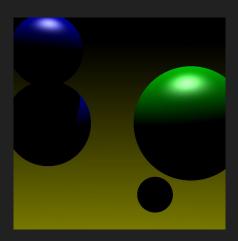
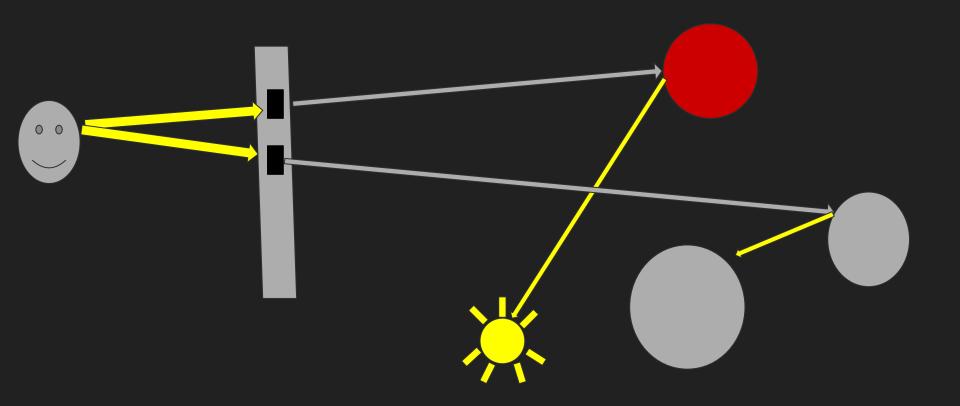


## Ce que fait le programme

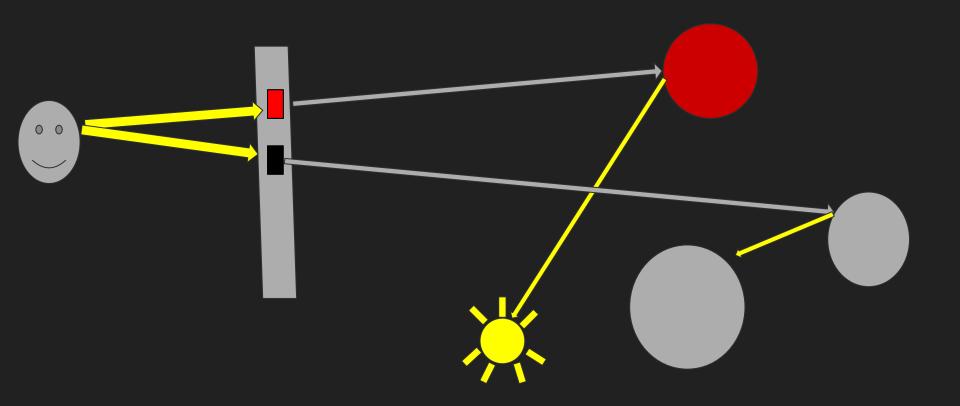
- Rendu d'image à partir d'une description par dans une fonction main,
- Rendu d'objet simple tel que ne sphère ou un plan (Le rendu de triangle a échoué),
- Calcul de l'illumination avec le modèle de Phong,
- Le calcul de la réflexion a échoué,
- Rendu d'image avec un moteur utilisant le multithreading.



