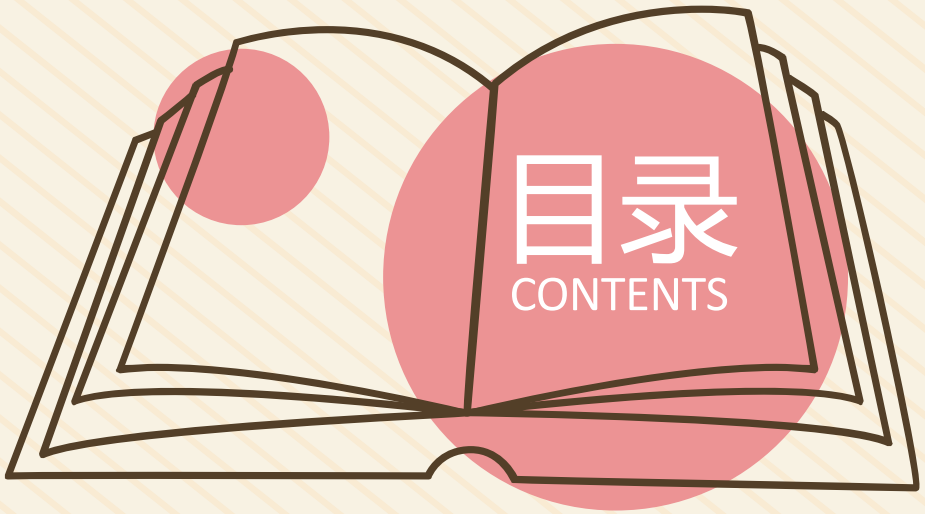
Animal Break

37组，观察类

王衔飞 丁玲 李思涵

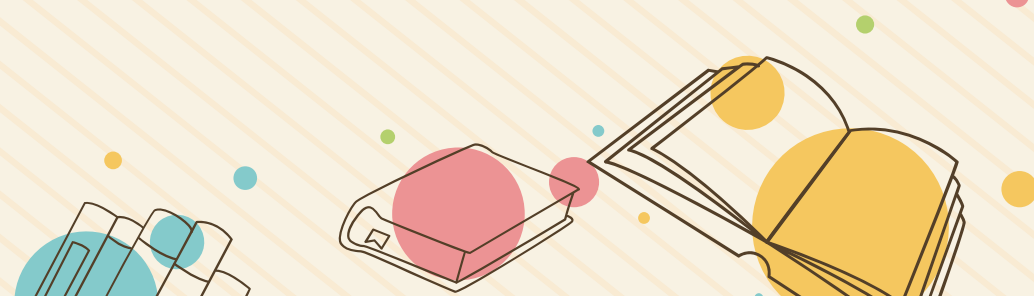


分工情况

王衍飞（组长）：负责总体架构设计及任务分配、开发游戏核心库、编写及研发CGI网页程序。

丁玲：负责桌面应用游戏界面，判断游戏胜负及星级，鼠标点击图标的响应，刷新和暂停功能，代码连接及完善

李思涵：负责桌面应用中的封面设计及封面显示函数、关卡设计及选择函数 ，游戏界面所有素材的寻找，版面的设计，音效的添加分工情况：



技术架构

游戏算法及数据结构

用户界面及交互设计

游戏呈现形式

游戏核心库
Game.h(.cpp)

类中的
接口

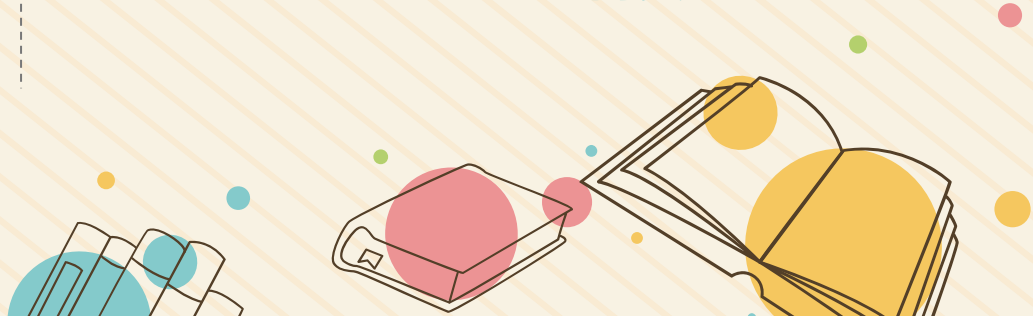
GUI应用程序
（EGE图形库）

CGI应用程序

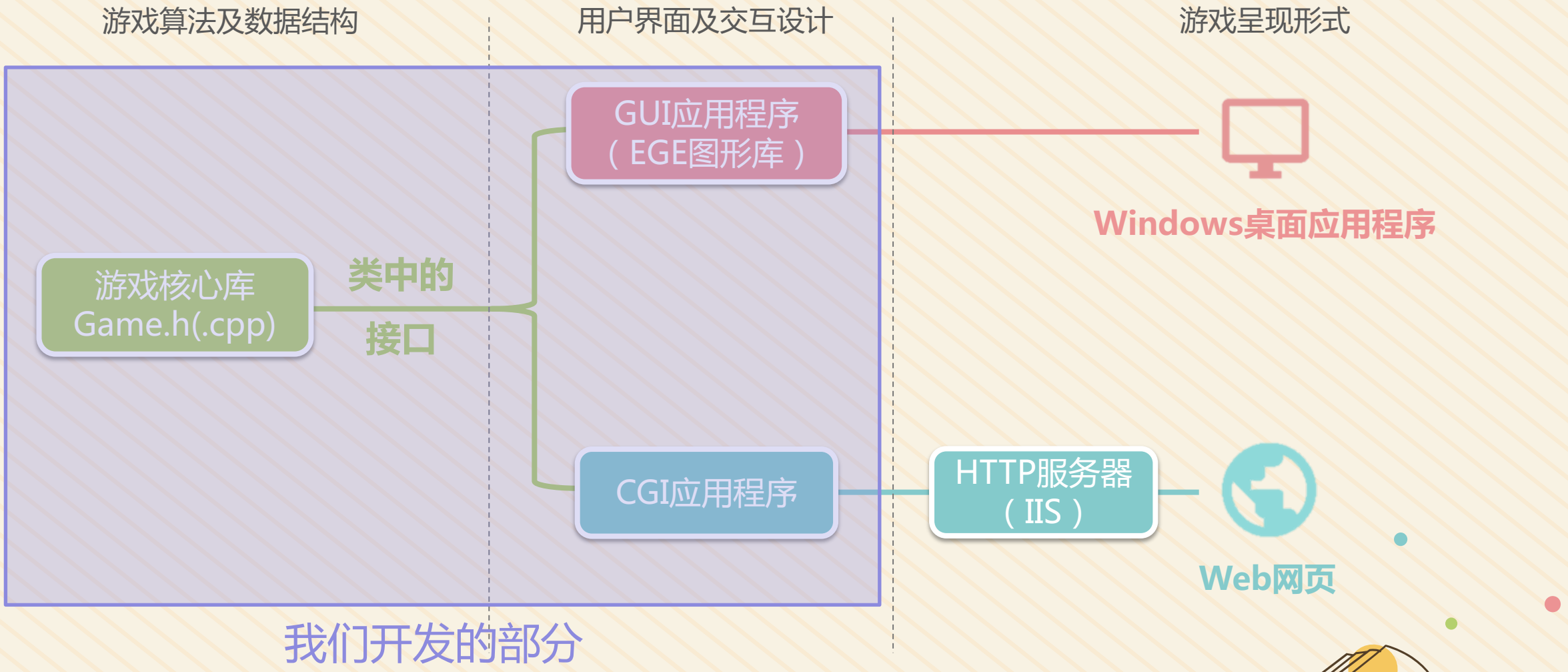
HTTP服务器
（IIS）

Windows桌面应用程序

Web网页



技术架构


游戏算法及数据结构
用户界面及交互设计
游戏呈现形式
GUI应用程序
（EGE图形库）
游戏核心库
Game.h(.cpp)
类中的
接口
CGI应用程序
HTTP服务器
（IIS）
Windows桌面应用程序
Web网页
我们开发的部分

01

游戏介绍及规则

游戏规则：该项目类似于消消乐，即将相邻动物互换位置，使同种动物在一行或一列上达到三个或以上的数量即可消去。在规定的步数内达到一定的分数为通关成功。

游戏功能：游戏总共有五关，前三关增加动物种类、增加目标分数使难度递增，最后两关增加小木块，限制玩家的部分操作。玩家可选择关卡，可在游戏中刷新界面，使重新排序，也可按菜单按钮进行返回关卡、暂停、退出等操作。游戏中还加入了各种音效，使玩家得到更好的体验。

