Hw13　事件处理与音效

黄建武

onTouchMoved - 响应触摸初始事件

```
auto touchListener = EventListenerTouchOneByOne::create();
touchListener->onTouchMoved = CC_CALLBACK_2(Thunder::onTouchMoved, this);
this->getEventDispatcher()->addEventListenerWithSceneGraphPriority(touchListener, this);
```

getDelta 获取鼠标的偏移量

```
void Thunder::onTouchMoved(Touch *touch, Event *event) {
    Vec2 delta = touch->getDelta();
}
```

CC_CALLBACK_2

```
#define CC_CALLBACK_2(__selector__,__target__, ...)
        std::bind(&__selector__,__target__, std::placeholders::_1, std::placeholders::_2, ##__VA_ARGS__)
```

__selector__：绑定要回调的函数名，注意要使用命名空间::函数名，如Thunder::onTouchMoved

__target__：绑定一个对象，一般为this

CC_CALLBACK_N的作用是让_target_对象用_selector_函数时绑定第N个参数后面参数的值

```cpp
int Test::add4(int a, int b, int c, int d) {
    return a + b + c + d;
}

int Test::add2(int i, int j) {
    return i + j;
}

// 绑定第一个参数后面的参数
auto func1 = CC_CALLLBACK_1(Test::add2, this, 10);
func1(15);  // 15 + 10

// 绑定第二个参数后面的参数
auto func2 = CC_CALLLBACK_2(Test::add4, this, 10, 11);
func2(12, 13);  // 12 + 13 + 10 + 11
```

Lambda表达式

```
touchListener->onTouchMoved = [](Touch *touch, Event *event) {
    Vec2 delta = touch->getDelta();
    // do something
}
```

CallFunc

在一系列连续的动作之后执行一个回调函数

```
Sprite* tmep = enemy;
enemy->runAction(
  Sequence::create(
    Animate::create(
      Animation::createWithSpriteFrames(explore, 0.05f, 1)
    ),
    CallFunc::create([temp] {
      temp->removeFromParentAndCleanup(true);
    }),
    nullptr
  )
);
```

COCOS2DX

EventDispatcher 使用优先级来决定监听器对事件的分发

```
addEventListenerWithFixedPriority(EventListener* listener, int fixedPriority)
```

添加一个指定优先级的事件监听器 fixedPriority越小优先级越高

优先级不能为0，已经在 SceneGraphPriority 中被使用

```
addEventListenerWithSceneGraphPriority(EventListener* listener, Node* node)
```

事件监听器的优先级基于节点node的渲染顺序

Z Order 越小的节点优先级越高，确保前面的元素能获取触控事件。

COCOS2DX

使用以上两个方法时，会对当前使用的事件监听器添加一个已注册的标记，这使得它不能够被添加多次，需要重复使用某个listener时要调用clone函数。另外，FixedPriorityListener添加完之后需要手动remove，而SceneGraphPriorityListener是跟Node绑定的，在Node的析构函数中会被移除。

hw: 小蜜蜂

要求：

1. 利用键盘事件实现飞船左右移动。
2. 利用键盘和触摸事件实现子弹发射。
3. 用自定义事件实现：子弹和陨石相距小于一定距离时，陨石爆炸，子弹消失。
4. 游戏过程中有背景音乐，发射子弹、击中陨石有音效。
5. 注意飞船、子弹的移动范围。
6. 游戏结束飞船爆炸，移除所有监听器

