！下划线命名法！

问题:

内存管理？？

名称类型？？ char [32] / string

FBX读取没算切线

Over函数没写

Const没写

9.22.2022

Idea: pass的构建按照 pass\_layout,具体构建需要遍历object的type 及内部mesh的类型，输入输出texture的名称

Done: cpu基础类构建，

Todo:在render\_system 里构建pass，具体到 render的子类里面构建

9.23.2022

Idea: stre\_render 类全部封装为抽象类， 具体变量保存在子类中。

base\_type 也保存在stre\_render中，使外部用相同的基础类型

所有数据都保存在resource的data\_ptr中，只能在cg\_resource\_factory初始化和读取

每个资源类型必须有自己的结构体保存自己的数据（包括父类的数据）。

data\_ptr头储存为内容的结构体，后面为数组缓存区域，需要调整内容的指针指向这些区域

一个子物体一个材质一张贴图

Done: cpu基础类构建,内存的管理结构，fbx静态读取

Todo: 动画部分

9.24.2022

Idea: resource 具有布局引导render构建GPU资源

外部没有必要控制GPU的指令和内容

不同的render\_system可以用不同的render\_api,

gpu\_resource有多个gpu\_resource\_element以应对一个物体强耦合的多个资源

render内只具体化构建gpu\_resource\_element

Done：gpu资源构建的结构

Todo ：render内具体化构建

9.25.2022

Idea: 资源从 CPU到GPU

1.利用reource\_factory构建默认的resource  
 2.再利用resource 构建对应物体类型  
 3.物体类型的构造函数中构建 GPU layout

Done：gpu\_resource 的具体构建

Todo ：pass的具体构建

启发：

1. 类里面有该变量或指针（强关联，需要在意include结构），也可以在别的地方用数组保存用id识别
2. 父类包含许多获取的虚函数但不拥有变量，子类继承该虚函数并拥有私有变量，只允许用户使用父类，保证变量安全性
3. 工厂模式构建子类但返回父类指针，保证dll按父类的结构执行