当前剩余步数，少于5步时，以红字警示

刷新按钮：重新排列小动物的顺序

菜单按钮

分数进度条

第一关

第二关

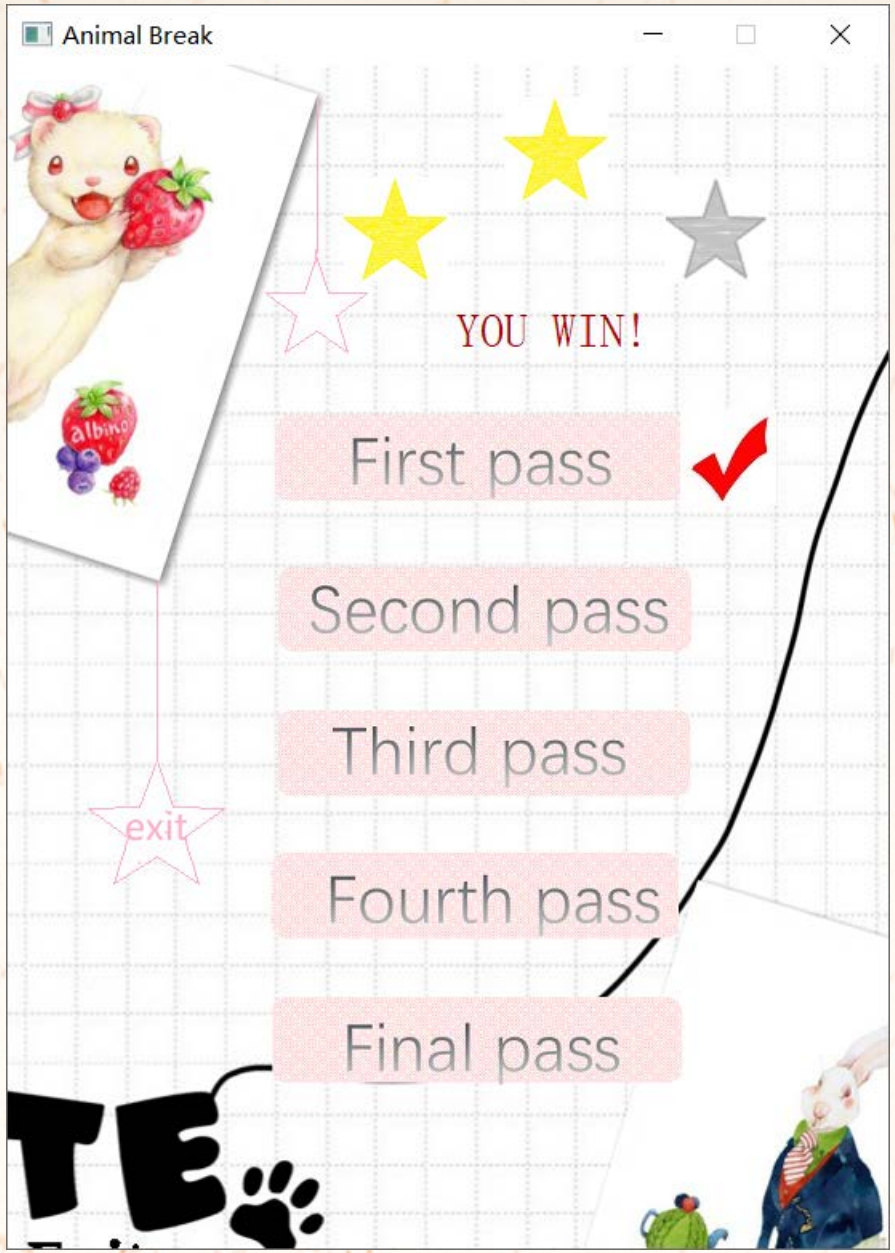
第三关

第一二三关以动物种类的递增来增加难度

第四关

第五关

第四五关加入小木块增加难度：小木块固定，不可与小动物交换



计分规则是2的幂次方，三个相连为2分，四个为4分，以此类推。在规定的步数之内达到预期分数即为胜利，再根据剩余步数评定星级，三星为剩余一半或以上步数，两星为剩余1/3至1/2的步数，一星为剩余少于1/3的步数；反之，则为失败，没有星级。

依次为 等级 目标分数 行列数 图标种类数 要求完成步数

```
animalBreak level1(1, 50, 7, 7, 4, 10);
animalBreak level2(2, 100, 7, 8, 5, 15);
animalBreak level3(3, 125, 8, 8, 6, 20);
animalBreak level4(4, 125, 9, 8, 7, 25);
animalBreak level5(5, 150, 9, 8, 7, 25);
```

02

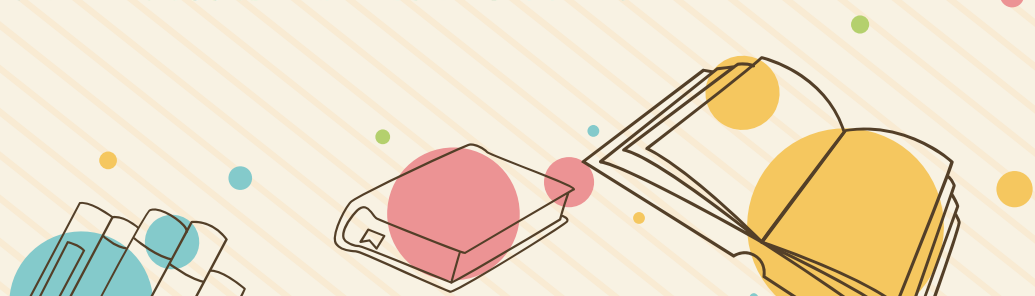
核心代码

游戏核心算法类（EGE版本及CGI版本公用）

Game.h 及 Game.cpp

```
class Game
{
  int row = 0, col = 0; // 游戏行和列
  int score = 0; // 存储游戏得分
  int num; //存储元素的种类个数
  std::vector<std::vector<int>>v;  // 创建存储游戏内容的动态数组

public:
  void handleZero(); // 该函数用来处理消除完之后剩余0的问题
  Game(std::string&,int,int,int zl=4); // 使用字符串构造数组的构造函数，用于打开保存的游戏及网页动态处理
  Game(int row, int col, int zl = 4); // 使用随机数构造数组
  const std::vector<std::vector<int>>& getVector(); // 返回存储数据状态的数组
  bool change(int x1,int y1,int x2,int y2);  // 进行游戏操作，如不能操作返回false
  int& getScore();  // 返回当前游戏得分
  std::string arrayToString(); // 将数据数组转为字符串输出，用于打开保存的游戏及网页动态处理
  std::pair<int,int> getRowCol(); // 返回行列值
};
```



02

核心代码

游戏核心算法类（EGE版本及CGI版本公用）

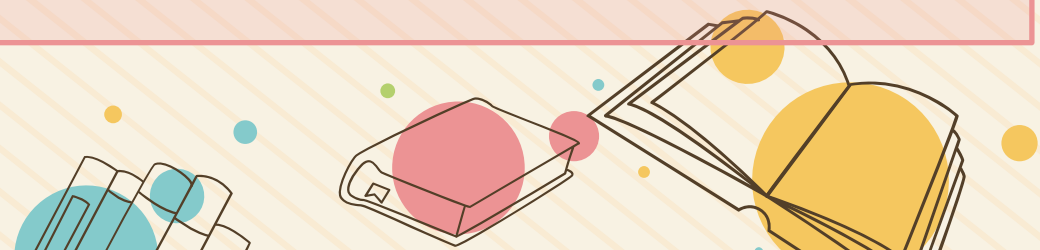
Game.h 及 Game.cpp

```
class Game
{
    int row = 0, col = 0; // 游戏行和列
    int score = 0; // 存储游戏得分
    int num; //存储元素的种类个数
    std::vector<std::vector<int>>v;   // 创建存储游戏内容的动态数组

public:
    void handleZero(); // 该函数用来处理消除完之后剩余0的问题
    Game(std::string&,int,int,int zl=4); // 使用字符串构造数组的构造函数，用于打开保存的游戏及网页动态处理
    Game(int row, int col, int zl = 4); // 使用随机数构造数组
    const std::vector<std::vector<int>>& getVector(); // 返回存储数据状态的数组
    bool change(int x1,int y1,int x2,int y2);  // 进行游戏操作，如不能操作返回false
    int& getScore();  // 返回当前游戏得分
    std::string arrayToString(); // 将数据数组转为字符串输出，用于打开保存的游戏及网页动态处理
    std::pair<int,int> getRowCol(); // 返回行列值
};
```

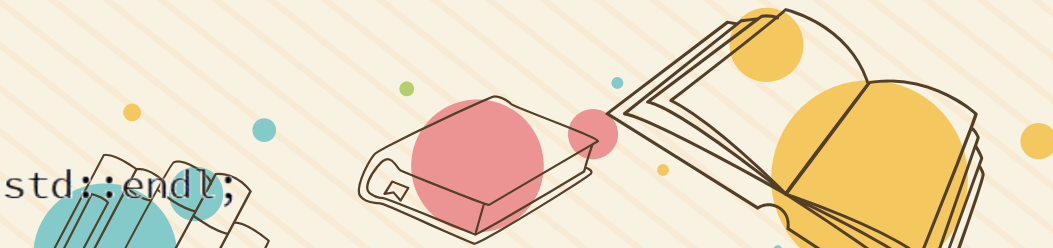
数据部分

函数接口部分

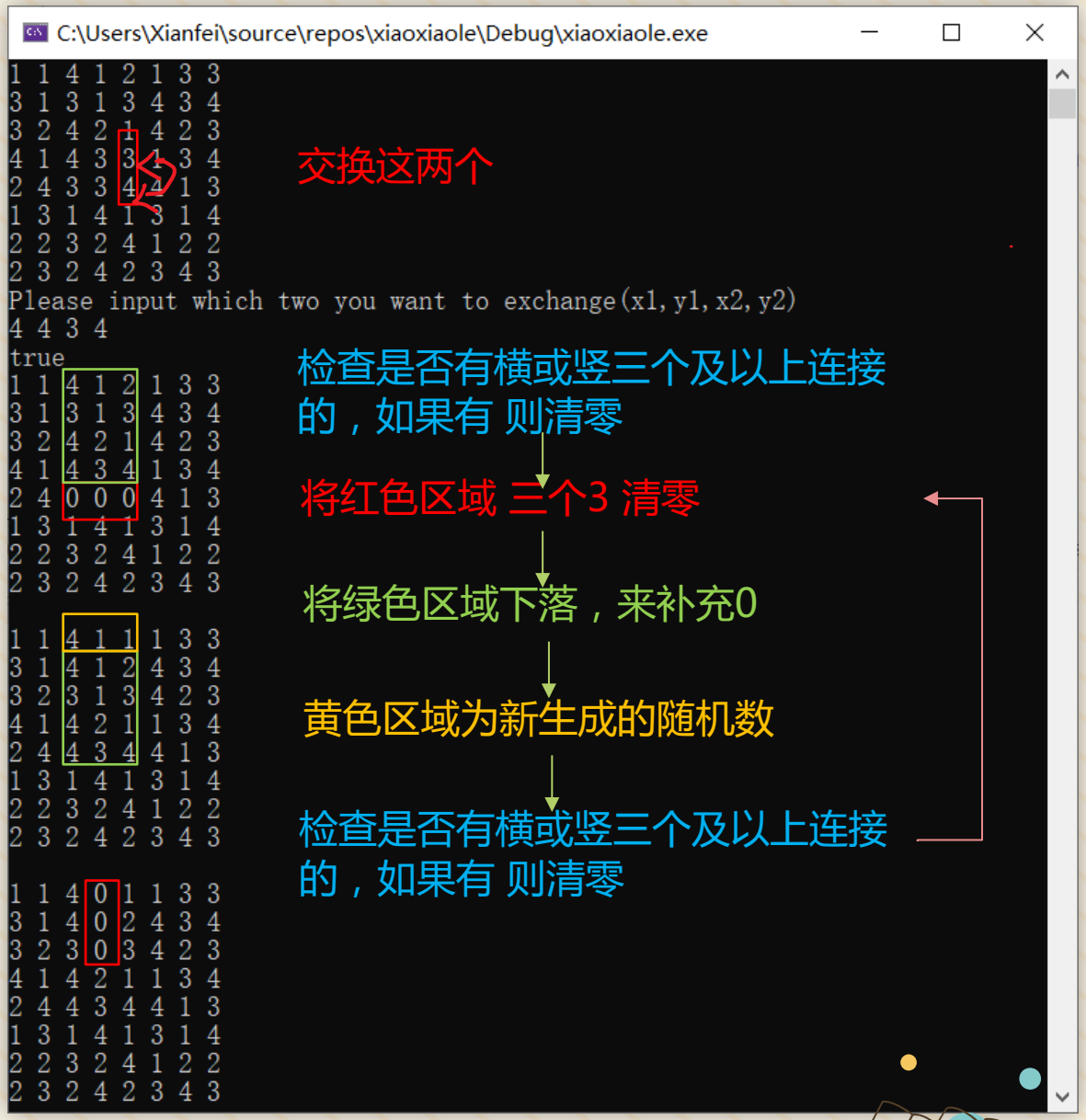


游戏核心算法类——运行示例

```
int main() {
    Game game(8, 8);
    for (;;) {
        for (auto i : game.getVector()) {
            for (auto j : i)std::cout << j << " ";
            std::cout << std::endl;
        }
        std::cout << "Please input which two you want to exchange(x1,y1,x2,y2)" << std::endl;
        int num[4];
        std::cin >> num[0] >> num[1] >> num[2] >> num[3];
        std::cout << std::boolalpha << game.change(num[0], num[1], num[2], num[3]) << std::endl;
        do {
            for (auto i : game.getVector()) {
                for (auto j : i)std::cout << j << " ";
                std::cout << std::endl;
            }
            game.handleZero();
            for (auto i : game.getVector()) {
                for (auto j : i)std::cout << j << " ";
                std::cout << std::endl;
            }
        } while (game.change(0, 0, 0, 0));
        std::cout << "Your Score is: " << game.getScore() << std::endl;
    }
```



游戏核心算法类——运行示例



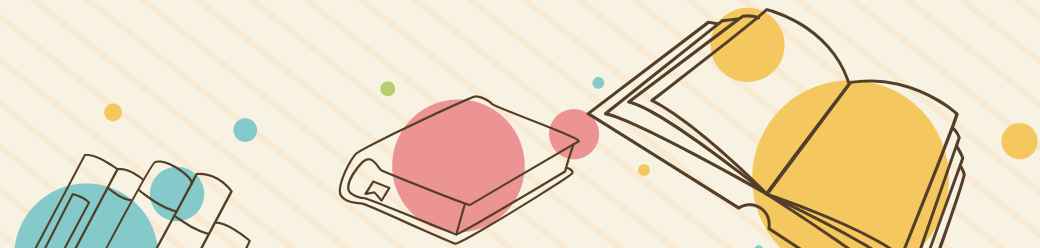
游戏核心算法类——随机数生成器

使用定义于头文件 <random>的mersenne_twister_engine实现了使用Mersenne Twister算法生成高质量的均匀分布随机数。代码如下：

（注意：该代码必须使用C++17或更新的标准编译，否则会报错）

```
std::mt19937 gen(std::random_device{}()); //初始化随机数生成器
for (int i = 0; i < row; i++) {
  v.push_back(std::vector<int>{});
  for (int j = 0; j < col; j++) {
    v[i].push_back(std::uniform_int_distribution{ 1, num }(gen));
     // 用1-n之间的随机数填充每一个元素
  }
}
```

*参考资料：https://en.cppreference.com/w/cpp/numeric/random



03 GUI应用程序版本(使用EGE图形库)

aniamlBreak子类中主要创建了游戏进行中的界面，包含鼠标控制交换，点击时的阴影效果，进度条，剩余步数，暂停界面及刷新功能

feng函数是游戏的开始界面，包含了开始功能和帮助功能

Game基类中主要包含了消除的算法，以及计分规则

guan函数是游戏的关卡选择界面，共有5关和退出游戏功能供玩家选择，5个关卡之间以行列数，动物种类，目标分数和有无木块区分，同时该函数也是游戏胜利界面

lose函数是游戏失败界面

main函数调用feng函数出现开始界面,再调用playGame函数进行游戏

从屏幕上获取图像，保存图像的对象，再屏幕上回值指定图像。

```
PIMAGE feng = newimage();        //获取封面
getimage(feng, "cover.jpg");
putimage(0, 0, feng);
```

鼠标（获取鼠标信息）

```
while (is_run())
{
  msg1 = getmouse();
  if (msg1.is_down()){
     if ((msg1.x > 390) && (msg1.x < 500) && (570 < msg1.y)
        && (msg1.y < 620))   //开始游戏
        {
           return (guanimg(0,0,false));
        }
   }
}
```

添加音效

```
#include "windows.h"
#include “mmsystem.h
#pragma comment(lib,"winmm.lib")
PlaySound(TEXT("sounds\\lose.wav"),
NULL, SND_FILENAME | SND_ASYNC);
```

- 添加进度条

```
void animalBreak::progress(int scores)     //进度条
{
  char s1[5],s2[5];
  sprintf_s(s1, "%d", scores);
  sprintf_s(s2, "%d", allScores);
  setfont(20, 0, "幼圆");
  setcolor(BLACK);
  outtextxy(10, 700, s1);    //显示分数
  outtextxy(500, 700, s2);    //显示总分数
  setfillcolor(0x8ba4e1);
  bar(0, 720, scores*500/allScores, 730);
}
```

- 显示步数

```
void animalBreak::showStep()    //显示步数
{
  char s1[10];
  sprintf_s(s1, "%d", steps);
  setfont(40, 0, "幼圆");
  if (steps > 5)
  setcolor(BLACK);
  else
  setcolor(RED);          //当步数<5时颜色为红色
  outtextxy(70, 25, s1);
}
```

什么是CGI

CGI（通用网关接口）

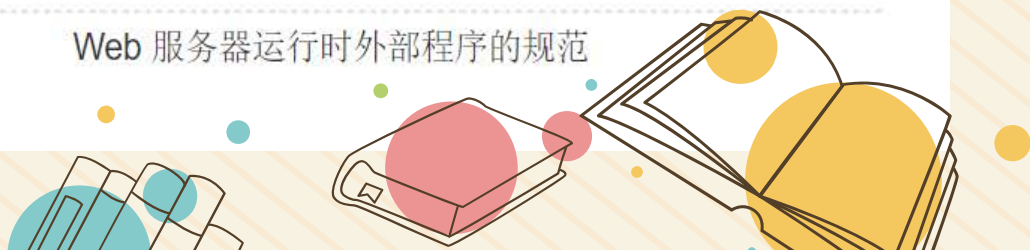
编辑 讨论

同义词 ＣＧＩ一般指CGI（通用网关接口）

本词条由“科普中国”科学百科词条编写与应用工作项目 审核 。

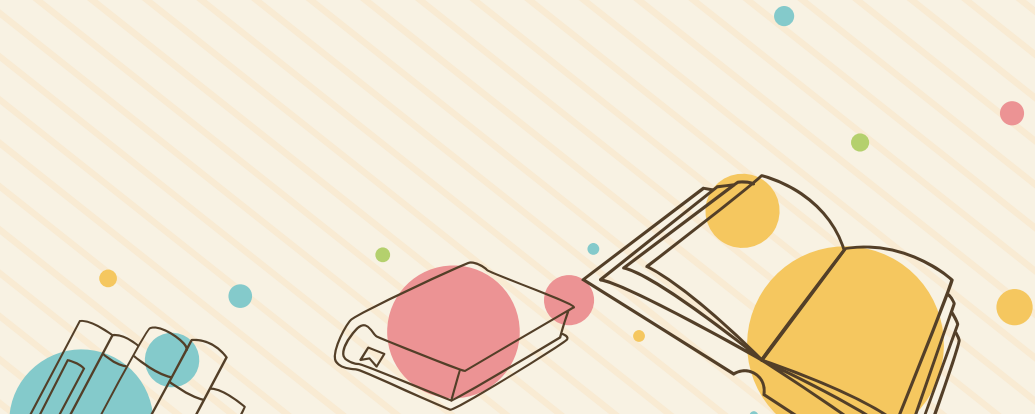
CGI是Web 服务器运行时外部程序的规范，按CGI 编写的程序可以扩展服务器功能。CGI 应用程序能与浏览器进行交互，还可通过数据API与数据库服务器等外部数据源进行通信，从数据库服务器中获取数据。格式化为HTML文档后，发送给浏览器，也可以将从浏览器获得的数据放到数据库中。几乎所有服务器都支持CGI，可用任何语言编写CGI，包括流行的C、C ++、VB 和 Delphi 等。CGI分为标准CGI和间接CGI两种。标准CGI使用命令行参数或环境变量表示服务器的详细请求，服务器与浏览器通信采用标准输入输出方式。间接CGI又称缓冲CGI，在CGI程序和CGI接口之间插入一个缓冲程序，缓冲程序与CGI接口间用标准输入输出进行通信 [1] 。

中文名	公共网关接口	功　能	用来解释处理来自表单的输入信息
外文名	Common Gateway Interface	类　别	标准CGI和间接CGI
学　科	计算机科学	定　义	Web 服务器运行时外部程序的规范



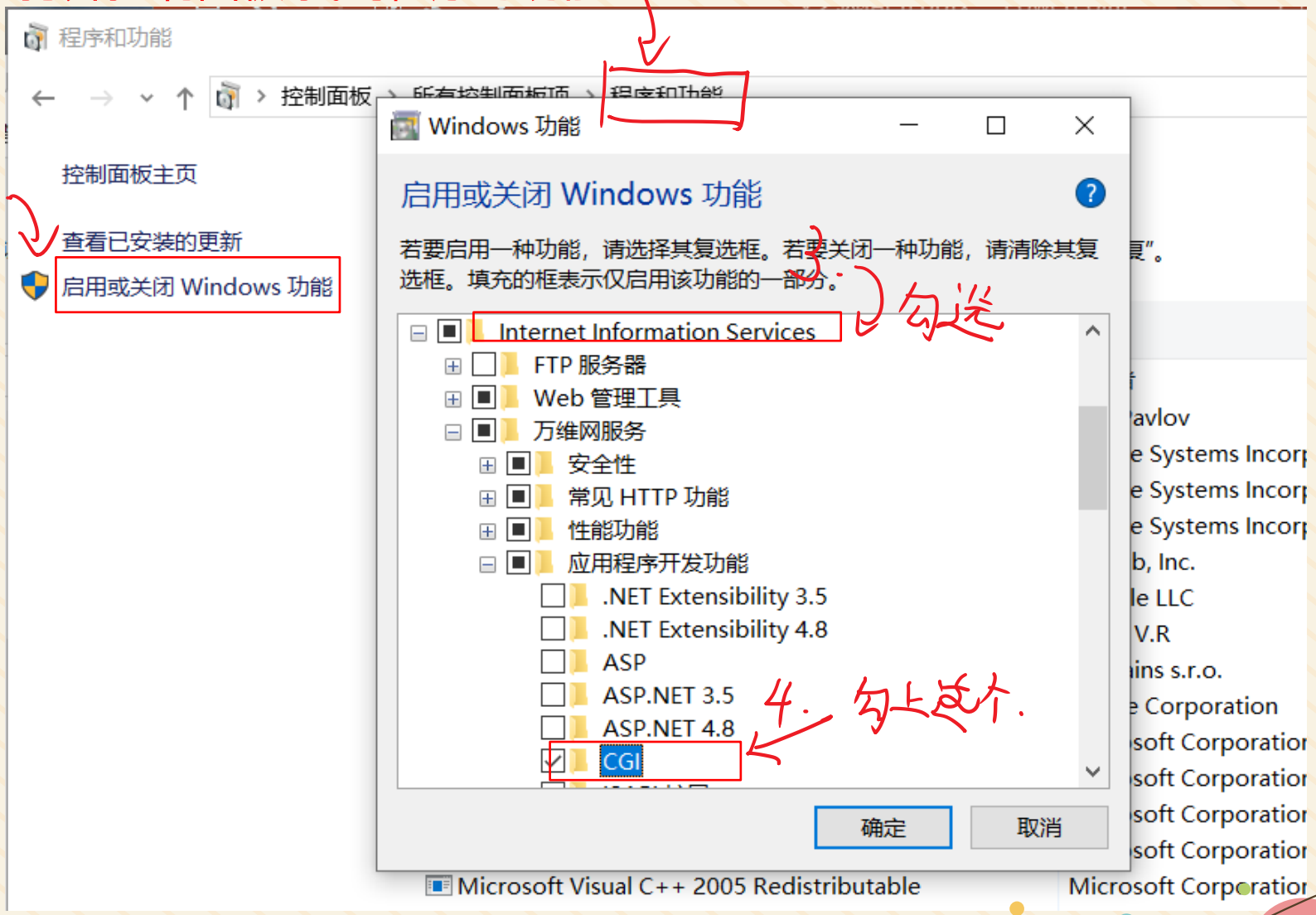
部署环境

该CGI程序为跨平台而优化，可以在不同平台下使用不同HTTP服务器软件进行CGI映射。例如在Windows平台下使用MSVC编译并使用Microsoft IIS作为HTTP服务器，亦可在Linux平台下使用GCC编译并使用Apache httpd作为服务器软件。后文以前者环境进行部署及演示。

