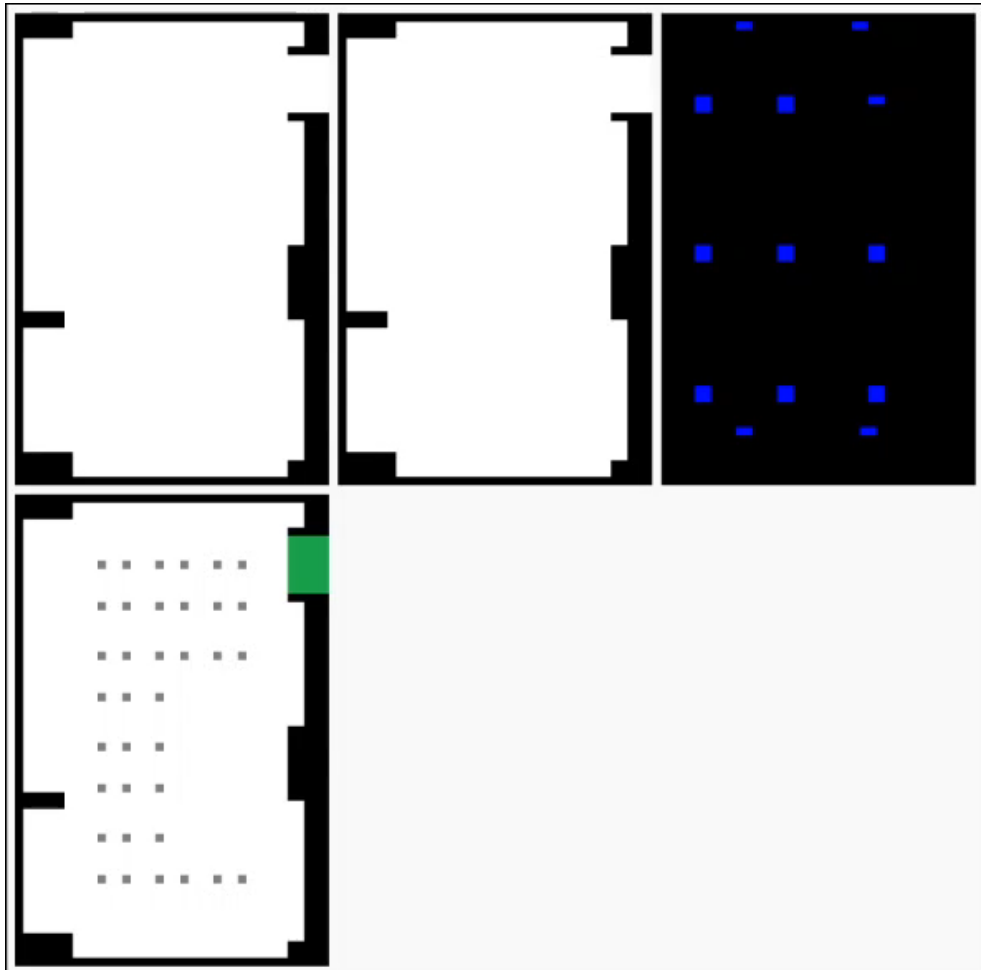
Scenario Conversion Program - Images

- Image-scenario scaling
  - Early versions of the program required the dimension of the image to match those of the desired scenario
  - The program can scale down a scenario to match user-specified dimensions
- Colour and alpha tolerance
  - The program uses colours to determine the type of cells being placed
  - Pixels are ideally solid (100% opaque) and have each value of RGB as either 0 or 255
    - White, black, red, green, blue, yellow, cyan, magenta
  - Tolerance for colours and transparency are set in the configuration file

Scenario Conversion Program - File Types

- If the input file for the program (as specified in the configuration file) is any of BMP, JPG, JPEG, or PNG, the program will attempt image conversion
- If the input file for the program is a JSON file, the program will attempt to make the (presumed) 2D scenario 3-dimensional

