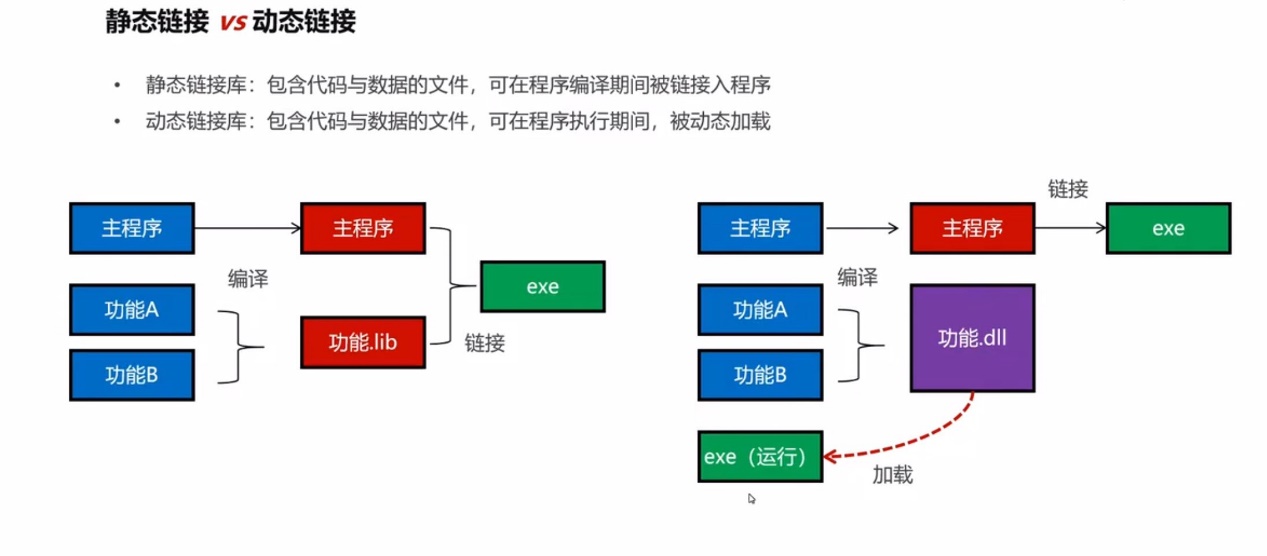
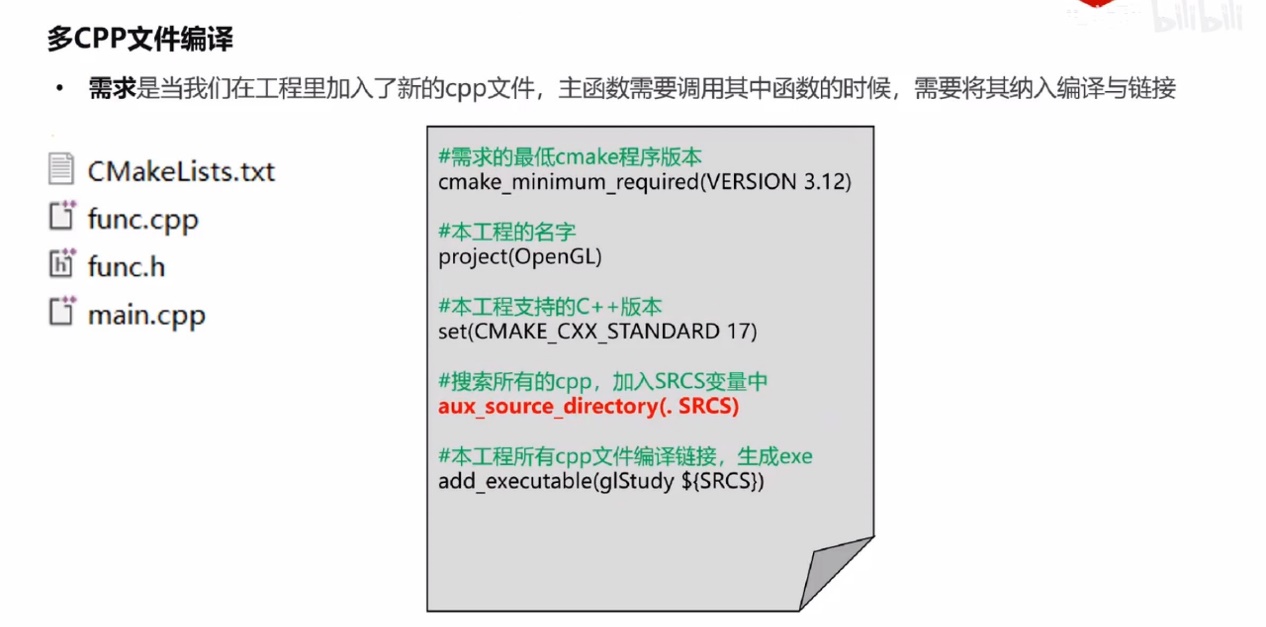
CMake使用

1. 静态链接.lib和动态链接.dll



1. CMake的静态链接

## 多cpp文件同目录下链接



CMakeLists.txt的内容

#需求的最低cmake程序版本

cmake\_minimum\_required(VERSION 3.12)

#本工程的名字括号内是名字，**可自拟**

project(test)

#本工程支持的C++版本

set(CMAKE\_CXX\_STANDARD 17)

#搜索所有的cpp，加入SRCS变量中source的缩写，**可自拟**。“.”表示当前目录

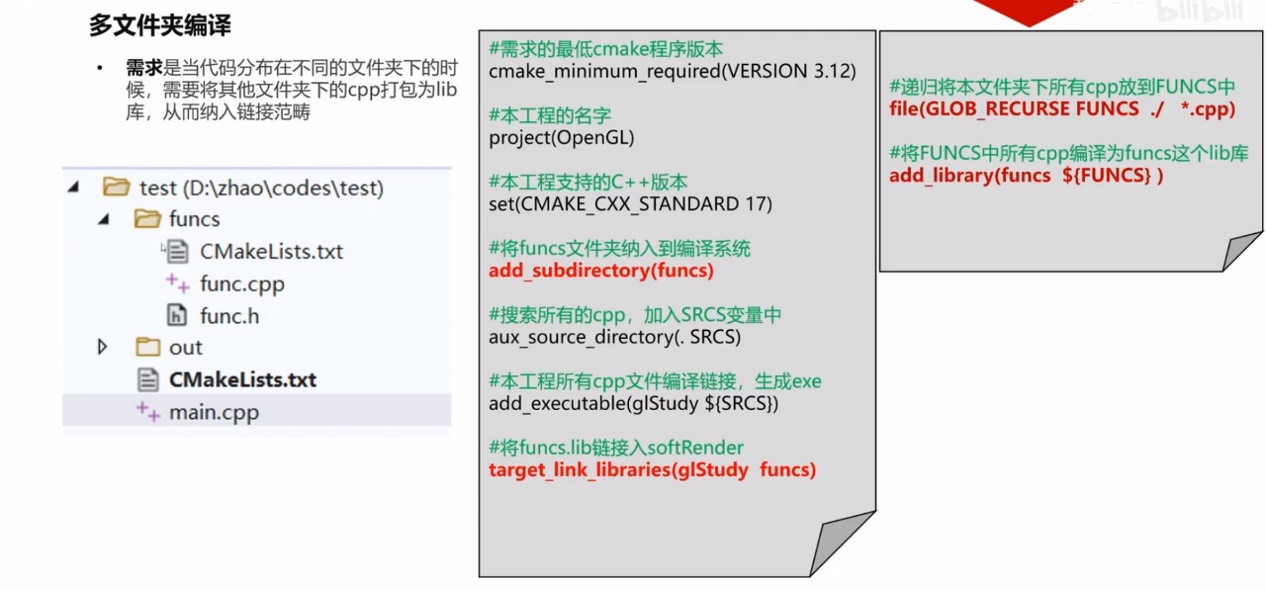
aux\_source\_directory(. SRCS)

#本工程所有cpp文件编译链接，生成exe，glStudy是可执行程序名字，**可自拟**

add\_executable(glStudy ${SRCS})

## 不同目录下编译（静态链接）

有两个CMakeLists.txt



**Main.cpp的CMakeLists.txt内容：**

#需求的最低cmake程序版本

cmake\_minimum\_required(VERSION 3.12)

#本工程的名字括号内是名字，**可自拟**

project(test)

#本工程支持的C++版本

set(CMAKE\_CXX\_STANDARD 17)

#搜索所有的cpp，加入SRCS变量中source的缩写，**可自拟**。“.”表示当前目录

aux\_source\_directory(. SRCS)

#本工程所有cpp文件编译链接，生成exe，glStudy是可执行程序名字，**可自拟**

add\_executable(glStudy ${SRCS})

#将funcs.lib链接入softRender

target\_link\_libraries(glStudy myFuncs)

（如果这里的“myFuncs”已经是一个.lib文件，可以忽略递归合并成.lib文件的步骤，也就是说只需要一个CMakeLists.txt即可。.lib文件在链接阶段被连接器链接.）

**funcs的CMakeLists.txt内容：**

#递归将本文件夹下所有cpp放到FUNCS中

file(GLOB\_RECURSE FUNCS ./ \*.cpp)

#将FUNCS中所有cpp编译为myFuncs这个lib库

add\_library(myFuncs ${FUNCS} )

1. CMake的动态链接



在**Main.cpp的CMakeLists.txt中添加的内容：**

#把需要拷贝的资源路径都放到ASSETS里

file(GLOB ASSETS "./assets" "thirdParty/assimp-vc143-mtd.dIl")

#把ASSETS指代的目录集合的内容，“都拷贝到可执行文件目录下

#CMAKE\_BINARY\_DIR是宏，CMake的二进制可执行文件

file(COPY ${ASSETS)}DESTINATION ${CMAKE\_BINARY\_DIR})

lib\.dll和include文件

任何库的包含分为两个文件：include和lib。Include文件指明功能函数的定义（头文件.h）lib文件实现功能函数