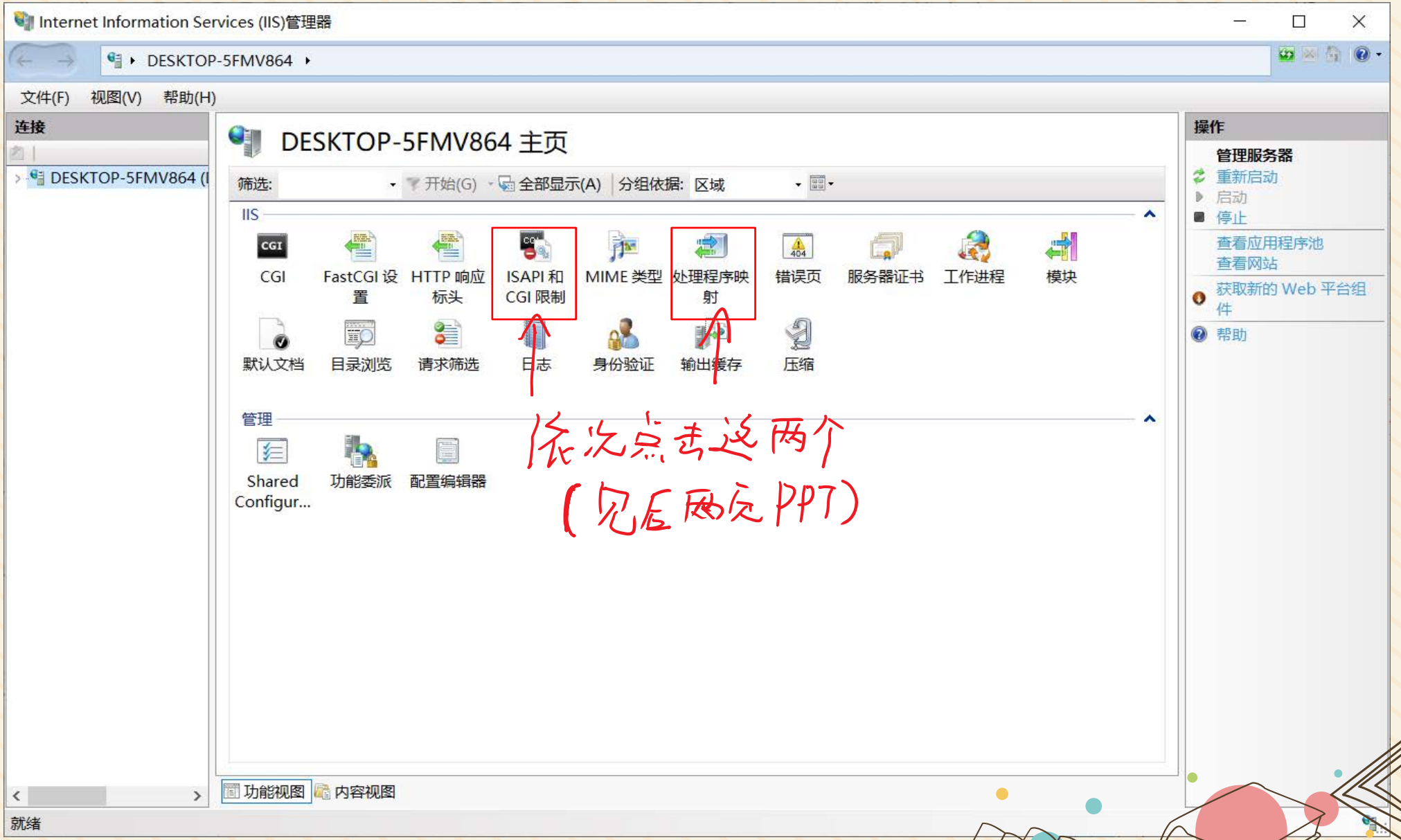


在Windows上配置iis服务器用于映射CGI请求

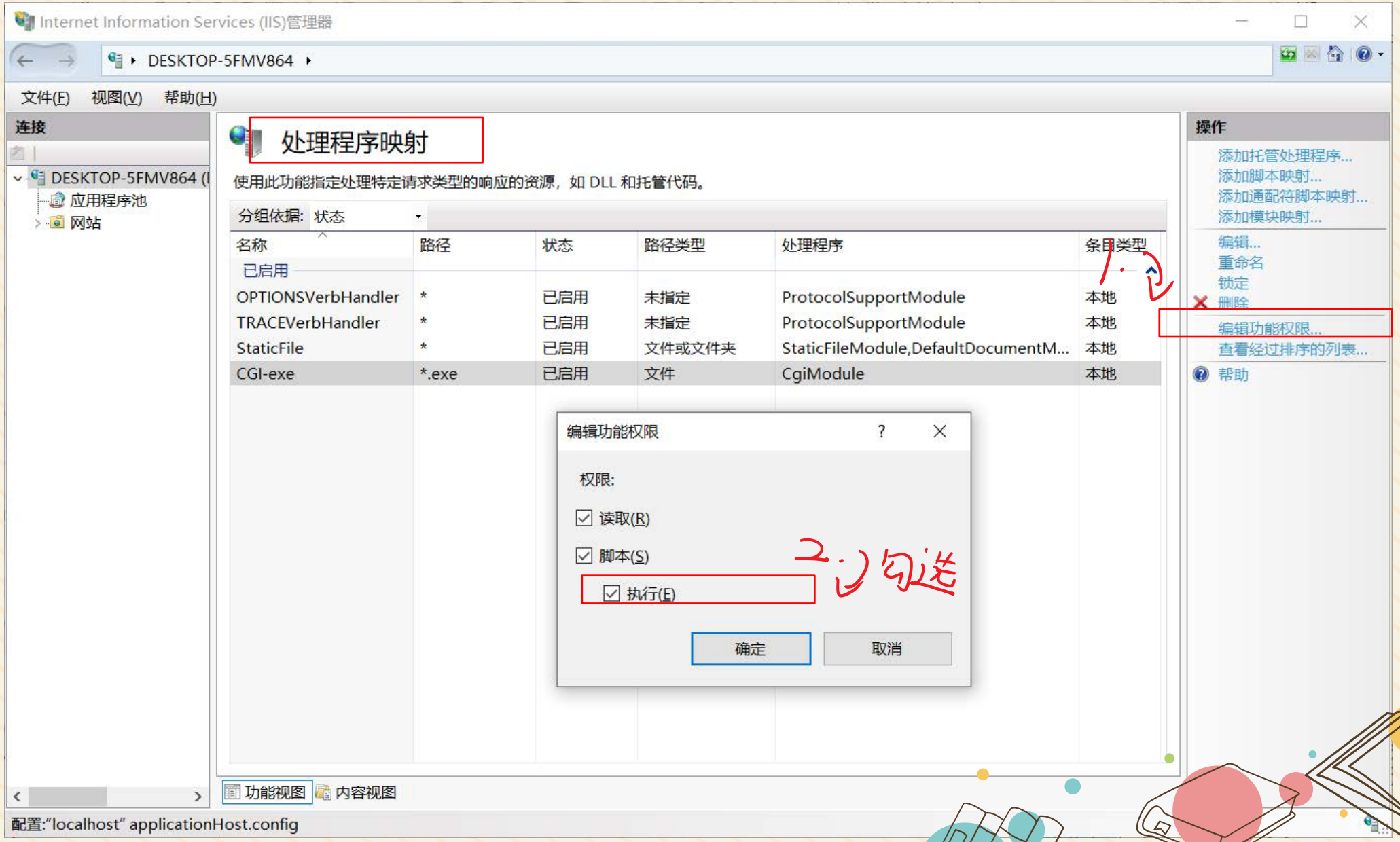
1.打开控制面板 找到程序与功能



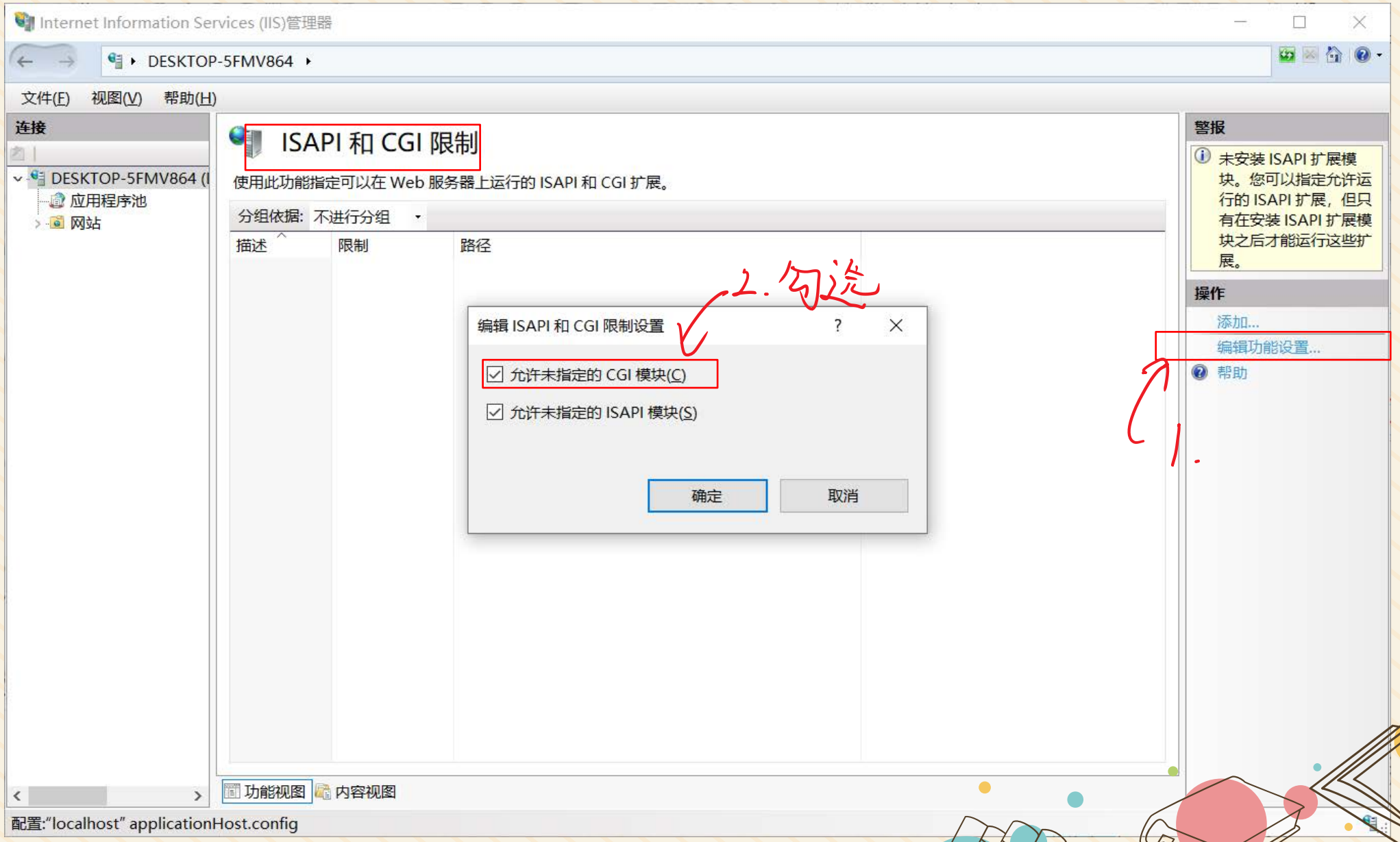
在Windows上配置iis服务器用于映射CGI请求



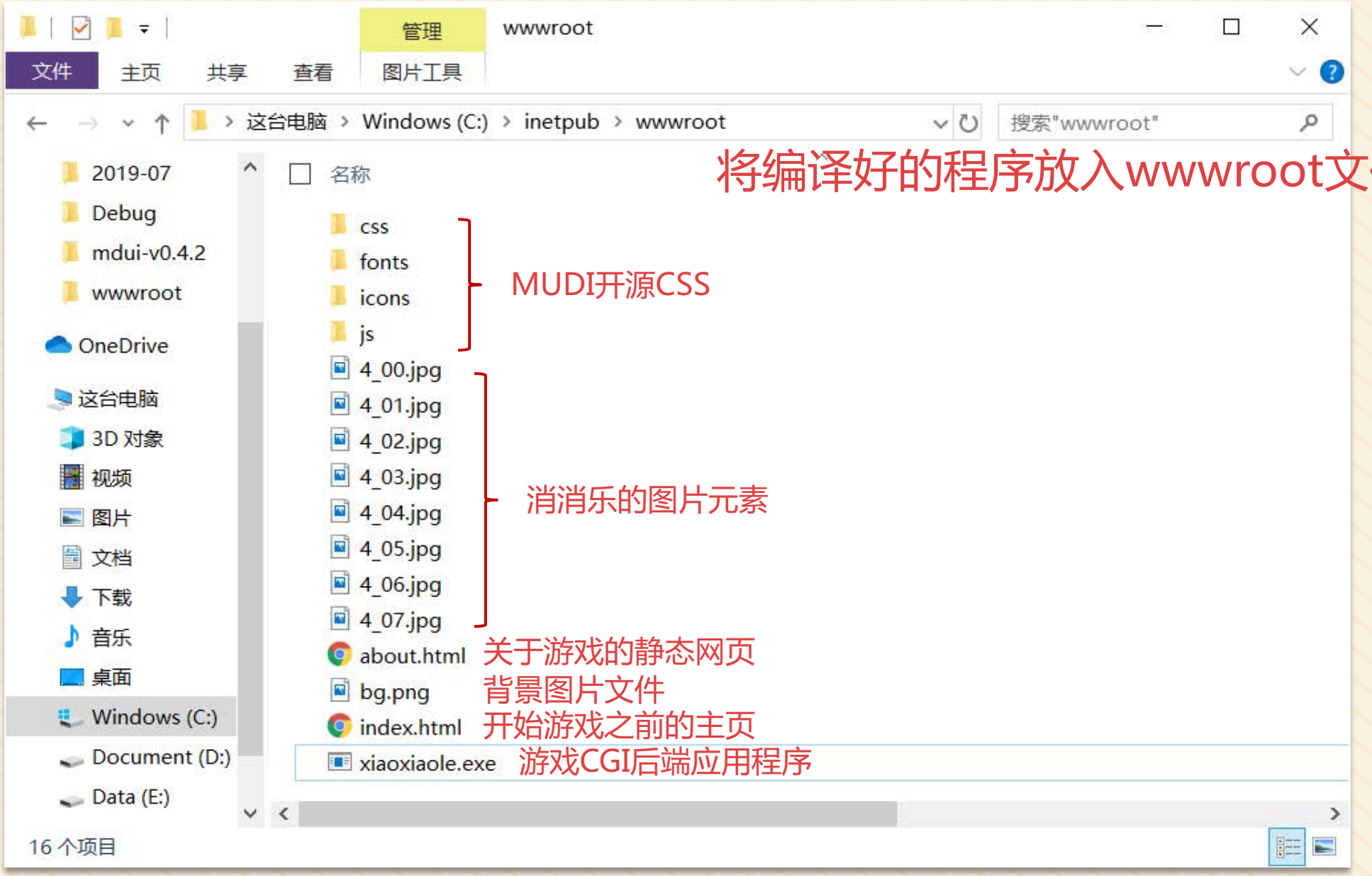
在Windows上配置iis服务器用于映射CGI请求



在Windows上配置iis服务器用于映射CGI请求

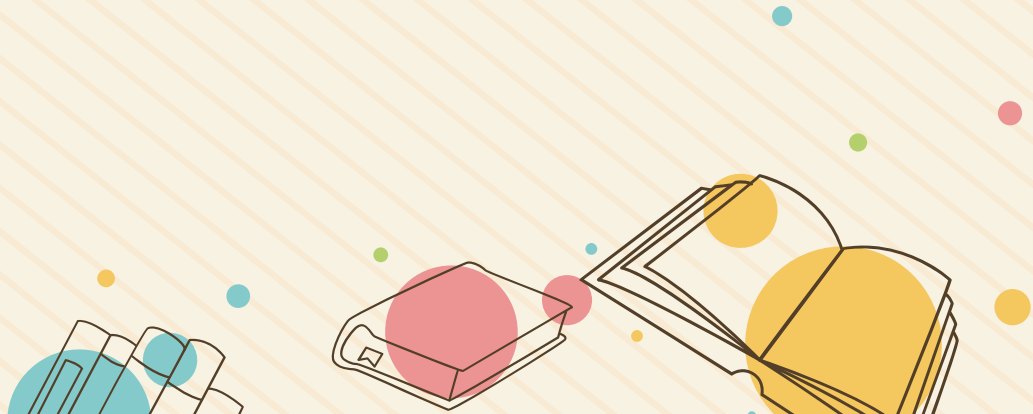


在Windows上配置iis服务器用于映射CGI请求

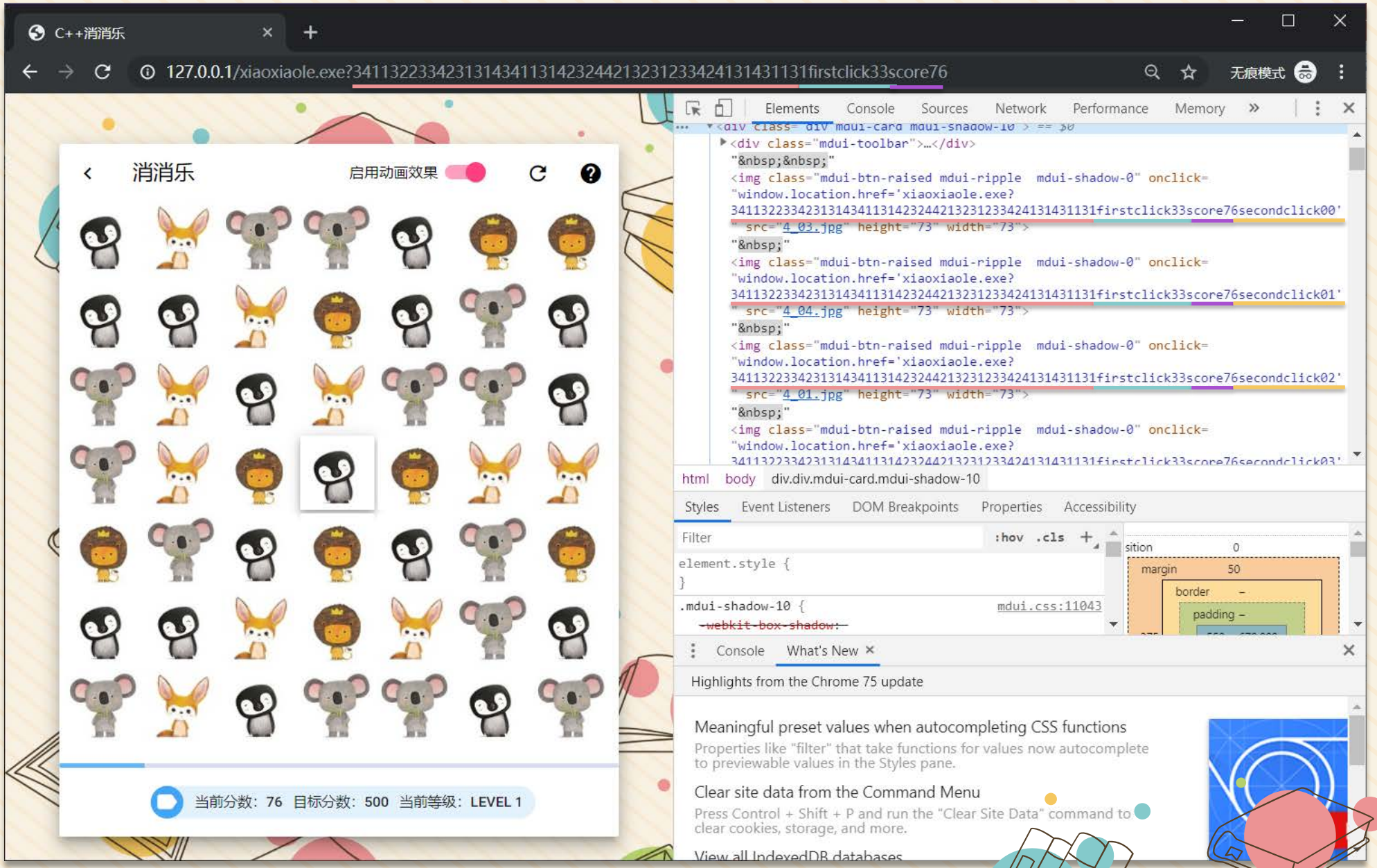


运行机制

使用getenv()函数（在头文件iostream中，系统自带）获取环境变量中的QUERY_STRING变量，即URL中问号后面的部分，当做传入程序的参数，并对其进行分析输出相应的HTML代码。



使用URL中的询问字符串进行传参



游戏布局数组