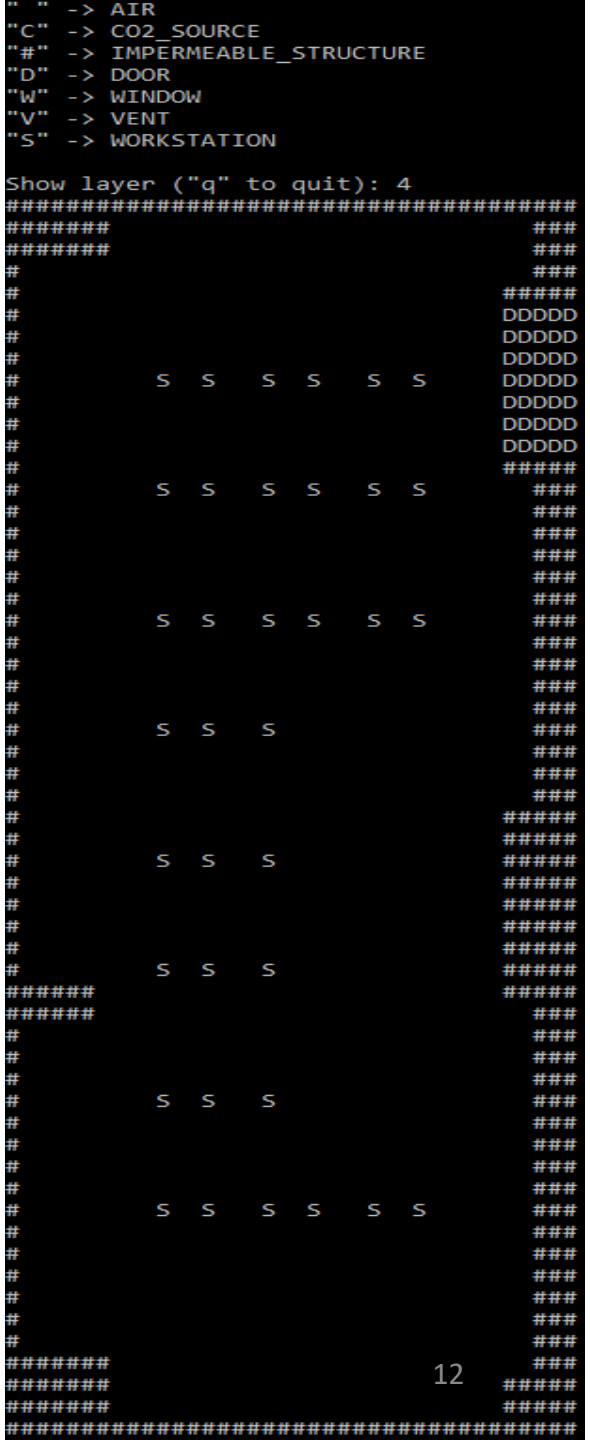
Scenario Conversion Program - Counters



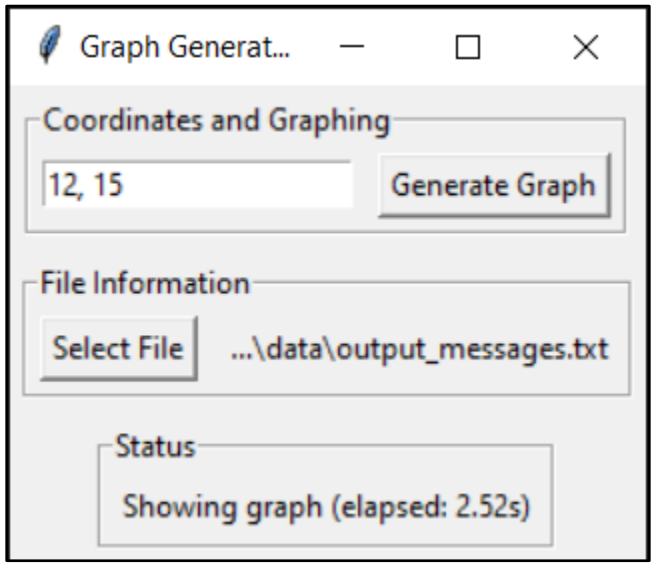
- Panels
  - Layer 7 – height of CO_2 sensors
  - Layer 4 – height of workstations
  - Layer 12 – ceiling and ventilation
  - Layer 4 – workstations (map (type) port)
- Counter values are randomly (but repeatably) assigned to each workstation
  - The counter determines when a workstation cell becomes a CO_2 source