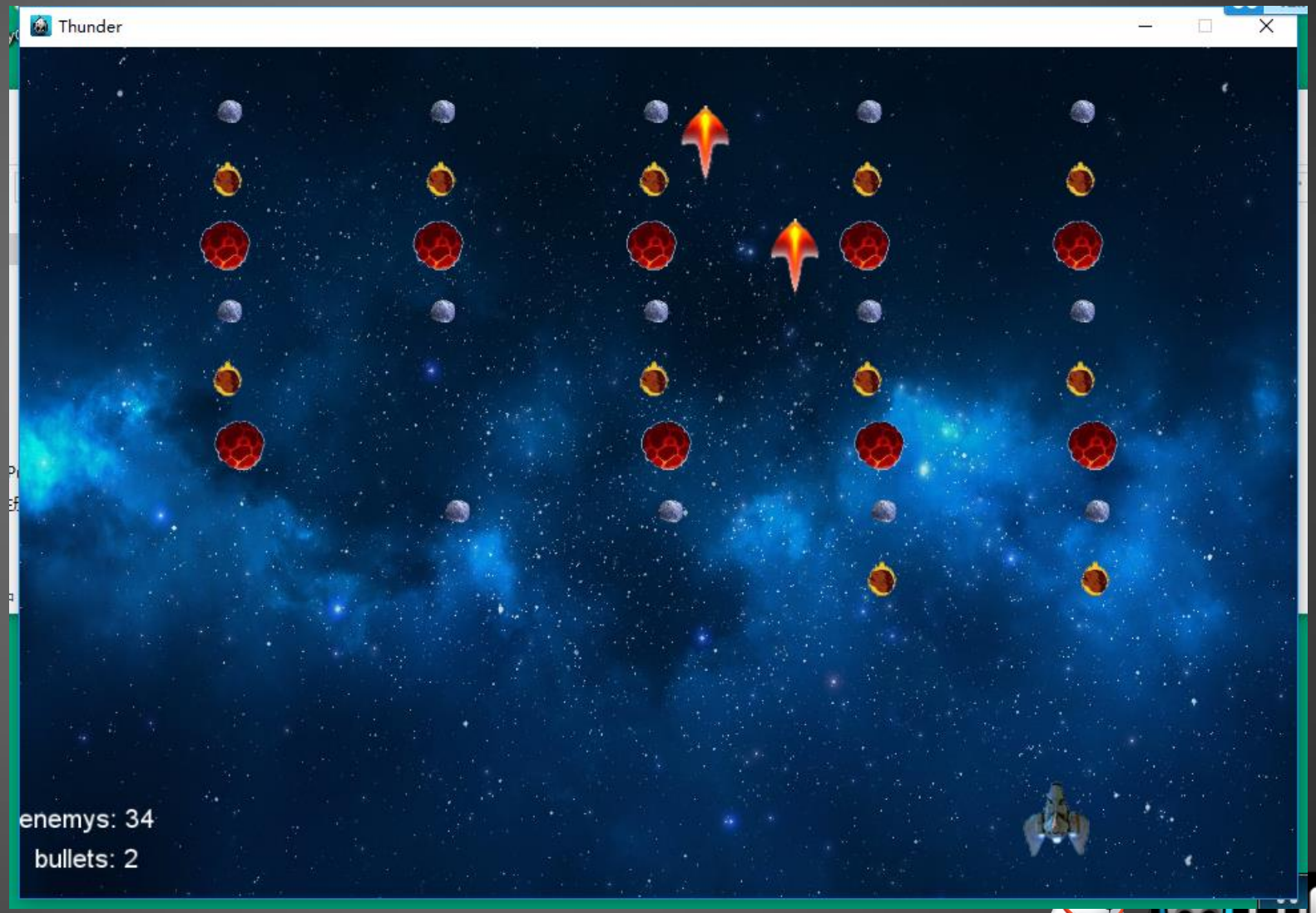
hw: 小蜜蜂

加分项：

1. 利用触摸事件实现飞船移动。（点击飞船后拖动鼠标）
2. 陨石向下移动并生成新的一行陨石
3. 子弹和陨石的数量显示正确

hw: 小蜜蜂

Demo演示：



hw: 小蜜蜂

利用键盘事件实现飞船左右移动。

键盘事件是瞬间事件，只在按下和放开时调用，如何产生按住一直移动的效果？

通过键盘事件设置状态变量，使用调度器，每隔一定时间根据状态变量更新飞船、子弹和岩石的状态、位置。

注意：按住A不放再按B，会触发键盘按下和键盘释放事件（相当于A被释放了）

hw: 小蜜蜂

用自定义事件实现：子弹和陨石相距小于一定距离时，陨石爆炸，子弹消失。

陨石那么多，如何判断子弹和哪个陨石距离相近？

最简单就是遍历。(子弹和陨石都用list存储)