第一次点击行列

游戏分数

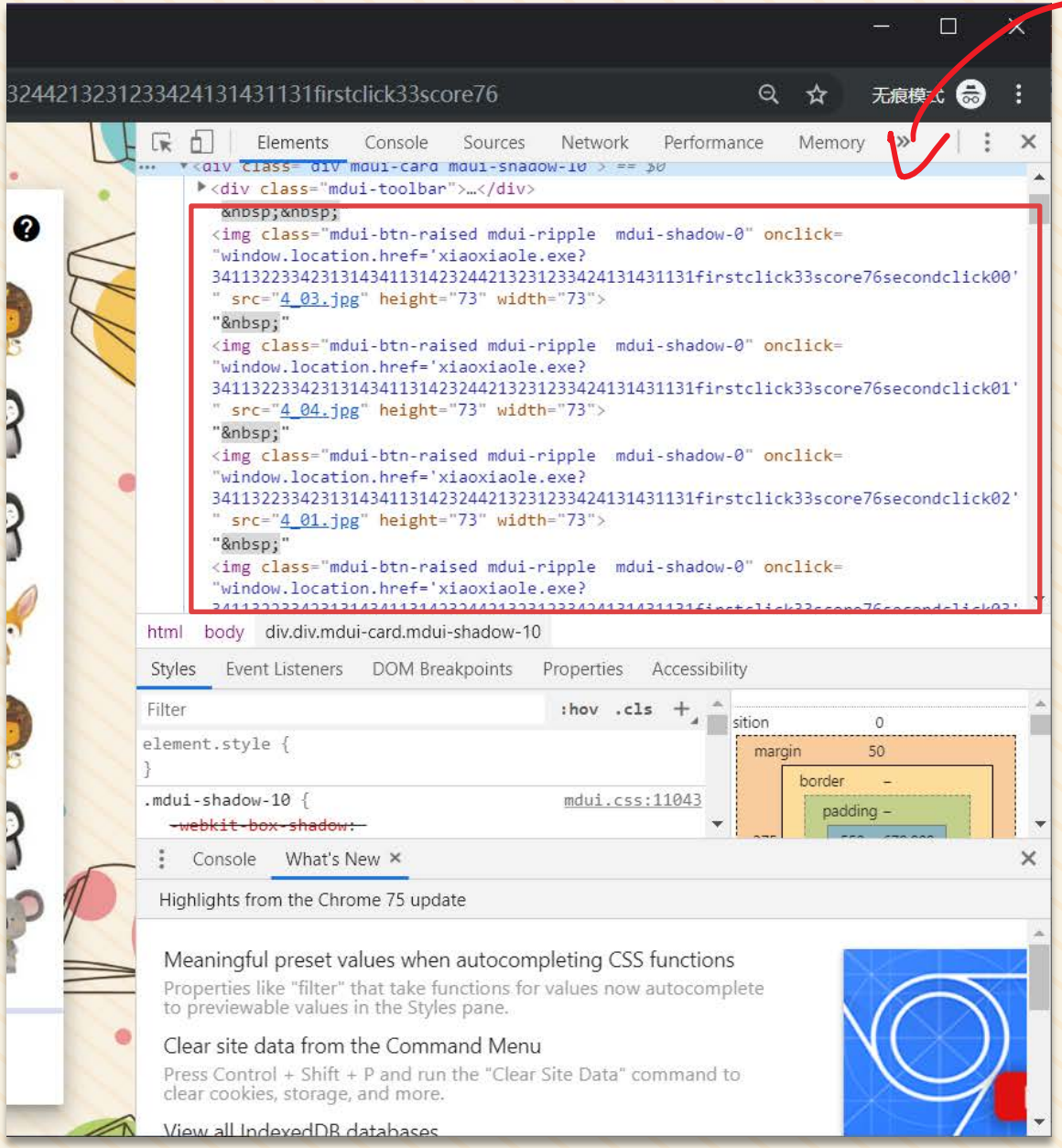
第二次点击行列

处理URL中的询问字符串的后端代码

```
37      }
38      int level = score / 200 + 1;  // 每200分升一级
39      int ROW = 9, COL = 9, NUM = 5,bianchang = 56; //边长测定数据 10 -> 50  8 -> 63  7-> 73  9 -> 56
40      // 等级设定部分
41      switch (level) { ... }
67      // 如果url获取的参数是空的则创建新的游戏
68      if (str.empty()) { ... }
86      // 用于处理动画中消除0的动画
87      else if (str.find("handlezero") != std::string::npos) { ... }
114     // 如果url获取的参数含有第二次点击的信息  并且启用动画
115     else if (str.find("secondclick") != std::string::npos && showAnimation) { ... }
168     // 如果url获取的参数含有第二次点击的信息  并且不启用动画
169     else if (str.find("secondclick") != std::string::npos && !showAnimation) { ... }
203     // 如果url获取的参数只含有第一次点击的信息
204     else if (str.find("firstclick") != std::string::npos) { ... }
231     // 如果这些情况都不符合，则抛出异常
232     else {
233       throw std::exception{ "Illegal Argument Exception" };;
234     }
235   }
236   catch (std::exception& e) {
237     CGI::outException(e.what());
238   }catch (...) {
239     CGI::outException("unkown exception");
240   }
```

