Class Demo Singly Linked List

0.1.0

Generated by Doxygen 1.8.17

1 Class Index 1

1.1 Class List . . . 1

2 File Index 3

2.1 File List . . . 3

3 Class Documentation 5

3.1 Node Class Reference . . . 5

3.1.1 Detailed Description . . . 5

3.1.2 Constructor & Destructor Documentation . . . 5

3.1.2.1 Node() . . . 6

3.1.3 Member Data Documentation . . . 6

3.1.3.1 data . . . 6

3.1.3.2 nextNode . . . 6

3.2 SLL Class Reference . . . 6

3.2.1 Detailed Description . . . 7

3.2.2 Constructor & Destructor Documentation . . . 7

3.2.2.1 SLL() . . . 7

3.2.3 Member Function Documentation . . . 7

3.2.3.1 addToTail() . . . 7

3.2.3.2 printList() . . . 8

3.2.3.3 removeHead() . . . 8

3.2.4 Member Data Documentation . . . 9

3.2.4.1 head . . . 9

3.2.4.2 n . . . 9

3.2.4.3 tail . . . 9

4 File Documentation 11

4.1 /home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp File Reference . . . 11

4.1.1 Detailed Description . . . 12

4.1.2 Function Documentation . . . 12

4.1.2.1 main() . . . 12

Index 13

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Node 5
SLL 6

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp
This is a demo of making a singly linked list 11

Chapter 3

Class Documentation

3.1 Node Class Reference

Collaboration diagram for Node:

Public Member Functions

- Node (int d)

Public Attributes

- int data
- Node * nextNode

3.1.1 Detailed Description

Definition at line 13 of file main.cpp.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Node()

```
Node::Node (
            int d )  [inline]
```

Constructor

Definition at line 21 of file main.cpp.

```
21                  {
22          data = d;
23          nextNode = NULL;
24      }
```

3.1.3 Member Data Documentation

3.1.3.1 data

```
int Node::data
```

Definition at line 15 of file main.cpp.

3.1.3.2 nextNode

```
Node* Node::nextNode
```

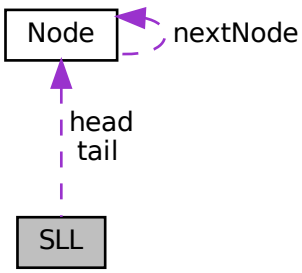
Definition at line 16 of file main.cpp.

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

- /home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp

3.2 SLL Class Reference

Collaboration diagram for SLL:



Public Member Functions

- SLL ()
- bool addToTail (int d)
- int removeHead ()
- void printList ()

Public Attributes

- Node * head
- Node * tail
- int n

3.2.1 Detailed Description

Definition at line 27 of file main.cpp.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 SLL()

```
SLL::SLL ( )  [inline]
```

Constructor

Definition at line 36 of file main.cpp.

```
36          {
37         head = NULL;
38         tail = NULL;
39         n = 0;
40     }
```

3.2.3 Member Function Documentation

3.2.3.1 addToTail()

```
bool SLL::addToTail (
             int d )  [inline]
```

Adds node to tail of list

Parameters

d	integer to add to tail of list

Definition at line 47 of file main.cpp.

```
47                              {
48          Node* newNode = new Node(d);
49          if(n == 0) { // the list is empty
50              head = newNode;
51              tail = newNode;
52              n++;
53          } else { // list is not empty
54              tail->nextNode = newNode; // point tail node to newNode
55              tail = newNode; // point tail to newNode
56              n++; // increment counter
57          }
58          return(true);
59      }
```

3.2.3.2 printList()

```
void SLL::printList ( )  [inline]
```

prints the contents of the singly linked list

Definition at line 82 of file main.cpp.

```
82                          {
83          Node* curNode;
84          //if(n == 0) { // the list is empty
85          if(head == NULL) { // the list is empty could also check that head != NULL
86              cout « "Empty list" « endl;
87          } else {
88              curNode = head;
89              //for(int ii = 0; ii < n; ii++){ // since n is known
90              while(curNode->nextNode != NULL) { // not dependent on n
91                  cout « curNode->data;
92                  if(curNode->nextNode != NULL){
93                      cout « " -> ";
94                  }
95                  curNode = curNode->nextNode;
96              }
97              cout « curNode->data; // required for while loop method
98              cout « endl;
99          }
100     }
```

3.2.3.3 removeHead()

```
int SLL::removeHead ( )  [inline]
```

Removes the head node and returns the data from it

Definition at line 64 of file main.cpp.

```
64                          {
65          int val;
66          Node* old;
67          if(head != NULL) {
68              val = head->data; // collect the value
69              old = head; // collect pointer to head node to delete it
70              head = head->nextNode; // move the head pointer
71              delete old; // free the memory used by this node required since created with new
72              n--; // without this get a segfault
73              return(val);
74          } else {
75              return(-999999); // need to pass by reference and return bool to fix this
76          }
77      }
```

3.2.4 Member Data Documentation

3.2.4.1 head

```
Node* SLL::head
```

Definition at line 29 of file main.cpp.

3.2.4.2 n

```
int SLL::n
```

Definition at line 31 of file main.cpp.

3.2.4.3 tail

```
Node* SLL::tail
```

Definition at line 30 of file main.cpp.

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

- /home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp

Chapter 4

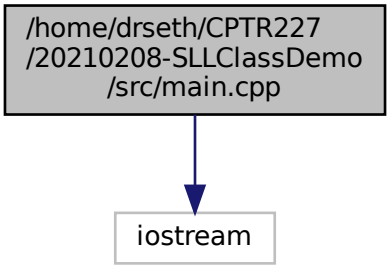
File Documentation

4.1 /home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp File Reference

This is a demo of making a singly linked list.

```
#include <iostream>
```

Include dependency graph for main.cpp:



Classes

- class Node
- class SLL

Functions

- int main (int, char **)

4.1.1 Detailed Description

This is a demo of making a singly linked list.

Based on ODS book examples

Author

Seth McNeill

Date

2021 February 08

4.1.2 Function Documentation

4.1.2.1 main()

```
int main (
             int ,
             char **  )
```

Definition at line 103 of file main.cpp.

```
103                       {
104     SLL myList;
105
106     myList.printList();
107     myList.addToTail(1);
108     myList.printList();
109     myList.addToTail(2);
110     myList.printList();
111     myList.addToTail(3);
112     myList.printList();
113     myList.addToTail(4);
114     myList.printList();
115     cout « "removed " « myList.removeHead() « endl;
116     myList.printList();
117     cout « "removed " « myList.removeHead() « endl;
118     myList.printList();
119     cout « "removed " « myList.removeHead() « endl;
120     myList.printList();
121     cout « "removed " « myList.removeHead() « endl;
122     myList.printList();
123     cout « "removed " « myList.removeHead() « endl;
124     myList.printList();
125     cout « "removed " « myList.removeHead() « endl;
126     myList.printList();
127 }
```

Index

/home/drseth/CPTR227/20210208-SLLClassDemo/src/main.cpp,
11

addToTail
SLL, 7

data
Node, 6

head
SLL, 9

main
main.cpp, 12
main.cpp
main, 12

n
SLL, 9
nextNode
Node, 6
Node, 5
data, 6
nextNode, 6
Node, 5

printList
SLL, 8

removeHead
SLL, 8

SLL, 6
addToTail, 7
head, 9
n, 9
printList, 8
removeHead, 8
SLL, 7
tail, 9

tail
SLL, 9