同样利用调度器，每固定间隔进行一次遍历，判断子弹是否和某个陨石距离足够近，是则触发自定义事件。

```
(*it1)->getPosition().getDistance((*it2)->getPosition()) < 25
```

hw: 小蜜蜂

removeFromParentAndCleanup(bool cleanup)

从父节点移除掉节点

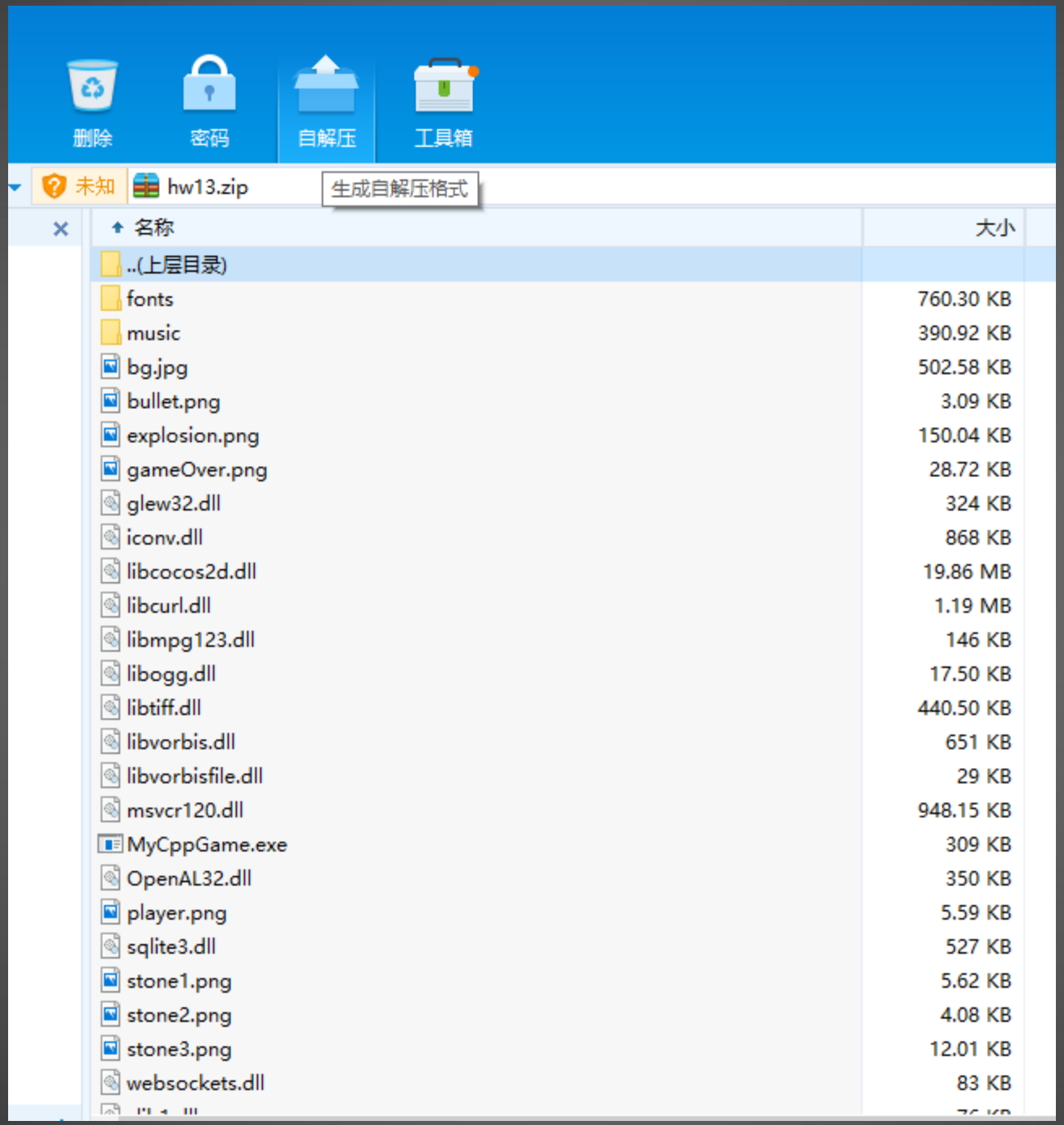
cleanup为true时同时移除该节点所有动作和回调函数

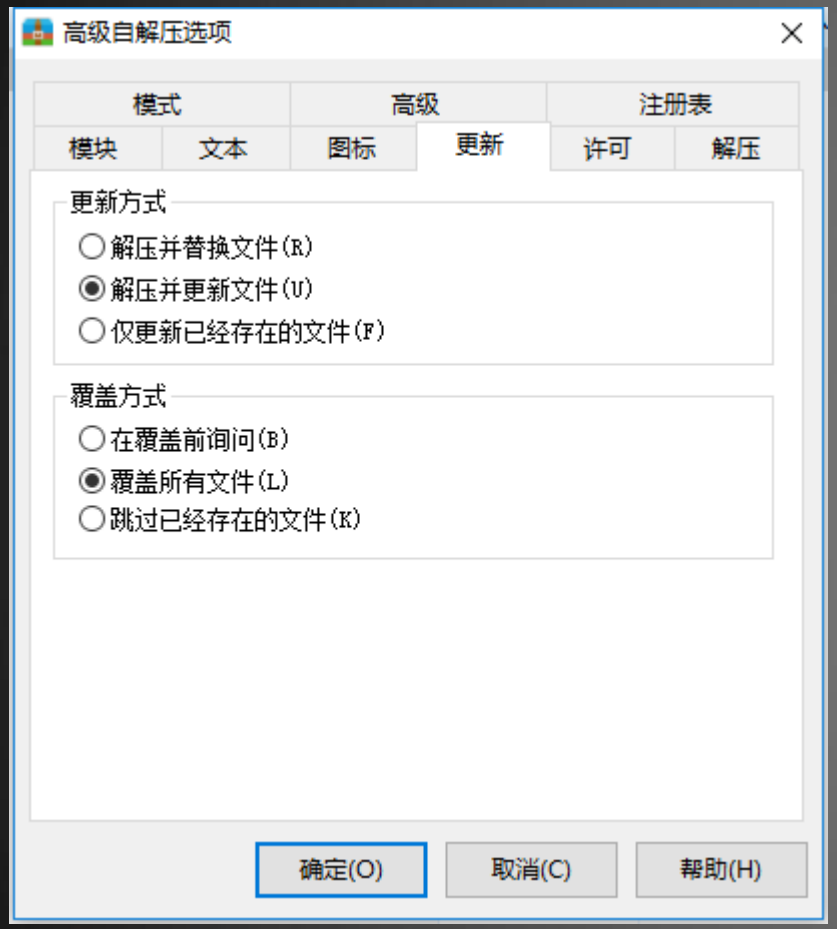
```
Sprite* tmep = enemy;
enemy->runAction(
  Sequence::create(
    Animate::create(
      Animation::createWithSpriteFrames(explore, 0.05f, 1)
    ),
    CallFunc::create([temp] {
      temp->removeFromParentAndCleanup(true);
    }),
    nullptr
  )
);
```