121 % 未找到相关问题

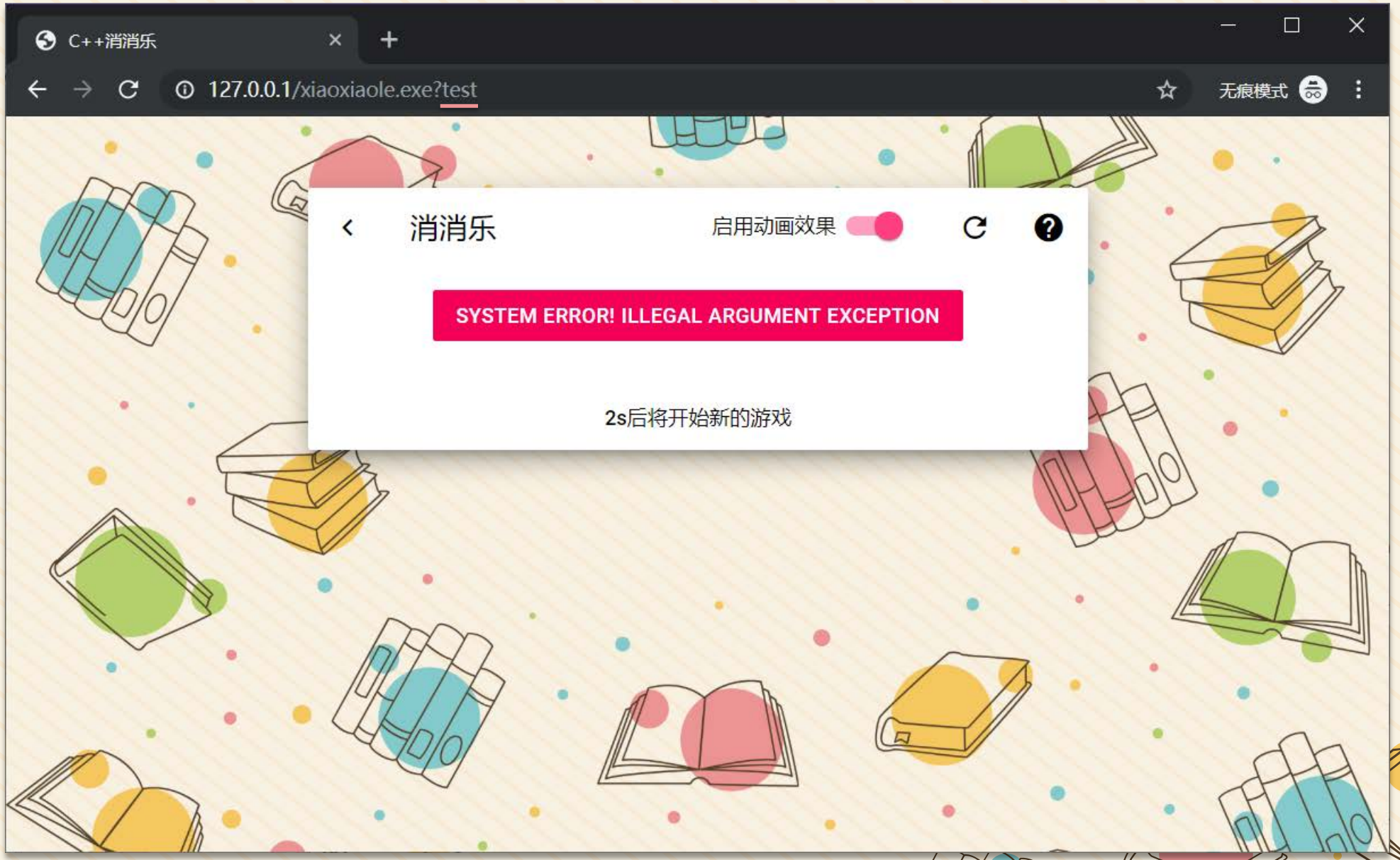
Buffer Graph: C/C++ (TextBuffer)



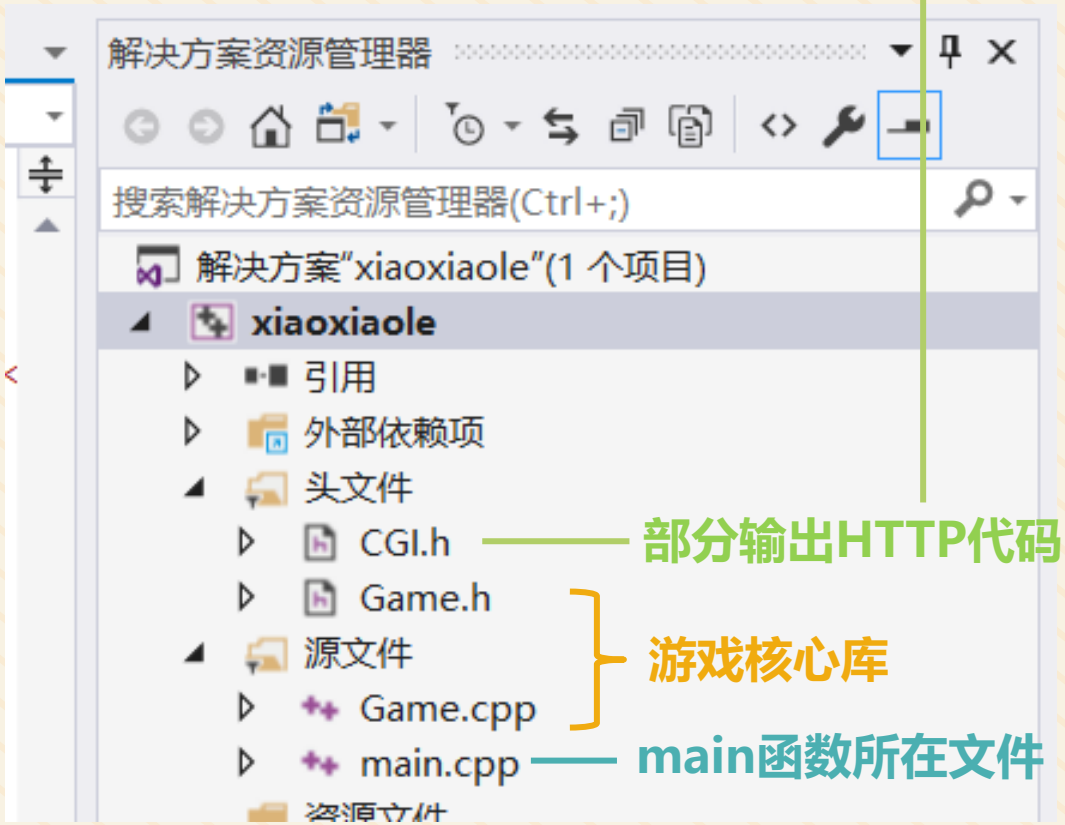
这些HTML文本由如下代码生成

```
for (int i = 0; i < game->getRowCol().first; i++) {
    for (int j = 0; j < game->getRowCol().second; j++) {
        if (game->getVector()[i][j] == 7) {
            std::cout << R"(<img class="mdui-shadow-0" src="4_0)";          // 如果是不可点击的木块
            std::cout << game->getVector()[i][j] << R"(.jpg" height=")" << bianchang << R"(" width=")" << bianchang << R"(" /></a> )";
        }else {                                                            // 如果是可以点击的
            std::cout << R"(<img class="mdui-btn-raised mdui-ripple mdui-shadow-0" onclick="window.location.href=')" << "xiaoxiaole.exe?" << array << "firstclick" << i << j << "score" << game->getScore() << R"(' src="4_0)";
            std::cout << game->getVector()[i][j] << R"(.jpg" height=")" << bianchang << R"(" width=")" << bianchang << R"(" /></a> )";
        }
    }
    std::cout << "<br>  ";
}
```

具备异常处理系统（图中为传入错误的参数造成）



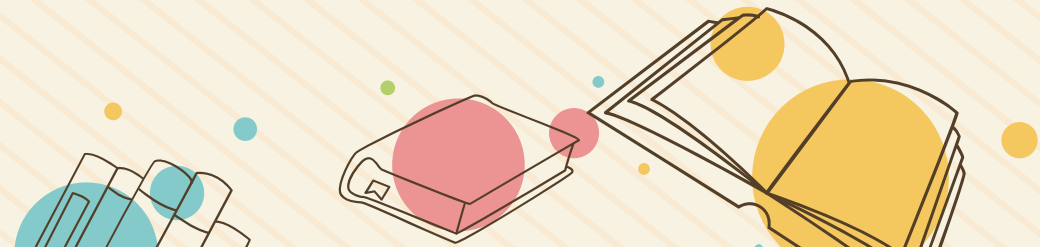
代码布局



ClassDiagram.cd | CGI.h | main.cpp | Game.h | Game.cpp

xiaoxiaole | CGI

```
1   #pragma once
2   #include <string>
3   #include <iostream>
4   namespace CGI
5   {
6
7       // 输出CGI必要的HTTP头文件 及HTML描述文件头部
8       inline void outHead(std::string str,int score){ .
61
62      // 用于输出得分、计算等级、输出进度条
63      inline void outScoreBar(double score){ ... }
131
132     // 输出异常提示
133     inline void outException(const char* except){ ...
153
154     // 输出自动跳转语句
155     inline void outJump(int ms, const char* addr){ ..
171
172 }
173
```



跨平台（图中为在Android手机和iPad上运行的截图）