`plot` Design

`plots` `greenerthumb` JSON messages from STDIN.

UI

`plot` will convert input messages to points on a line-graph. The expected message format is:

```
{
  "Name": <message_name>,
  "Timestamp": <timestamp>,
  <name>: <value>,...
}
```

Messages don't necessarily need to be `greenerthumb` messages. They just need to fit this format.

Each non-ID and non-timestamp field will become a line in the graph. The name of the line will be determined by concatenating the `<message_name>` and the field's `<name>`. The line's will be assigned unique random colors which will be displayed in a legend with the message name's on the right side of the plot.

Each line will be overlayed to allow trend comparison. To do this, each line will have units normalized to eachother. The x-axis will have units of hours scaled to the period of all the received messages. Ranges of units are presented in the legend to account for the normalization.

If received messages have the same timestamp, the newest message will overwrite the older messages.

A save button makes screenshots.

Program

```
./plot
```

Example

```
./plot
```

```
< {"Name": "Soil", "Timestamp": 0, "Moisture": 0.37}
< {"Name": "Air", "Timestamp": 3600, "Temperature": 84.5}
< {"Name": "Soil", "Timestamp": 3600, "Moisture": 0.35}
< {"Name": "Air": "Timestamp": 7200, "Temperature": 82.1}
```

This will plot 2 lines labelled 'Soil Moisture' and 'Air Temperature'. Each will have 2 points. The 'Soil'-line will start at hour 0 and finish at hour 1. The 'Air'-line will start at hour 1 and finish at hour 2. The entire plot will occupy 2 hours. The range for 'Soil Moisture' will be [0.35, 0.37] and the range for 'Air Temperature' will be [82.1, 84.5].

Pressing 's' will also save a screenshot of the plot with a unique name.