


prologue blog

map library tag guest



 레틴
(vosej_v)

그림쟁이 프로그래머
프로필 쪽지

... C / C++ 8개의 글

목록닫기

글 제목

작성일

[C++ 라이브러리] String 클래스 함수 정리 (12)

2013. 7. 24.

MSDN 표준 C++ 라이브러리 링크 (1)

2013. 7. 16.

why ? GRIPPER, NOGRIPPER

2013. 5. 22.

virtual 멤버 함수, 순수가상함수

2013. 5. 21.

CR(carriage return) LF(line feed), 개행문자 \r\n

2013. 5. 17.

5줄 보기 ▾

1 2

[C++라이브러리]String클래스 함수 정리

... C / C++ / 프로그래밍 : study

2013. 7. 24. 20:00

http://blog.naver.com/vosej_v/50176084445 복사

번역하기

전용뷰어 보기

category

전체 보기 (115)

HAPPY :) 유능한 레틴
(4)

Hello, Mr.blog

하루의기록 : daily

레틴의공방 : make

... 핸드메이드 (14)

... 일러스트 (5)

... 자수 (22)

... 초상화 (1)

망상과상상 : daydream

꿈꾸는선율 : music

지식보관함 : ITC, etc

나의도서관 : book

프로그래밍 : study

... C / C++ (8)

... JAVA (4)

... Android (8)

... MFC/API (12)

... LINUX (2)

... MCU (2)

... Network (1)

... Database (3)

... Embedded (3)

... Intel edison (1
3)

string class

Header : <string>

.assign : 문자열을 할당

(문자열) : 문자열을 할당한다.

(개수, 문자) : 문자를 개수만큼 할당한다

(문자열, 시작위치, 개수) : 매개변수 문자열의 시작위치부터 개수만큼을 호출한 문자열에 할당

```
string s1, s2, s3 ;
s1.assign( "ABCDEFGFG" ) ; // s1 = "ABCDEFGFG"
s2.assign( 3, 'a' ) ; // s2 = "aaa"
s3.assign( s, 2, 4 ) ; // s3 = "CDEF" ( 문자열 s의 2부터 4개를 복사하여 할당 )
```

.append : +연산자의 역할처럼 문자열을 끝에 더한다.

(문자열) : 문자열을 더한다

(개수, 문자) : 문자를 개수만큼 끝에 더한다

(문자열, 시작위치, 개수) :

```
string s, s2 ;
s.append( "ABCDEFGF" ) ; // s = "ABCDEFGF"
s.append( 3, 'x' ) ; // s = "ABCDEFGFxxx"
s2.append( s, 2, 4 ) ; // s2 = "CDEF"
s2 += "x" ; // s2 = "CDEFx"
```

레틴의공방

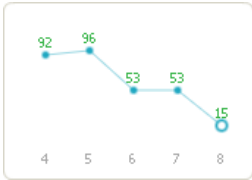
이 블로그에서 검색



5



1/3 전체보기



15
184,910



레틴
이웃 커넥트

내가 추가한 나를 추가한



이스완



지유짱맘



GlassM



변화



kk9722는



DD1202

1/ 15

레틴님 이웃의 새글보기 ▶

s.clear() ;

.compare : 문자열을 비교 (사전순으로 비교)

s.compare(s2) ; // s == s2이면 0, s<s2이면 음수, s>s2이면 양수를 반환

.empty : 문자열이 비었는지 확인

s.empty() ;

.erase : 문자열을 지운다

(시작위치, 개수) : 시작위치부터 개수만큼의 문자를 지운다.

```
string s = "ABCDEF" ;  
s.erase( 0, 3 ) ; // s = "DEF" ( 인덱스 0부터 3개의 문자를 지운다 )
```

.find : 특정 문자열을 찾고, 그 시작위치를 반환

(문자) : 인덱스 0부터 해당 문자를 찾고, 시작위치를 반환한다.

(문자열) : 인덱스 0부터 해당 문자열을 찾고, 그 시작위치를 반환한다.

(문자열, 시작위치) : 시작위치부터 문자열을 찾고, 시작위치를 반환한다.

```
string s1 = "abcd" ;  
string s2 = "b" ;
```

```
int location = s1.find( s2 ) ;  
location = s1.find( s2, x ) ;  
location = s1.find_first_of( s2, x ) ;  
location = s1.find_first_not_of( s2, x ) ;  
location = s1.find_last_of  
s1.find_last_not_of
```

.replace : 문자열을 대체

(시작위치, 개수, 문자열) : 호출한 문자열의 시작위치부터 개수만큼의 문자를 매개변수 문자열로 대체한다.

```
string s = "abc_def" ;  
s.replace( 4, 3, "zzz" ) ; // s = "abc_zzz" ( 인덱스 4부터 3개의 문자를 "zzz"로 대체 )
```

.insert : 문자열을 지정한 위치에 삽입

(시작위치, 문자열) : 시작위치에 문자열을 삽입한다.

```
string s = "ABCDEF" ;  
s.insert( 2, "xx" ) ; // s = "ABxxCDEF"
```

.pop_back : 문자열에서 가장 뒤의 문자 하나를 뺀다.

.push_back : 문자열의 가장 뒤에 문자 하나를 추가한다.

```
string s = "ABCDEF" ;  
s.pop_back() ; // s = "ABCDE"  
s.push_back( 'x' ) ; // s = "ABCDEx"
```

.resize : 문자열의 크기를 재설정

s

.size, .length : 문자열의 크기를 반환

.max_size : 문자열이 최대로 가질수 있는 길이를 반환한다.

```
string s = "ABCDEF" ;  
int size = s.size() ; // size = 6 ( 실제 사용되고 있는 크기 )  
int length = s.length() ; // length = 6 ( 문자열의 길이 )
```

.capacity : 할당된 메모리의 크기를 반환 (reallocation 없이 사용할 수 있는 문자수를 반환)

```
string s = "ABCDEF" ;  
int capacity = s.capacity() ;    // size = 6, capacity = 15
```

★ **capacity가 size보다 클때, 속도는 더 빠르다 !**

= capacity가 size보다 크면, 기존 공간에 문자를 추가하면 되지만
그 반대의 경우에는 문자를 추가하기 위해 새로운 메모리를 할당해야 하기 때문이다.

.reserve : reallocation을 피하기 위해, 메모리의 최소용량을 지정
(크기) : 크기만큼의 여유 메모리를 할당한다.
** 이때, 매개변수의 크기는 현재 capacity보다 크지않으면 의미도 효과도 없다.

```
string s = "ABCDEF" ;    // size = 6, capacity = 15  
s.reserve( 100 ) ;    // size = 6, capacity = 111  
s.reserve( 1000 ) ;    // size = 6, capacity = 1007
```

.substr : 문자열의 일부분을 문자열로 반환
(시작위치) : 시작위치부터 끝까지의 문자들을 문자열로 반환
(시작위치, 개수) : 시작위치부터 개수만큼의 문자를 문자열로 반환

```
string s = "ABCDEF" ;  
string s2 = s.substr( 4 ) ;    // s2 = "EF" ( 인덱스 4부터 끝까지의 문자열을 반환 )  
string s3 = s.substr( 1, 3 ) ; // s3 = "BCD" ( 인덱스 1부터 3까지의 문자열을 반환 )
```

.swap : 문자열을 서로 바꾼다

```
string a = "ABCD" ;  
string b = "WXYZ" ;  
a.swap( b ) ;    // a = "WXYZ", b = "ABCD"  
b.swap( a ) ;    // a = "ABCD", b = "WXYZ"
```

.at : 문자열에서 특정위치의 문자를 액세스

```
string s = "ABCDEF" ;  
char c = s.at(3) ;    // c = 'D'
```

.c_str : string 문자열을 char* 형으로 바꾸어 반환한다.

```
string s = "ABCDEF" ;  
int length = strlen( s.c_str() ) ;
```

.date
.copy
.back
.front
.begin
.end

#IT·컴퓨터 #string



♡ 공감 8 | ∨

💬 댓글 12 | ∨

인쇄

이 블로그 ... C / C++ 카테고리 글

전체글 보기

[C++ 라이브러리] String 클래스 함수 정리 (12)

2013. 7. 24.

MSDN 표준 C++ 라이브러리 링크 (1)

2013. 7. 16.

virtual 멤버 함수, 순수가상함수

2013. 5. 21.

CR(carriage return) LF(line feed), 개행문자 \r\n

2013. 5. 17.

[< 이전](#) [다음 >](#)

[▲ TOP](#)