Alt键在 文件夹中显示tool（设置文件夹中的文件的后缀名的显示）

Alt+Tab键退出游戏

IncredBuild联合编译，使得在同一网段的闲置电脑编译本机程序。

Unicode 文件的编码问题：将相应的文件重新保存一回。

2016-4-12

两个相似的文件对比有哪些不同:

不是一个一个的看，文件量很大，而是将两个文件使用正则表达式将要比较的内容输出到两个.txt（egrep -oi "([a-zA-Z]:)?([\\\\[a-zA-Z0-9\_.-](file:///\\[a-zA-Z0-9_.-)]+)+" projects/win32/graphic/graphic.vcproj|sort|uniq(git bash shell 命令行中)

），然后使用compareBeyond. 使用Nopad++ 可以下载一个ToolBucket插件，放在nopad++的安装路径下。使用正则表达式匹配内容，可以recoding.

2016-4-15

P4:  1. 首先checkout下本地要修改的文件， 在右面的窗口的pending中写注释

2. 进行修改

3. sumbit

Git:

1. Git branch –r 查看分子
2. 首先git stash save本地的修改 ， 然后git pull 更新从server的最新代码 ， 然后git stash pop 出你本地的修改， 再 git commit –m “message  ti jiao ” , 再使用arc diff （前提是安装了phabricator）

Vs2015中ctrl+F7编译单个文件

Git add 修改的文件/新添加的文件

1，找不到stdx.h文件时打开对应文件的路劲查找该路劲下是否有该文件

2，在vs项目中add exsit item

3, function no define check {} 配对

Git reset HEAD “已经缓冲的文件（取消被提交）”

Git clone xx  +（clone到的目录）

git clone [git@10.0.0.99:ACII/ACII.git](mailto:git@10.0.0.99:ACII/ACII.git) ACII/

git reset  HEAD –hard //remove all not commited changes

git中增加了文件但是在git status中没有， 是因为有些文件被过滤了使用如下命令查看哪些文件被过滤了：

cat  .gitignore

2016-4-26

Static\_cast

1.用于类层次结构中基类和子类之间指针和或引用的转换

（进行上行转换，把子类的指针或引用转换成基类）是安全的

（进行下行转换，把基类指针或引用转换成子类表示时，由于没有动态类型检查，所以不安全）

2，用于基本类型之间的转换，如把int转成char,把int转成enum.

3，把控指针转成目标类型的空指针

4，把任何类型的表达式转成void类型

注意: static\_cast 不能转换掉exdivssion的cost、volitale、或者\_unaligned属性

Dynamic\_cast

用法：Dynamic\_cast<type-id>(exdivssion)

Dynamic\_cast主要用于类层次间的上行转换和下行转换，还可以用于类之间的交叉转换。

在类层次间进行上行转换时，dynamic\_cast和static\_cast的效果是一样的。  
   在进行下行转换时，dynamic\_cast具有类型检查的功能，比static\_cast更安全。

Const\_cast

常量指针被转化成非常量指针，并且仍然指向原来的对象。

常量引用被转换成非常量引用，并且仍然指向原来的对象；常量对象被转换成非常量对象。

抽象类没有必要写函数体，因为写了会占用空间。

纯函数一定不会被调用，所以抽象类不允许创建对象，只能用它来指向子类对象或引用子类对象。

关于git,

如果现在大家没法提交的话, 可以发给我一个patch我帮大家提交.

相关命令:

首先查看commit的id

git log -1 HEAD

然后生成patch

git format-patch -1 <id>

然后把生成的patch 发给我

git checkout Dev\_ps4

git pull origin branch\_name

git checkout branch\_name

git merge –abort

checkout -f -B Dev\_ps4 remotes/origin/Dev\_ps4 –

2016-4-29

在项目中看是否所有的工程被包含进去了（一键编译）

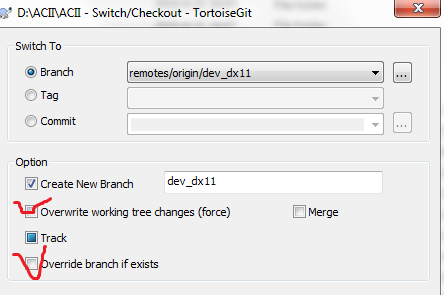
修改了.vcproj的文件，需要重新rebuild一遍。

git reset --hard 5e8d27952fe4fca5c11fe7b17c9294b3e691ddf0

2016-5-6

HEAD detached at origin/VS2015\_xb1

使用git遇到这种情况时，本地的修改又是可以忽略的，可以将以下红色两个勾勾选上。



                                                                                    C++代码优化

一. 优化之前

在进行优化之前，我们首先应该做的是发现我们代码的瓶颈（bottleneck）在哪里。然而当你做这件事情的时候切忌从一个debug-version进行推断，因为debug-version中包含了许多额外的代码。一个debug-version可执行体要比release-version大出40%。那些额外的代码都是用来支持调试的，比如说符号的查找。大多数实现都为debug-version和release-version提供了不同的operator new以及库函数。而且，一个release-version的执行体可能已经通过多种途径进行了优化，包括不必要的临时对象的消除，循环展开，把对象移入寄存器，内联等等。

另外，我们要把调试和优化区分开来，它们是在完成不同的任务。 debug-version 是用来追捕bugs以及检查程序是否有逻辑上的问题。release-version则是用来做一些性能上的调整以及进行优化。

下面就让我们来看看有哪些代码优化技术吧：

二. 声明的放置

（1）       请使用初始化而不是赋值

1. 把声明放在合适的位置上

（3）       初始化列表

                   初始化列表直接是在数据成员声明的时候就进行了初始化，因此它只执行了一次copy constructor，数据成员都是基本类型的话，两者产生的汇编代码是相同的。

1. postfix VS prefix 运算符

三. 内联函数

1. 好处大家都知道。（嵌入，替换宏，容易调试）
2. 编译器有权利忽略这个建议（大小，是否有变量声明，循环，switch等）
3. 缺点
   1. 想内联的不内联，增加程序体大小，函数体的多次拷贝
   2. 维护，每改变内联函数，使用他们的地方需要从新编译，而不是连接。

四. 变量

* 1. 使用32位数据类型
  2. 明智使用有符号类型数据：整型到浮点转化时，使用大于16位的有符号整型比较快。因为x86构架中提供了从有符号整型转化到浮点型的指令，但没有提供从无符号整型转化到浮点的指令。
  3. 总结：  
     　　无符号类型用于：   
     　　除法和余数   
     　　循环计数   
     　　数组下标  
     　　有符号类型用于：   
     　　整型到浮点的转化

五：while VS. for

       for(;;)比while(1)好for (；；)指令少，不占用寄存器，而且没有判断跳转，比while (1)

好

       六 ：用数组型代替指针

       　使用指针会使编译器很难优化它。因为缺乏有效的指针代码优化的方法，编译器总是假设指针可以访问内存的任意地方，包括分配给其他变量的储存空间。所以为了编译器产生优化得更好的代码，要避免在不必要的地方使用指针。数组型代码会让优化器减少产生不安全代码的可能性。源代码的转化是与编译器的代码发生器相结合的。从源代码层次很难控制产生的机器码。在源代码转化后检查性能是否真正提高了，再选择使用指针型还是数组型。  
         七：循环

* + 1. 充分分解小的循环。
    2. 避免没有必要的读写依赖, 数据必须在正确写入后才能再次读取.
    3. Switch 的用法：最有可能的放前面
    4. 提升循环的性能 要提升循环的性能，减少多余的常量计算非常有用（比如，不随循环变化的计算） 不好的代码(在for()中包含不变的if())

把本地函数声明为静态的(static)

1. 使用显式的并行代码

float a[100], sum1, sum2, sum3, sum4, sum；  
int i；  
sum1 = sum2 = sum3 = sum4 = 0.0；  
for (i = 0； i <； 100； i += 4)  
{  
　　sum1 += a[i]；  
　　sum2 += a[i+1]；  
　　sum3 += a[i+2]；  
　　sum4 += a[i+3]；  
}  
sum = (sum4+sum3)+(sum1+sum2)；

八 :函数   
如果一个函数在实现它的文件外未被使用的话，把它声明为静态的(static)以强制使用内部连接。否则，默 认的情况下会把函数定义为外部连接。这样可能会影响某些编译器的优化——比如，自动内联。

九 字节对齐

（1）结构，类添充

struct  
{  
    double x；  
    long k；  
    char a[5]；  
    char pad[7]；  
}

（2 ）按数据类型的长度排序本地变量。该把长的变量放在短的变量前面。如果第一个变量对齐了，其它变量就会连续的存放，而且不用填充字节自然就会对齐

十 ：代码最好简短( 讲解下原子操作Atom operation )