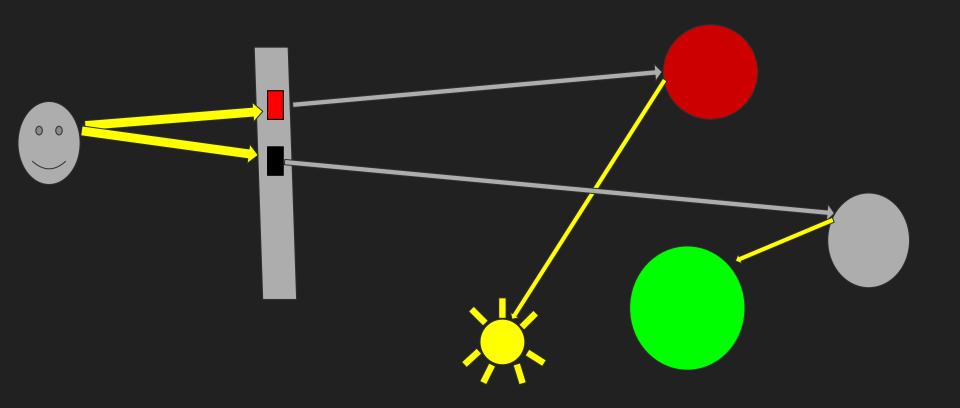
# Illustration : Ce que fait le moteur



# Illustration : Ce que fait le moteur

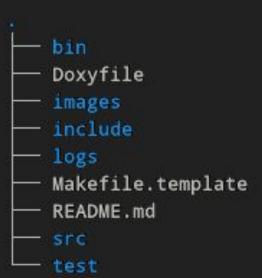


# Illustration : Ce que fait le moteur

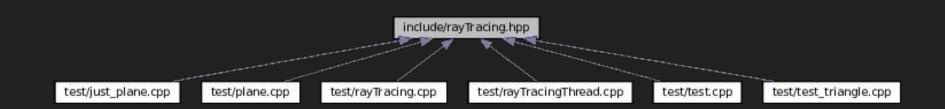


## Architecture du projet

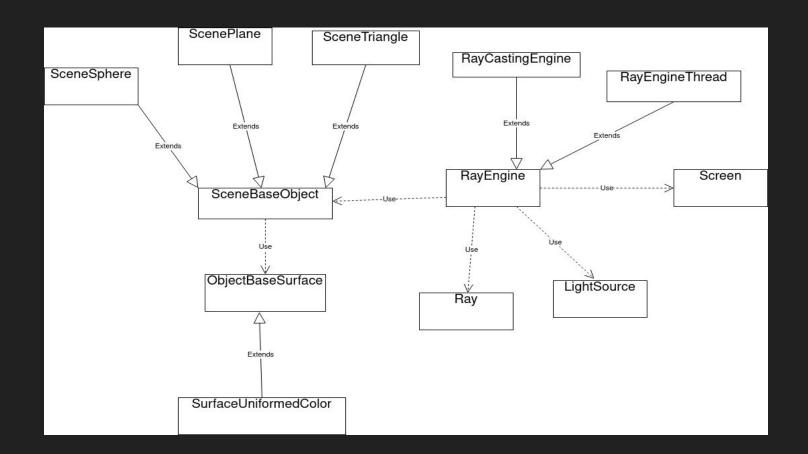
- bin : ensemble des fichiers binaires,
- images : répertoire des fichiers de sortie,
- include : répertoire des headers,
- logs : répertoires des fichiers de logs utiles pour le débuggage,
- src : répertoire des fichiers sources,
- test : répertoire des fichiers de test,



## Point d'entrée du projet : RayTracing.hpp



### Graphe des relations entre les différentes classes principales :



#### SceneBaseObject # m surface # m reflect # m refract # m spec reflect # m diff reflect # m amb reflect # m shine # MIN PROXIMITY + SceneBaseObject() + SceneBaseObject() + SceneBaseObject() + SceneBaseObject() + ~SceneBaseObject() + intersecDist() + getNormal() + getSurface() + getColor() + get spec reflect() + get diff reflect() + get amb reflect() + get shine() + get reflect() ScenePlane SceneSphere SceneCube SceneTriangle + ScenePlane() + SceneSphere() + SceneCube() + ScenePlane() + SceneTriangle() + SceneSphere() + SceneCube() + ScenePlane() + intersecDist() + intersecDist() + SceneCube() + getNormal() + getNormal() + getNormal() + ~SceneCube() + intersecDist() + aetColor() + getColor() + getColor()

# La classe SceneBaseObject

- Classe servant de d'interface avec la classe RayEngine
- Classe mère de toutes les classes objets

### RavEngine # m rays # m objects # m sources # m screen # m obs pos # m background color # m amb lighting # m deepth # HORIZON + RavEngine() + RavEngine() + RavEngine() + RayEngine() + ~RavEngine() + compute() + aet screen() + generate fundamental rays()

### RayCastingEngine

- + RayCastingEngine()
- + RayCastingEngine()
- + RayCastingEngine()
- + compute()
- + get intersection()
- + get\_reachable\_sources()
- + generate reflection ray()
- + generate reflection ray()
- + compute color()
- + step()
- + intermediairy step()

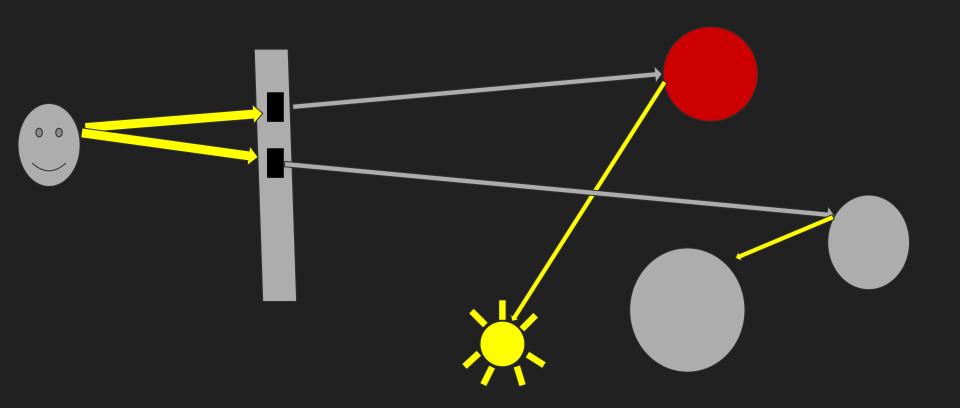
### RayEngineThread

- + RavEngineThread()
- + RayEngineThread()
- + RayEngineThread()
- + compute()
- + get intersection()
- + get reachable sources()
- + generate reflection ray() + generate reflection ray()
- + compute color()
- + step()
- + intermediairy step()

## La classe RayEngine

- Classe mère de toutes les classes ray engine,
- L'implémentation d'un nouveau moteur de rendu se fait par la redéfinition de la méthode compute.

## Le multithreading :



## Les librairies utilisées

spdlog :: C++ , header-only C++ logger for C++11, adapté au multithreading.

## Améliorations possibles du projet

- Améliorer la prise en compte des propriétés de réflexion et de réfraction,
- Caractérisé les coefficients d'illumination par un vecteur,
- Ajout d'objets triangle dans la librairie des objets visibles,
- Implémentation de la gestion des langages de description de scène,
- Augmenter la bibliothèque des objets modélisables,
- Améliorer la prise en charge du multithreading.