LinkedList

1

Generated by Doxygen 1.8.1.2

Fri Mar 22 2013 08:11:00

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 linked_list Struct Reference

`#include <LinkedList.h>`

Data Fields

- pnode head
- pnode last
- pnode runner
- int runnerpos
- int size

3.1.1 Detailed Description

Represents the actual list. Three pointers are defined: head, last and a runner. runnerpos defines the current index of the node* runner. size defines the current size of the list.

3.1.2 Field Documentation

3.1.2.1 pnode head

Pointer to the first element of the list.

3.1.2.2 pnode last

Pointer to the last element of the list.

3.1.2.3 pnode runner

Pointer to some element of the list. The current position is specified in runnerpos.

3.1.2.4 int runnerpos

Defines the current position of runner.

3.1.2.5 int size

Stores the list's current number of elements.

The documentation for this struct was generated from the following file:

- /home/paul/Git_home/Util/LinkedList/LibLinkedList/LinkedList.h

3.2 node Struct Reference

`#include <LinkedList.h>`

Data Fields

- struct node * next
- struct node * previous
- void * content

3.2.1 Detailed Description

Represents the data node. Pointer to following and previous node are stored.

3.2.2 Field Documentation

3.2.2.1 void* content

The actual content of this Node. The size only knows the creator of this object. If there are dynamically sized objects to be stored here with corresponding size it is recommended to create a struct with these informations and store a pointer to that struct in a Node.

3.2.2.2 struct node* next

Pointer to the following Node.

3.2.2.3 struct node* previous

Pointer to the previous Node. previous->next should be this Node.

The documentation for this struct was generated from the following file:

- /home/paul/Git_home/Util/LinkedList/LibLinkedList/LinkedList.h

Chapter 4

File Documentation

4.1 /home/paul/Git_home/Util/LinkedList/LibLinkedList/Debug/LinkedList.d File Reference

4.2 /home/paul/Git_home/Util/LinkedList/LibLinkedList/LinkedList.c File Reference

```
#include "LinkedList.h"
#include <stdlib.h>
#include <stdio.h>
```

Functions

- int abs (int a)
- void run (plinked_list list, int position)
- void list_append (plinked_list list, void *value)
- void list_insert (plinked_list list, int position, void *value)
- void list_remove (plinked_list list, int position)
- void * list_get (plinked_list list, int position)
- void list_add_all (plinked_list list, void **elems, int count)
- plinked_list init_list ()

4.2.1 Function Documentation

4.2.1.1 int abs (int *a*) [inline]

4.2.1.2 plinked_list init_list ( )

Initializes an empty list. Works like an constructor.

4.2.1.3 void list_add_all (plinked_list *list*, void ** *elems*, int *count*)

Inserts all elements of elems into the list. count has to be defined.

4.2.1.4 void list_append (plinked_list *list*, void * *value*)

Appends value to the end of the list.

4.2.1.5 void* list_get (plinked_list *list,* int *position*)

Returns a pointer to the object at position.

4.2.1.6 void list_insert (plinked_list *list,* int *position,* void * *value*)

Inserts value into the list at the specified position. The current object at position will be at (position+1) after this operation.

4.2.1.7 void list_remove (plinked_list *list,* int *position*)

Removes the element of the list at the specified position.

4.2.1.8 void run (plinked_list *list,* int *position*)

Run with the list->runner to the position

4.3 /home/paul/Git_home/Util/LinkedList/LibLinkedList/LinkedList.h File Reference

Data Structures

- struct node
- struct linked_list

Typedefs

- typedef struct node node_t
- typedef node_t * pnode
- typedef struct linked_list linked_list_t
- typedef linked_list_t * plinked_list

Functions

- void list_append (plinked_list list, void *value)
- void list_insert (plinked_list list, int position, void *value)
- void list_remove (plinked_list list, int position)
- void * list_get (plinked_list list, int position)
- void list_add_all (plinked_list list, void **elems, int count)
- plinked_list init_list ()

4.3.1 Typedef Documentation

4.3.1.1 typedef struct linked_list linked_list_t

4.3.1.2 typedef struct node node_t

4.3.1.3 typedef linked_list_t* plinked_list

4.3.1.4 typedef node_t* pnode

4.3.2 Function Documentation

4.3.2.1 plinked_list init_list ( )

Initializes an empty list. Works like an constructor.

4.3.2.2 void list_add_all (plinked_list *list,* void ** *elems,* int *count*)

Inserts all elements of elems into the list. count has to be defined.

4.3.2.3 void list_append (plinked_list *list,* void * *value*)

Appends value to the end of the list.

4.3.2.4 void* list_get (plinked_list *list,* int *position*)

Returns a pointer to the object at position.

4.3.2.5 void list_insert (plinked_list *list,* int *position,* void * *value*)

Inserts value into the list at the specified position. The current object at position will be at (position+1) after this operation.

4.3.2.6 void list_remove (plinked_list *list,* int *position*)

Removes the element of the list at the specified position.

Index