

G++ 2.91.57, cygnus\cygwin-b20\include\g++\stl\_construct.h 完整列表

```
/*
 *
 * 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.
 *
 *
 * Copyright (c) 1996,1997
 * 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.
 */

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

#ifndef __SGI_STL_INTERNAL_CONSTRUCT_H
#define __SGI_STL_INTERNAL_CONSTRUCT_H

#include <new.h>

__STL_BEGIN_NAMESPACE

template <class T>
inline void destroy(T* pointer) {
    pointer->~T();
}

template <class T1, class T2>
inline void construct(T1* p, const T2& value) {
    new (p) T1(value); // jjhou: placement new; invoke ctor T1(value);
}
```

```
template <class ForwardIterator>
inline void
__destroy_aux(ForwardIterator first, ForwardIterator last, __false_type) {
    for ( ; first < last; ++first)
        destroy(&*first);
}

template <class ForwardIterator>
inline void __destroy_aux(ForwardIterator, ForwardIterator, __true_type) {}

template <class ForwardIterator, class T>
inline void __destroy(ForwardIterator first, ForwardIterator last, T*) {
    typedef typename __type_traits<T>::has_trivial_destructor trivial_destructor;
    __destroy_aux(first, last, trivial_destructor());
}

template <class ForwardIterator>
inline void destroy(ForwardIterator first, ForwardIterator last) {
    __destroy(first, last, value_type(first));
}

inline void destroy(char*, char*) {}
inline void destroy(wchar_t*, wchar_t*) {}

__STL_END_NAMESPACE

#endif /* __SGI_STL_INTERNAL_CONSTRUCT_H */

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