Phases:

[Main focus, Features, Diagram, Decisions, Tests, Notes, Modules and Responsibilities]

What it is:

Better optimized math visualizer

For whoe:

- People want fast graphing tool
- People that hate python
- C++ lovers
- Mathematicians

Libs:

- Cwindow

What I want:

- 2D/3D
- Camera movment
- Script hot swap ✓
- Update base on changes
- Flags
- Run from path ✓
- Auto init ✓
- load templates [Examples]
- Acces to window
- Simple UI
- Config
- Easy install ✓
- Install and write ✓
- Error handling
- Compute code SSBO
- Swaping
- Stability
- Modularity
- Plug in system for comunity
- Window system
- View ports/multiple canvas [plots]
- timming
- save output
- User I/O
- Loging/debug/Profiling
- Test/docs
- Architecture
- Live edition ✓
- Math lib
- Points
- Lines
- Surfaces
- Plot
- Polynomials
- equations
- trigonometry
- function
- integral/derivitive etc
- exp/log
- parametric functions
- Matrices/vectors
- Gradients

✓ 0.0.0 (Prototype)

Main focus:

Not optimisation just minimal workin project with no errors

Tests: None.

Notes:

- Points Render is extreamly unefficient.
- No edge cases.
- No Tests
- No CI/CD

- Weak Shader
- Weak modularity
- Just simple slow cause and testing
- Waste of mem
- Points uses Heap to much

- Good option could be SSBO
- No architecture and future plan.
- Script seg foult
- SVG like render system
- threads points gen
- only lines in SSBO
- Void*

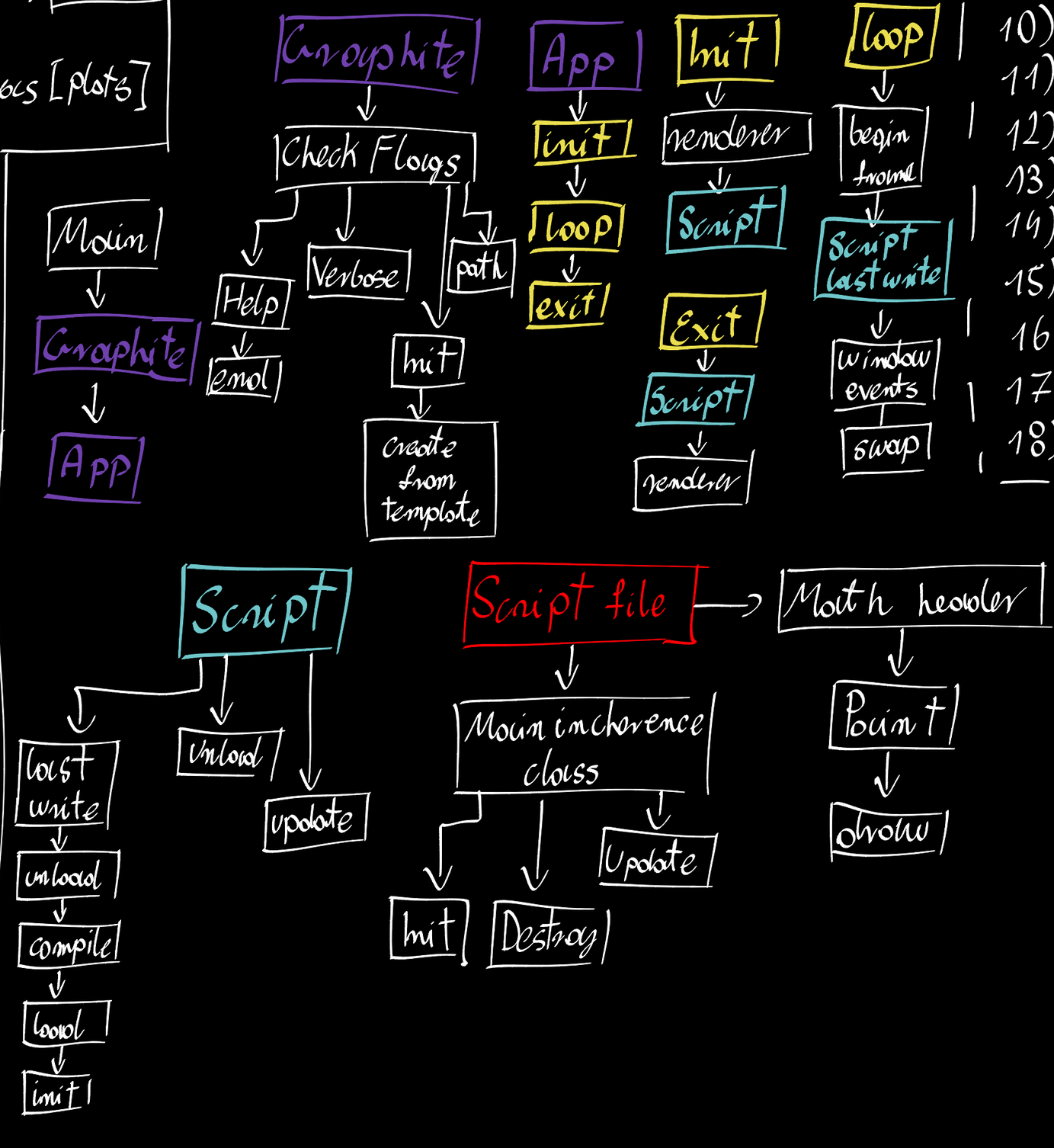
Decisions:

- Multiple points render each one creates his own uniform/shade, mesh uses void* because dll's need to alloc mem. And rendering issues will be chosed for using plot system and probably SSBO for prototype is ok.
- Dynamic script loading for easy editability we one checking last write time and compile → swap.
- Install include interfaces for compilation and vs code snipset.