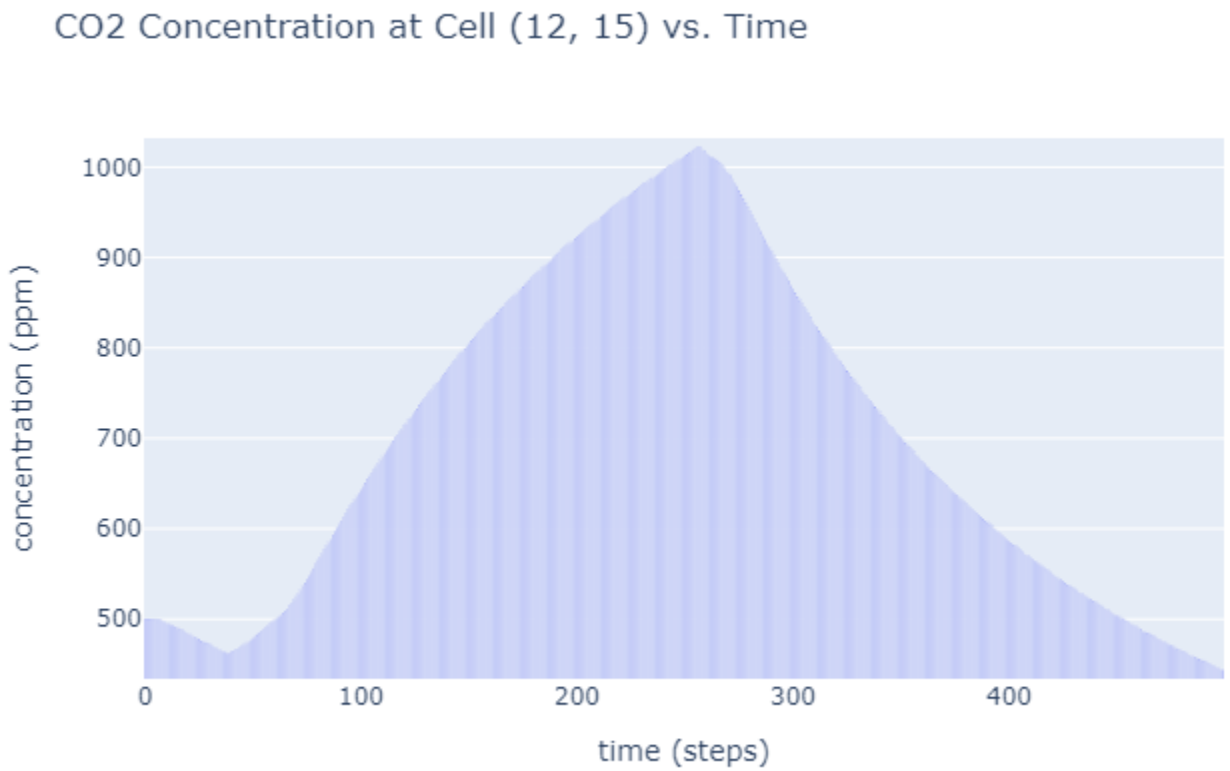
Creating Scenario Visualizations

- Cadmium simulations can take a significant amount of time to run
- The script creates a quick command line representation of a scenario from the JSON file alone (without the need to first run the simulator)
- Can show both 2D and 3D scenarios
  - Asks for which layer to show for 3D scenarios
- Helps to check that an image was converted as the user expected



Creating Graphs for CO_2 Model Simulations



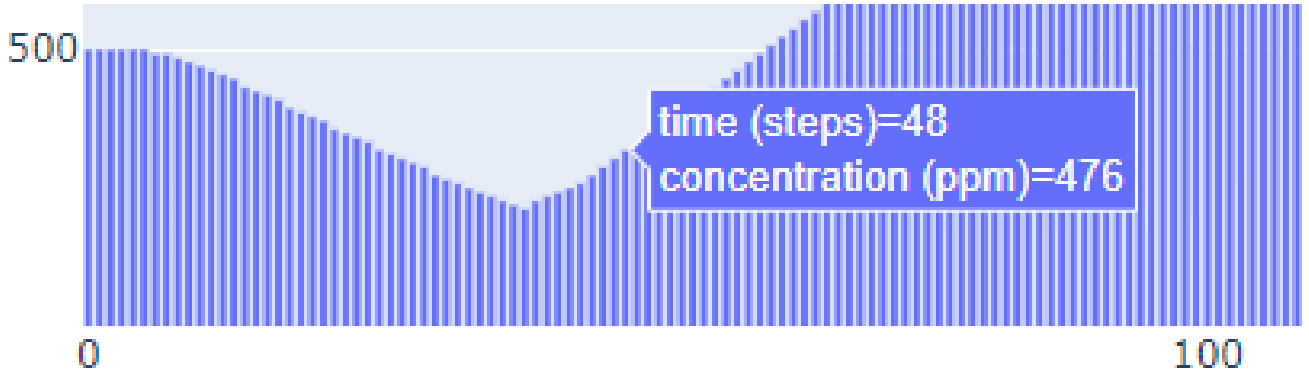


Graphing Program

- Provides a way to see the rises and falls of CO_2 concentration for a cell in the scenario
- Uses output from running the model using Cadmium
  - Works with both “output_messages.txt” and “state.txt”
  - Best to use the smallest one
- Based on Hoda’s graph-generating Java program
  - Rewritten from scratch in Python

Graphing Program

- Input file and coordinates are provided through the graphical user interface
- Graphs are generated and displayed in a web browser (opens a new tab automatically)
  - Done using a 3rd party library called Plotly
  - Graphs are interactive in the web browser



Graphing Program - Modes

- There are two different modes: transient and store
  - Transient
    - Accesses the output file for each coordinate/graph
    - Quick initial loading time, longer loading time for each graph
  - Store
    - Parses the entire file when initially loading to allow for quick graph generation
    - Slow initial loading time, quick loading time for each graph
- Time taken for loading (in both modes) depends on the size of the Cadmium output file used
  - Using a smaller output file reduces loading times
- Mode is specified on the command line

Graphing Program - Generalizing

- The program was built with the CO_2 model in mind
  - Should be able to be used for other models (assuming the format of Cadmium's output files remain the same)
  - Changes to graph labels should be the only necessary modifications
    - Easily changed through Constants.py
  - Has only been tested with the CO_2 model

```
labelX = "time (steps)"
labelY = "concentration (ppm)"
title = None  # set to None for Graph.getTitle to create a title
```

CO_2 Model Calibration

- Work in progress
- Make the simulation produce data that can match graphs of actual measurements of CO_2 concentration
- Tweak parameters of the simulation to mimic how CO_2 spreads in the real world
  - Concentration of CO_2 from ventilation
  - Time that students spend at workstations
  - Base cell concentration