# 一、源代码说明

本文介绍的是开发框架的字符串操作函数和类。

开发框架函数和类的声明文件是/project/public/\_public.h。

开发框架函数和类的定义文件是/project/public/\_public.h.cpp。

示例程序位于/project/public/demo目录中。

编译规则文件是/project/public/demo/makefile。

# 二、字符串复制

## 1、STRCPY函数

安全的strcpy函数。

函数声明：

char \*STRCPY(char\* dest,const size\_t destlen,const char\* src);

参数说明：

dest：目标字符串，不需要初始化，在STRCPY函数中有初始化代码。

destlen：目标字符串dest占用内存的大小。

src：原字符串。

返回值：目标字符串dest的地址。

**示例（demo1.cpp）**

/\*

\* 程序名：demo1.cpp，此程序演示框架中STRCPY函数的使用。

\*/

#include "\_public.h"

int main()

{

char str[11]; // 字符串str的大小是11字节。

STRCPY(str,sizeof(str),"google"); // 待复制的内容没有超过str可以存放字符串的大小。

printf("str=%s=\n",str); // 出输结果是str=google=

STRCPY(str,sizeof(str),"www.google.com"); // 待复制的内容超过了str可以存放字符串的大小。

printf("str=%s=\n",str); // 出输结果是str=www.google=

}

## 2、STRNCPY函数

安全的strncpy函数。

函数声明：

char \*STRNCPY(char\* dest,const size\_t destlen,const char\* src,size\_t n);

参数说明：

dest：目标字符串，不需要初始化，在STRNCPY函数中有初始化代码。

destlen：目标字符串dest占用内存的大小。

src：原字符串。

n：待复制的字节数。

返回值：目标字符串dest的地址。

**示例（demo2.cpp）**

/\*

\* 程序名：demo2.cpp，此程序演示开发框架中STRNCPY函数的使用。

\*/

#include "\_public.h"

int main()

{

char str[11]; // 字符串str的大小是11字节。

STRNCPY(str,sizeof(str),"google",5); // 待复制的内容没有超过str可以存放字符串的大小。

printf("str=%s=\n",str); // 出输结果是str=googl=

STRNCPY(str,sizeof(str),"www.google.com",8); // 待复制的内容没有超过str可以存放字符串的大小。

printf("str=%s=\n",str); // 出输结果是str=www.goog=

STRNCPY(str,sizeof(str),"www.google.com",17); // 待复制的内容超过了str可以存放字符串的大小。

printf("str=%s=\n",str); // 出输结果是str=www.google=

}

# 三、字符串拼接

## 1、STRCAT函数

安全的strcat函数。

函数声明：

char \*STRCAT(char\* dest,const size\_t destlen,const char\* src);

参数说明：

dest：目标字符串，注意，如果dest从未使过，那么需要至少一次初始化。

destlen：目标字符串dest占用内存的大小。

src：待追加字符串。

返回值：目标字符串dest的地址。

**示例（demo4.cpp）**

/\*

\* 程序名：demo4.cpp，此程序演示开发框架中STRCAT函数的使用。

\*/

#include "../\_public.h"

int main()

{

char str[11]; // 字符串str的大小是11字节。

STRCPY(str,sizeof(str),"www");

STRCAT(str,sizeof(str),".fr"); // str原有的内容加上待追加的内容没有超过str可以存放的大小。

printf("str=%s=\n",str); // 出输结果是str=www.fr=

STRCAT(str,sizeof(str),"eecplus.net"); // str原有的内容加上待追加的内容超过了str可以存放的大小。

printf("str=%s=\n",str); // 出输结果是str=www.freecp=

}

## 2、STRNCAT函数

安全的strncat函数。

函数声明：

char \*STRNCAT(char\* dest,const size\_t destlen,const char\* src,size\_t n)

参数说明：

dest：目标字符串，注意，如果dest从未使过，那么需要至少一次初始化。

destlen：目标字符串dest占用内存的大小。

src：待追加字符串。

n：待追加的字节数。

返回值：目标字符串dest的地址。

**示例（demo5.cpp）**

/\*

\* 程序名：demo5.cpp，此程序演示开发框架中STRNCAT函数的使用。

\*/

#include "../\_public.h"

int main()

{

char str[11]; // 字符串str的大小是11字节。

STRCPY(str,sizeof(str),"www");

STRNCAT(str,sizeof(str),".free",10); // str原有的内容加上待追加的内容没有超过str可以存放的大小。

printf("str=%s=\n",str); // 出输结果是str=www.free=

STRCPY(str,sizeof(str),"www");

STRNCAT(str,sizeof(str),".freecplus.com",6); // str原有的内容加上待追加的内容没有超过str可以存放的大小。

printf("str=%s=\n",str); // 出输结果是str=www.freec=

STRCPY(str,sizeof(str),"www");

STRNCAT(str,sizeof(str),".freecplus.com",10); // str原有的内容加上待追加的内容超过了str可以存放的大小。

printf("str=%s=\n",str); // 出输结果是str=www.freecp=

}

# 四、格式化输出到字符串

## 1、SPRINTF函数

安全的sprintf函数，将可变参数(...)按照fmt描述的格式输出到dest字符串中。

函数声明：

int SPRINTF(char \*dest,const size\_t destlen,const char \*fmt,...);

参数说明：

dest：输出字符串，不需要初始化，在SPRINTF函数中会对它进行初始化。

destlen：输出字符串dest占用内存的大小，如果格式化后的字符串内容的长度大于destlen-1，后面的内容将丢弃。

fmt：格式控制描述。

...：填充到格式控制描述fmt中的参数。

返回值：格式化后的内容的长度，程序员一般不关心返回值。

**示例（demo7.cpp）**

/\*

\* 程序名：demo7.cpp，此程序演示开发框架中SPRINTF函数的使用。

\*/

#include "\_public.h"

int main()

{

char str[21]; // 字符串str的大小是21字节。

SPRINTF(str,sizeof(str),"name:%s,no:%d","messi",10);

printf("str=%s=\n",str); // 出输结果是str=name:messi,no:10=

SPRINTF(str,sizeof(str),"name:%s,no:%d,job:%s","messi",10,"striker");

printf("str=%s=\n",str); // 出输结果是str=name:messi,no:10,job=

}

## 2、SNPRINTF函数

安全的snprintf函数，将可变参数(...)按照fmt描述的格式输出到dest字符串中。

函数声明：

int SNPRINTF(char \*dest,const size\_t destlen,size\_t n,const char \*fmt,...);

参数说明：

dest：输出字符串，不需要初始化，在SNPRINTF函数中会对它进行初始化。

destlen：输出字符串dest占用内存的大小，如果格式化后的字符串内容的长度大于destlen-1，后面的内容将丢弃。

n：把格式化后的字符串截取n-1存放到dest中，如果n>destlen，则取destlen-1。

fmt：格式控制描述。

...：填充到格式控制描述fmt中的参数。

返回值：格式化后的内容的长度，程序员一般不关心返回值。

**示例（demo8.cpp）**

/\*

\* 程序名：demo8.cpp，此程序演示开发框架中SNPRINTF函数的使用。

\*/

#include "\_public.h"

int main()

{

char str[26]; // 字符串str的大小是11字节。

SNPRINTF(str,sizeof(str),5,"messi");

printf("str=%s=\n",str); // 出输结果是str=mess=

SNPRINTF(str,sizeof(str),9,"name:%s,no:%d,job:%s","messi",10,"striker");

printf("str=%s=\n",str); // 出输结果是str=name:mes=

SNPRINTF(str,sizeof(str),30,"name:%s,no:%d,job:%s","messi",10,"striker");

printf("str=%s=\n",str); // 出输结果是str=name:messi,no:10,job:stri=

}

# 五、删除字符串左、右和两边字符

删除字符串左边、右边和两边指定的字符。

函数声明：

void DeleteLChar(char \*str,const char chr); // 删除字符串左边指定的字符。

void DeleteRChar(char \*str,const char chr); // 删除字符串右边指定的字符。

void DeleteLRChar(char \*str,const char chr); // 删除字符串左右两边指定的字符。

参数说明：

str：待处理的字符串。

chr：需要删除的字符。

**示例（demo10.cpp）**

/\*

\* 程序名：demo10.cpp，此程序演示开发框架中删除字符串左、右、两边指定字符的使用。

\*/

#include "../\_public.h"

int main()

{

char str[11]; // 字符串str的大小是11字节。

STRCPY(str,sizeof(str)," 西施 ");

DeleteLChar(str,' '); // 删除str左边的空格

printf("str=%s=\n",str); // 出输结果是str=西施 =

STRCPY(str,sizeof(str)," 西施 ");

DeleteRChar(str,' '); // 删除str右边的空格

printf("str=%s=\n",str); // 出输结果是str= 西施=

STRCPY(str,sizeof(str)," 西施 ");

DeleteLRChar(str,' '); // 删除str两边的空格

printf("str=%s=\n",str); // 出输结果是str=西施=

}

注意，如果要删除字符串中间的字符，可以用开发框架中的UpdateStr函数，后面的章节中会介绍它。

# 六、字符串大小写转换

把字符串中的小写字母转换成大写，忽略不是字母的字符。

函数声明：

void ToUpper(char \*str); // 把字符串中的小写字母转换成大写,参数是C语言风格的char []。

void ToUpper(string &str); // 把字符串中的小写字母转换成大写,参数是C++语言风格的string。

void ToLower(char \*str); // 把字符串中的大写字母转换成小写,参数是C语言风格的char []。

void ToLower(string &str); // 把字符串中的大写字母转换成小写,参数是C++语言风格的string。

参数说明：

str：待转换的字符串，函数重载了char[]和string两种数据类型。

**示例（demo12.cpp）**

/\*

\* 程序名：demo12.cpp，此程序演示开发框架中字符串大小写转换函数的使用。

\*/

#include "../\_public.h"

int main()

{

char str1[31]; // C语言风格的字符串。

STRCPY(str1,sizeof(str1),"12abz45ABz8西施。");

ToUpper(str1); // 把str1中的小写字母转换为大写。

printf("str1=%s=\n",str1); // 出输结果是str1=12ABZ45ABZ8西施。=

STRCPY(str1,sizeof(str1),"12abz45ABz8西施。");

ToLower(str1); // 把str1中的大写字母转换为小写。

printf("str1=%s=\n",str1); // 出输结果是str1=12abz45abz8西施。=

string str2; // C++语言风格的字符串。

str2="12abz45ABz8西施。";

ToUpper(str2); // 把str2中的小写字母转换为大写。

printf("str2=%s=\n",str2.c\_str()); // 出输结果是str2=12ABZ45ABZ8西施。=

str2="12abz45ABz8西施。";

ToLower(str2); // 把str2中的大写字母转换为小写。

printf("str2=%s=\n",str2.c\_str()); // 出输结果是str1=12abz45abz8西施。=

}

# 七、字符串替换

把字符串中的内容替换成其它的内容，在字符串str中，如果存在字符串str1，就替换为字符串str2。

函数声明：

void UpdateStr(char \*str,const char \*str1,const char \*str2,const bool bLoop=true);

参数说明：

str：待处理的字符串。

str1：旧的内容。

str2：新的内容。

bloop：是否循环执行替换。

注意：

1）如果str2比str1要长，替换后str会变长，所以必须保证str有足够的长度，否则内存会溢出。

2）如果str2中包函了str1的内容，且bloop为true，存在逻辑错误，将不执行任何替换。

**示例（demo21.cpp）**

/\*

\* 程序名：demo21.cpp，此程序演示开发框架中字符串替换UpdateStr函数的使用。

\*/

#include "../\_public.h"

int main()

{

char str[301];

STRCPY(str,sizeof(str),"name:messi,no:10,job:striker.");

UpdateStr(str,":","="); // 把冒号替换成等号。

printf("str=%s=\n",str); // 出输结果是str1=name=messi,no=10,job=striker.=

STRCPY(str,sizeof(str),"name:messi,no:10,job:striker.");

UpdateStr(str,"name:",""); // 把"name:"替换成""，相当于删除内容"name:"。

printf("str=%s=\n",str); // 出输结果是str1=messi,no:10,job:striker.=

STRCPY(str,sizeof(str),"messi----10----striker");

UpdateStr(str,"--","-",false); // 把两个"--"替换成一个"-"，bLoop参数为false。

printf("str=%s=\n",str); // 出输结果是str1=messi--10--striker=

STRCPY(str,sizeof(str),"messi----10----striker");

UpdateStr(str,"--","-",true); // 把两个"--"替换成一个"-"，bLoop参数为true。

printf("str=%s=\n",str); // 出输结果是str1=messi-10-striker=

STRCPY(str,sizeof(str),"messi-10-striker");

UpdateStr(str,"-","--",false); // 把一个"-"替换成两个"--"，bLoop参数为false。

printf("str=%s=\n",str); // 出输结果是str1=messi--10--striker=

STRCPY(str,sizeof(str),"messi-10-striker");

// 以下代码把"-"替换成"--"，bloop参数为true，存在逻辑错误，UpdateStr将不执行替换。

UpdateStr(str,"-","--",true); // 把一个"-"替换成两个"--"，bloop参数为true。

printf("str=%s=\n",str); // 出输结果是str1=messi-10-striker=

}

# 八、从字符串中提取数字

从一个字符串中提取出数字、符号和小数点，存放到另一个字符串中。

函数声明：

void PickNumber(const char \*src,char \*dest,const bool bsigned,const bool bdot);

参数说明：

src：源字符串。

dest：目标字符串。

bsigned：是否包括符号（+和-），true-包括；false-不包括。

bdot：是否包括小数点的圆点符号，true-包括；false-不包括。

**示例（demo16.cpp）**

/\*

\* 程序名：demo16.cpp，此程序演示开发框架中PickNumber函数的使用。

\*/

#include "../\_public.h"

int main()

{

char str[26]; // 字符串str的大小是11字节。

STRCPY(str,sizeof(str),"iab+12.3xy");

PickNumber(str,str,false,false);

printf("str=%s=\n",str); // 出输结果是str=123=

STRCPY(str,sizeof(str),"iab+12.3xy");

PickNumber(str,str,true,false);

printf("str=%s=\n",str); // 出输结果是str=+123=

STRCPY(str,sizeof(str),"iab+12.3xy");

PickNumber(str,str,true,true);

printf("str=%s=\n",str); // 出输结果是str=+12.3=

}

# 九、正则表达式

正则表达式，判断一个字符串是否匹配另一个字符串。

函数声明：

bool MatchStr(const string str,const string rules);

参数说明：

str：需要判断的字符串，精确表示的字符串，如文件名"\_public.h.cpp"。

rules：匹配规则表达式，用星号"\*"表示任意字符串，多个字符串之间用半角的逗号分隔，如"\*.h,\*.cpp"。

注意，str参数不支持"?"，rules只支持"\*"，函数在判断str是否匹配rules的时候，会忽略字母的大小写。

**示例（demo18.cpp）**

/\*

\* 程序名：demo18.cpp，此程序演示开发框架正则表达示MatchStr函数的使用。

\*/

#include "../\_public.h"

int main()

{

char filename[301];

STRCPY(filename,sizeof(filename),"\_public.h");

// 以下代码将输出yes。

if (MatchStr(filename,"\*.h,\*.cpp")==true) printf("yes\n");

else printf("no\n");

// 以下代码将输出yes。

if (MatchStr(filename,"\*.H")==true) printf("yes\n");

else printf("no\n");

// 以下代码将输出no。

if (MatchStr(filename,"\*.cpp")==true) printf("yes\n");

else printf("no\n");

}

# 十、字符串的拆分

CCmdStr类用于拆分用分隔符分隔字段内容的字符串。

字符串的格式为：字段内容1+分隔符+字段内容2+分隔符+字段内容3+分隔符+....+字段内容n。

例如："messi,10,striker,30,1.72,68.5,Barcelona"，这是足球运动员梅西的资料，包括姓名、球衣号码、场上位置、年龄、身高、体重和效力的俱乐部，字段之间用半角的逗号分隔开。

CCmdStr类的声明：

// CCmdStr类用于拆分用分隔符分隔字段的字符串。

// 字符串的格式为：字段内容1+分隔符+字段内容2+分隔符+字段内容3+分隔符+...+字段内容n。

// 例如："messi,10,striker,30,1.72,68.5,Barcelona"，这是足球运动员梅西的资料，包括姓名、

// 球衣号码、场上位置、年龄、身高、体重和效力的俱乐部，字段之间用半角的逗号分隔开。

class CCmdStr

{

public:

vector<string> m\_vCmdStr; // 存放拆分后的字段内容。

CCmdStr();

// 把字符串拆分到m\_vCmdStr容器中。

// buffer：待拆分的字符串。

// sepstr：buffer字符串中字段内容的分隔符，注意，分隔符是字符串，如","、" "、"|"、"~!~"。

// bdelspace：是否删除拆分后的字段内容前后的空格，true-删除；false-不删除，缺省删除。

void SplitToCmd(const string buffer,const char \*sepstr,const bool bdelspace=true);

// 获取拆分后字段的个数，即m\_vCmdStr容器的大小。

int CmdCount();

// 从m\_vCmdStr容器获取字段内容。

// inum：字段的顺序号，类似数组，从0开始。

// value：传入变量的地址，用于存放字段内容。

// 返回值：true-获取成功；false-获取失败。

bool GetValue(const int inum,char \*value,const int ilen=0); // 字符串，ilen缺省值为0。

bool GetValue(const int inum,int \*value); // int整数。

bool GetValue(const int inum,unsigned int \*value); // unsigned int整数。

bool GetValue(const int inum,long \*value); // long整数。

bool GetValue(const int inum,unsigned long \*value); // unsigned long整数。

bool GetValue(const int inum,double \*value); // 双精度double。

bool GetValue(const int inum,bool \*value); // bool型。

~CCmdStr();

};

**示例（demo20.cpp）**

/\*

\* 程序名：demo20.cpp，此程序演示开发框架拆分字符串的类CCmdStr的使用。

\*/

#include "../\_public.h"

// 用于存放足球运动员资料的结构体。

struct st\_player

{

char name[51]; // 姓名

char no[6]; // 球衣号码

bool striker; // 场上位置是否是前锋，true-是；false-不是。

int age; // 年龄

double weight; // 体重，kg。

long sal; // 年薪，欧元。

char club[51]; // 效力的俱乐部

}stplayer;

int main()

{

memset(&stplayer,0,sizeof(struct st\_player));

char buffer[301];

STRCPY(buffer,sizeof(buffer),"messi,10,true,30,68.5,2100000,Barcelona");

CCmdStr CmdStr;

CmdStr.SplitToCmd(buffer,","); // 拆分buffer

CmdStr.GetValue(0, stplayer.name,50); // 获取姓名

CmdStr.GetValue(1, stplayer.no,5); // 获取球衣号码

CmdStr.GetValue(2,&stplayer.striker); // 场上位置

CmdStr.GetValue(3,&stplayer.age); // 获取年龄

CmdStr.GetValue(4,&stplayer.weight); // 获取体重

CmdStr.GetValue(5,&stplayer.sal); // 获取年薪，欧元。

CmdStr.GetValue(6, stplayer.club,50); // 获取效力的俱乐部

printf("name=%s,no=%s,striker=%d,age=%d,weight=%.1f,sal=%ld,club=%s\n",\

stplayer.name,stplayer.no,stplayer.striker,stplayer.age,\

stplayer.weight,stplayer.sal,stplayer.club);

// 输出结果:name=messi,no=10,striker=1,age=30,weight=68.5,sal=21000000,club=Barcelona

}