Features:

1) Window initialization.
2) Script run-time compile.
3) Simple script interface.
4) Point class with render.
5) Multiple points render.
6) Simple sin example.
7) Simple instalation via cmake.
8) Readme.
9) License.
10) Code of conduct.
11) Contributing.
12) Security.
13) Issue templates.
14) Run with path provide.
15) Simple Flags
16) Init if does'nt exist.
17) App main loop.
18) Instaling interfaces.

Diagram:



Modules and responsibilites:

v1.0.0 - Stability, Modularity, Points optimization

Main Focus:

- Render Draw optimization
- Runtime all seg fault (sandbox)
- Single Shader, Mesh
- SSBO
- Plot class
- Architecture (modularity)
- Edge cases
- Tests
- Stability
- sep layers
- Examples
- init with template
- CI/CD
- Bin build
- Line (Math)
- Better Point class
- Future
- UI

Tests:

Notes:

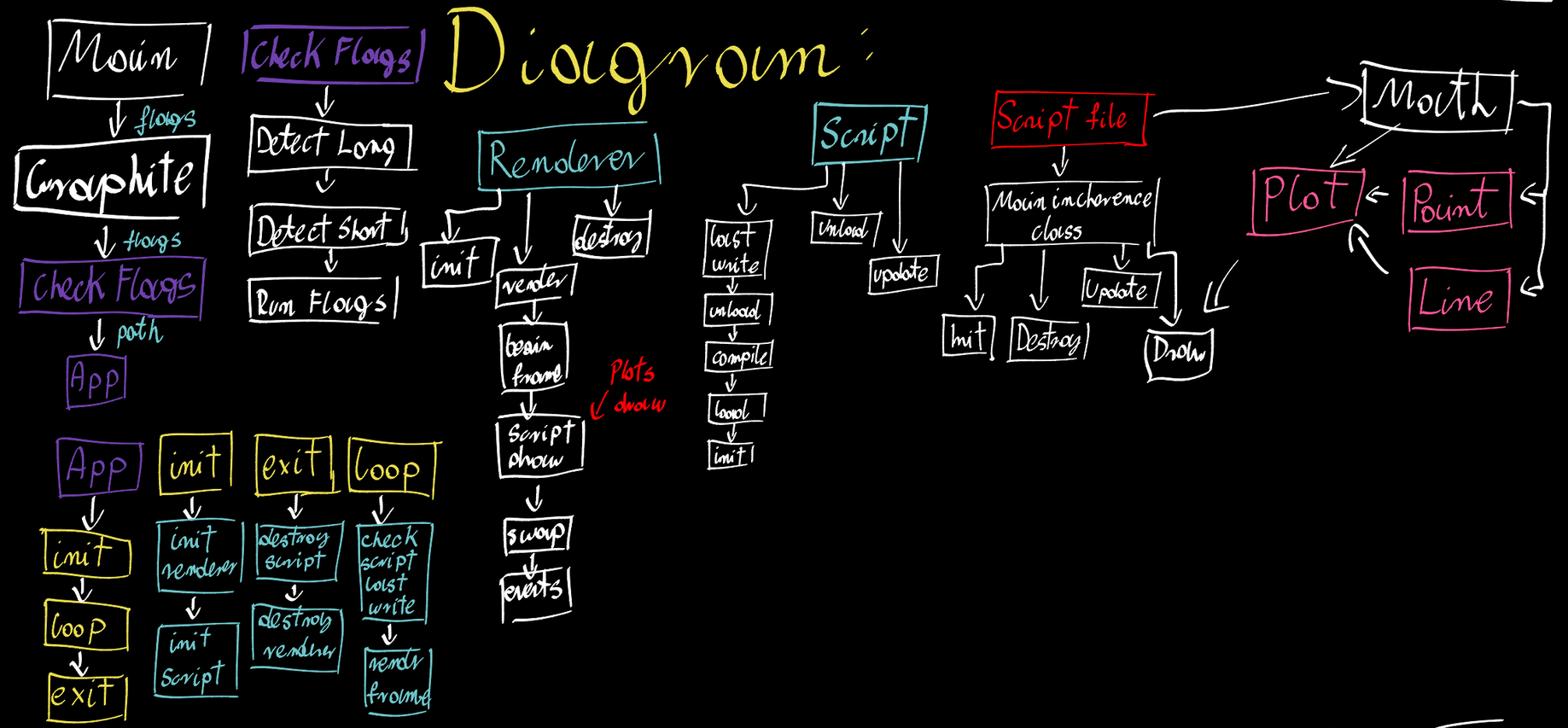
- 3 Packets
  - stability
  - optimization
  - enchance

Decisions:

Flags with params we can mash sequence of flags like -tw where both takes params then w len we are setting them first for t then for w. params can be autocomplete.

Plot creates SSBO's with points and lines then shader renders it for lines is sorted. Plot is used for render in script in Draw method. Plot mainly is for memory optimization

Diagram:



Features:

- Camera movement
- Camera zoom
- Plot class
- Line class

Modules and Responsibilities:

Graphite -

App -

Renderer -

Script -

Plot -

Point -

Line -

Future: