

```
G++ 2.91.57, cygnus\cygwin-b20\include\g++\stl_hash_fun.h 完整列表
/*
 * Copyright (c) 1996
 * Silicon Graphics Computer Systems, Inc.
 *
 * Permission to use, copy, modify, distribute and sell this software
 * and its documentation for any purpose is hereby granted without fee,
 * provided that the above copyright notice appear in all copies and
 * that both that copyright notice and this permission notice appear
 * in supporting documentation. Silicon Graphics makes no
 * representations about the suitability of this software for any
 * purpose. It is provided "as is" without express or implied warranty.
 *
 *
 * Copyright (c) 1994
 * Hewlett-Packard Company
 *
 * Permission to use, copy, modify, distribute and sell this software
 * and its documentation for any purpose is hereby granted without fee,
 * provided that the above copyright notice appear in all copies and
 * that both that copyright notice and this permission notice appear
 * in supporting documentation. Hewlett-Packard Company makes no
 * representations about the suitability of this software for any
 * purpose. It is provided "as is" without express or implied warranty.
 *
 */

/* NOTE: This is an internal header file, included by other STL headers.
 * You should not attempt to use it directly.
 */

#ifndef __SGI_STL_HASH_FUN_H
#define __SGI_STL_HASH_FUN_H

#include <stddef.h>

__STL_BEGIN_NAMESPACE

template <class Key> struct hash { };

inline size_t __stl_hash_string(const char* s)
{
    unsigned long h = 0;
    for ( ; *s; ++s)
        h = 5*h + *s;

    return size_t(h);
}
```

```

__STL_TEMPLATE_NULL struct hash<char*>
{
    size_t operator()(const char* s) const { return __stl_hash_string(s); }
};

__STL_TEMPLATE_NULL struct hash<const char*>
{
    size_t operator()(const char* s) const { return __stl_hash_string(s); }
};

__STL_TEMPLATE_NULL struct hash<char> {
    size_t operator()(char x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<unsigned char> {
    size_t operator()(unsigned char x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<signed char> {
    size_t operator()(unsigned char x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<short> {
    size_t operator()(short x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<unsigned short> {
    size_t operator()(unsigned short x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<int> {
    size_t operator()(int x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<unsigned int> {
    size_t operator()(unsigned int x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<long> {
    size_t operator()(long x) const { return x; }
};
__STL_TEMPLATE_NULL struct hash<unsigned long> {
    size_t operator()(unsigned long x) const { return x; }
};

__STL_END_NAMESPACE

#endif /* __SGI_STL_HASH_FUN_H */

// Local Variables:
// mode:C++
// End:

```