Curs 1 - Preliminarii

Pentru a rula template-ul

- se include header-ul *square-app.h* la inceputul fisierului *main.cpp*:

```
include "fmi/01-01-square/square-app.h"
```

- se obtine o instanta a clasei *SquareApp* si se ruleaza:

```
SquareApp::getInstance()->run(argc, argv);
```

Unde se gasesc fisierele

- header:
  - in `include/fmi/{nume_folder}/{nume_fisier.h}`
  - se include in alte fisiere cu `#include "fmi/{nume_folder}/{nume_fisier.h}"`
- sursa:
  - in `src/fmi/{nume_folder}/{nume_fisier.cpp}`
  - se include in alte fisiere cu `#include "fmi/{nume_folder}/{nume_fisier.h}"`
- shader:
  - in `shaders/{nume_folder}/{nume_fisier.vert/.frag}`
- textura:
  - in `textures/{nume_folder}/{nume_fisier}`

Cum se creeaza o noua aplicatie

- se creeaza un fisier header (.h) nou cu definitia clasei
- eventual se adauga shadere sau texturi
- se adauga implementarea intr-un fisier sursa (.cpp)
  - trebuie sa contina la inceput `include "{clasa}.h"`
- in main.cpp, se obtine o instanta a clasei si se ruleaza
  `Clasa::getInstance()->run(argc, argv);`
  - trebuie sa contina la inceput `include "{clasa}.h"`

Alte informatii

Structura unei clase pentru o aplicatie

Clasa template, precum si celelalte exemple, sunt clase singleton (la un moment dat exista cel mult o instanta a clasei) pentru a satisface cerintele pentru glut:

- glutDisplayFunc() si glutCloseFunc() glut necesita ca parametru o functie statica
  Din acest motiv functiile render() si cleanup() sunt statice si se folosesc the *instance* pentru a apela alte metode non-statice

Aceasta este doar o decizie de implementare, iar alte variante pot exista.

- **initialize()**

Functia care are ca scop crearea de shadere, texturi si obiecte.

- **render()**

Este functia care se ocupa de desenarea pe ecran. Poate apela si functi de desenare ale unor obiecte (ex: Square)

- **cleanup()**

Elibereaza resursele folosite in timpul rularii programului, dupa terminarea acestuia

- **run()**

Initializeaza glut si ruleaza aplicatia.