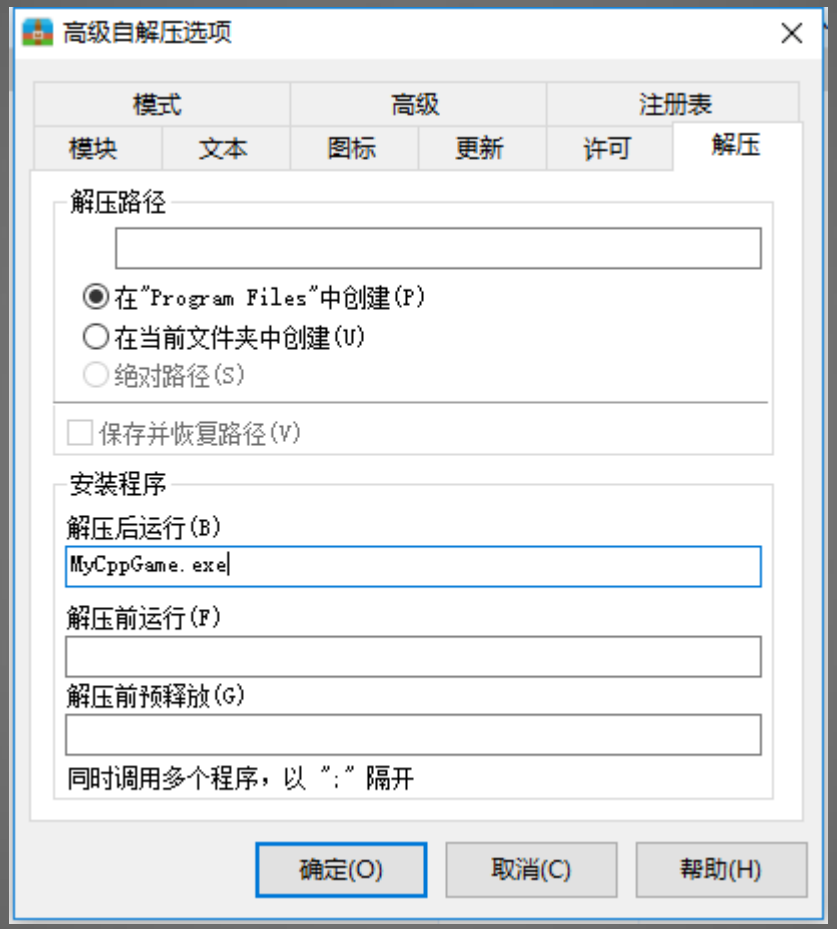
代码打包成exe

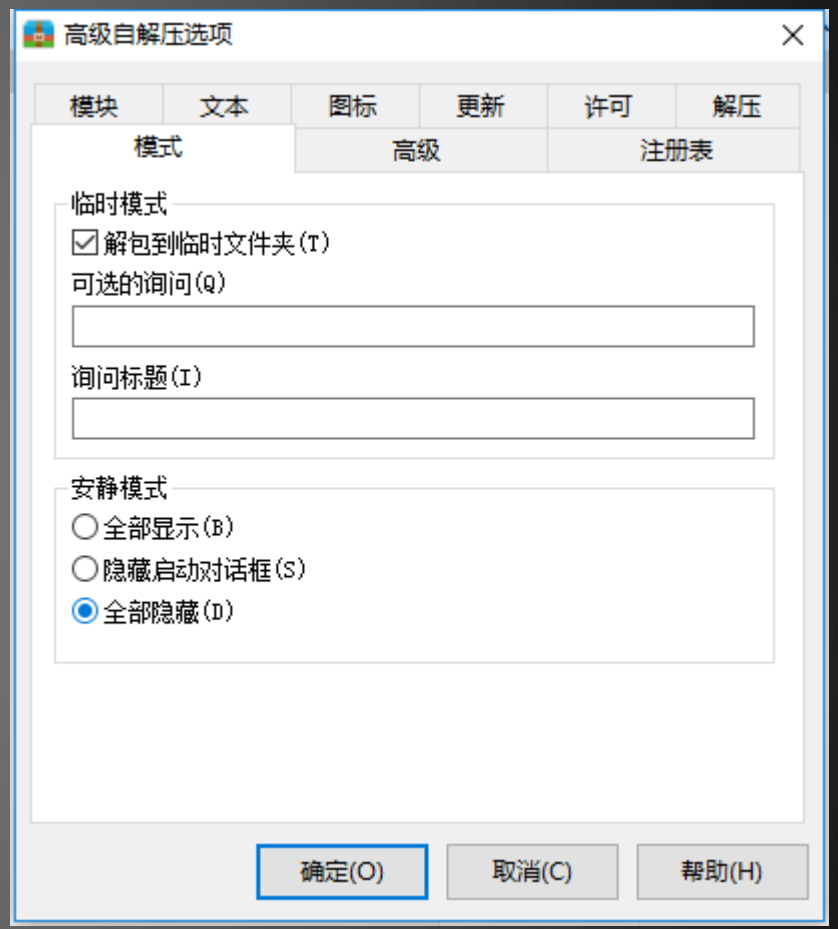
Debug.win32

名称	修改日期	类型	大小
fonts	2017/5/18 0:18	文件夹	
music	2017/5/18 0:20	文件夹	
bg.jpg	2016/5/16 16:57	JPG 文件	503 KB
bullet.png	2015/6/3 20:24	PNG 文件	4 KB
explosion.png	2016/5/17 23:29	PNG 文件	151 KB
gameOver.png	2013/1/31 12:49	PNG 文件	29 KB
player.png	2015/7/5 19:56	PNG 文件	6 KB
stone1.png	2015/6/21 16:27	PNG 文件	6 KB
stone2.png	2015/6/21 17:45	PNG 文件	5 KB
stone3.png	2015/6/21 17:43	PNG 文件	13 KB
MyCppGame.exe	2017/5/18 0:14	应用程序	309 KB
glew32.dll	2016/1/21 14:15	应用程序扩展	324 KB
iconv.dll	2016/1/21 14:15	应用程序扩展	868 KB
libcocos2d.dll	2017/5/16 22:18	应用程序扩展	20,338 KB
libcurl.dll	2016/1/21 14:15	应用程序扩展	1,222 KB
libmpg123.dll	2016/1/21 14:15	应用程序扩展	146 KB
libogg.dll	2016/1/21 14:15	应用程序扩展	18 KB
libtiff.dll	2016/1/21 14:15	应用程序扩展	441 KB
libvorbis.dll	2016/1/21 14:15	应用程序扩展	651 KB
libvorbisfile.dll	2016/1/21 14:15	应用程序扩展	29 KB
msvcr120.dll	2015/6/7 20:42	应用程序扩展	949 KB
OpenAL32.dll	2016/1/21 14:15	应用程序扩展	350 KB
sqlite3.dll	2016/1/21 14:15	应用程序扩展	527 KB
websockets.dll	2016/1/21 14:15	应用程序扩展	83 KB
