Debugging C with Erlang

John Hughes

Quviq AB/Chalmers University

Free Software Directory

GDSL

The Generic Data Structures Library (GDSL) is a collection of routines for generic data structures manipulation. It is a portable and re-entrant library designed to let C programmers access common data structures with powerful algorithms and hidden implementation. Available structures are lists, queues, stacks, hash tables, binary trees, binary search trees, red-black trees, 2D arrays, and permutations.

Last updated 19 Jul, 2005

User level: Submit a level

User Rating:

Homepage

License(s) :

GPLv2orlater

Rate it!

```
gdsl_list_t gdsl_list_delete( gdsl_list_t           L,
                              gdsl_compare_func_t   COMP_F,
                              const void *          VALUE
                            )
```

Delete a particular element from a list.

Search into the list L for the first element E equal to VALUE by using COMP_F. If E is found, it is removed from L and deallocated using the FREE_F function passed to **gdsl_list_alloc()**.

Note:

Complexity: O(|L| / 2)

Precondition:

L must be a valid gdsl_list_t & COMP_F != NULL

Parameters:

L	The list to destroy the element from
COMP_F	The comparison function used to find the element to destroy
VALUE	The value used to compare the element to destroy with

Returns:

the modified list L if the element is found.

NULL if the element to destroy is not found.

Testing gdsl_list_delete

```
prop_list_delete() ->
  ?FORALL(L,non_empty(list(int())),
    ?FORALL(X,elements(L),
      begin
     New = from_gdsl_list(
                gdsl_list_delete(X,to_gdsl_list(L))),
     New == lists:delete(X,L)
     orelse
     New == reverse(lists:delete(X,reverse(L)))
    end)).
```

```
6
[0,6] /= [6,0]
```

```
gdsl_list_t gdsl_list_delete( gdsl_list_t           L,
                              gdsl_compare_func_t   COMP_F,
                              const void *          VALUE
                            )
```

Delete a particular element from a lis

Search into the list L for the first element E equal to VALUE by using COMP_F. If E is found, it is removed from L and deallocated using the FREE_F function passed to **gdsl_list_alloc()**.

Note:

Complexity: O(|L| / 2)

Precondition:

L must be a valid gdsl_list_t & COMP_F != NULL

Parameters:

L	The list to destroy the element from
COMP_F	The comparison function used to find the element to destroy
VALUE	The value used to compare the element to destroy with

Returns:

the modified list L if the element is found.

NULL if the element to destroy is not found.

Testing the link to C

For a random type T, generate

```
T identity(T x)
{ return x; }
```

…and test identity(x)==x

```
struct s {int a; complex float b;}
```

GCC Bugzilla Bug 39678

Summary: ”complex type isn't passed correctly”

Reported: 7 April 2009

Our counterexample found: 9 April

Patch posted: 9 April

818 lines of comments in Bugzilla

Once the bug was simplified, a patch was quick to write

Want to try?

quviq-licencer.com/promotion.html