🎮 Simple-8 Fantasy Console Documentation

Simple-8 is a minimal fantasy console made in Python using Pygame and Tkinter. It lets you run short programs in a 24×24 pixel grid, great for learning code, making pixel demos, or experimenting with logic.

🔧 How It Works

Simple-8 gives you:

- A **24×24 pixel grid** you can draw on with code.
- A GUI for inputting code and variables.
- Full access to Python including `keyboard` module input.
- Buttons to run, stop, and view your code.

🖼 The Screen Layout

The Simple-8 console shows a Pygame window with:

- **The pixel grid** (24×24).
- **4 Buttons**:
  - 📱 **Click me** – Opens the code input menu.
  - 📄 **View Code** – Opens your current code in Notepad.
  - ▶ **Run** – Executes your code in a loop.
  - □ **Stop** – Stops the code and resets the grid.

📥 Code Entry Window ("Click me")

When you press **"Click me"**, you'll get a window where you can:

Field	Description
Code	Your Python code. It will be executed each loop.
Duration	Time between loops (in seconds). Can be a decimal like `0.1`.
Loops	Number of times your code will run.
px / py	Custom integer variables.
ox / oy	Arrays of 30 integers (comma-separated).

💾 You can **Save**, **Save As**, or **Load** code and variables.

🎨 Drawing on the Grid

Use these built-in functions in your code:

`update_grid(x, y, rgbcolor)`

Set the pixel at `(x, y)` to a specific color.

```
update_grid(0, 0, (255, 0, 0))  # Draw red in the top-left
```

`clear_grid()`

Resets all pixels to white.

```
clear_grid()
```

`fill_grid(rgbcolor)`

Fills the entire screen with a single color.

```
fill_grid((0, 0, 255))  # Entire screen becomes blue
```

🔁 Running Your Code

When you press **Run**:

1. Your code is executed repeatedly for `loops` times.
2. It waits `duration` seconds between each loop.
3. Code is live—draw on the grid, react to input, etc.

💡 All your variables (`px`, `py`, `ox`, `oy`) are global, so you can use them like:

```
update_grid(px, py, (0, 255, 0))
```

⌨️ Keyboard Input

Simple-8 **includes keyboard support out of the box** using the `keyboard` module.

You can just write normal Python keyboard input like this:

```
if keyboard.is_pressed('w'):
    update_grid(px, py, (255, 255, 0))  # Highlight player if W is
pressed
```

You don’t need to define anything extra—the keyboard module is already imported!

Note: Do not write `import keyboard` on the top of the code you’re gonna run, or else the code is not gonna work.

🕹 Example: Move a pixel around

```
fill_grid((255,255,255))

if keyboard.is_pressed('w'):

    py -= 1

if keyboard.is_pressed('s'):

    py += 1

if keyboard.is_pressed('a'):

    px -= 1

if keyboard.is_pressed('d'):

    px += 1

update_grid(px, py, (0, 255, 0))
```

Set `duration` to a low value like `0.1` and `loops` to a high number (like `99999`).

💾 Save and Load

When editing code, use:

- **Save** – Saves variables/code to `variables.txt`.
- **Save As** – Save to a custom `.txt` file.
- **Load** – Load any `.txt` you saved.

⚠ Uses `eval()` to read files, so only open trusted files!

□ Full Example: Diagonal Line with Delay

```
for i in range(24):
    update_grid(i, i, (255, 0, 0))
```

Set:

- Duration: `0.1`
- Loops: `24`

This draws a red line diagonally down the screen, pixel by pixel.

□ Stopping Code

If something goes wrong or you want to stop early:

- Click the □ **Stop** button.
- It will immediately halt and clear the grid.

🚨 Safety Notes

- You have full access to Python.
- There is no sandbox—avoid harmful or infinite loops.
- Always use `keyboard.is_pressed()` inside `if` blocks, not in infinite loops unless you use `break`.

🚀 Tips & Tricks

- Keep logic inside `if` or `for` blocks.
- Use `px`, `py`, `ox`, `oy` to build logic that can be controlled from the GUI.
- You can import other modules if needed—just write `import math`, etc.

🛠 Under the Hood

- Language: Python
- UI: Tkinter + Pygame
- Grid size: `24x24` pixels
- Cell size: `20x20` pixels
- Project name: **Simple-8**
- Icon: `./assets/S8_logo.png`

□ Coming Soon? (Ideas)

- Sound effects
- Virtual sprites
- Mouse or gamepad support
- Network multiplayer demos (!)