zlib1.dll	2016/1/21 14:15	应用程序扩展	76 KB


删除
密码
自解压
工具箱
未知
hw13.zip
生成自解压格式
名称
大小
..(上层目录)
fonts 760.30 KB
music 390.92 KB
bg.jpg 502.58 KB
bullet.png 3.09 KB
explosion.png 150.04 KB
gameOver.png 28.72 KB
glew32.dll 324 KB
iconv.dll 868 KB
libcocos2d.dll 19.86 MB
libcurl.dll 1.19 MB
libmpg123.dll 146 KB
libogg.dll 17.50 KB
libtiff.dll 440.50 KB
libvorbis.dll 651 KB
libvorbisfile.dll 29 KB
msvcr120.dll 948.15 KB
MyCppGame.exe 309 KB
OpenAL32.dll 350 KB
player.png 5.59 KB
sqlite3.dll 527 KB
stone1.png 5.62 KB
stone2.png 4.08 KB
stone3.png 12.01 KB
websockets.dll 83 KB


高级自解压选项
模式
高级
注册表
模块
文本
图标
更新
许可
解压
更新方式
解压并替换文件(R)
解压并更新文件(U)
仅更新已经存在的文件(F)
覆盖方式
在覆盖前询问(B)
覆盖所有文件(L)
跳过已经存在的文件(K)
确定(O)
取消(C)
帮助(H)


高级自解压选项
模式
高级
注册表
模块
文本
图标
更新
许可
解压
解压路径
在"Program Files"中创建(P)
在当前文件夹中创建(U)
绝对路径(S)
保存并恢复路径(V)
安装程序
解压后运行(B)
MyCppGame.exe
解压前运行(F)
解压前预释放(G)
同时调用多个程序，以 ";" 隔开
确定(O)
取消(C)
帮助(H)


高级自解压选项
模块
文本
图标
更新
许可
解压
模式
高级
注册表
临时模式
解包到临时文件夹(T)
可选的询问(Q)
询问标题(I)
安静模式
全部显示(B)
隐藏启动对话框(S)
全部隐藏(D)
确定(O)
取消(C)
帮助(H)


中山大學
SUN YAT-SEN UNIVERSITY


COCOS2DX

THE END

THANKS FOR WATCHING