
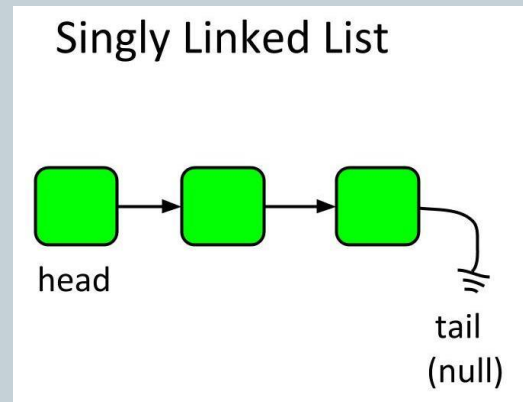


And Even More C++

A large, blue, 3D-rendered logo of the C++ programming language. The 'C' is a large, rounded letter, and the two '+' signs are smaller, positioned to the right of the 'C'. The logo has a slight shadow and a gradient, giving it a three-dimensional appearance.

Outline

- Coming Up:
 - C++ Classes
 - Special Members
 - Friendship
- But first...
 - A review of linked lists

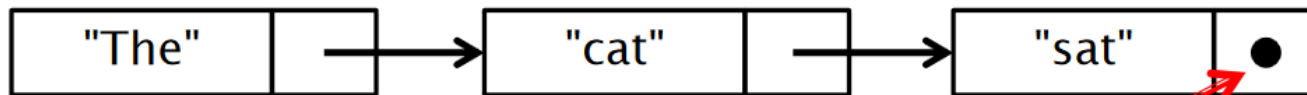


Linked List

- Linked list
 - Simplest linked data structure
 - A recursive data structure
 - Each node contains:
 - ✦ An item (some data)
 - ✦ A pointer to next node in the list
 - An inner-class, declared inside parent class

```
private class Node
{
    private String item;
    private Node next;
}
```

Three Node objects hooked together to form a linked list



Special pointer value null terminates the list.
We denote with a dot.

Traversing a List

- Iterate over all elements in a linked list
 - Assume list is null terminated
 - Assume `first` instance variable points to start of list
 - Print all the strings in the list

```
Node current = first;  
while (current != null)  
{  
    System.out.println(current.item);  
    current = current.next;  
}
```

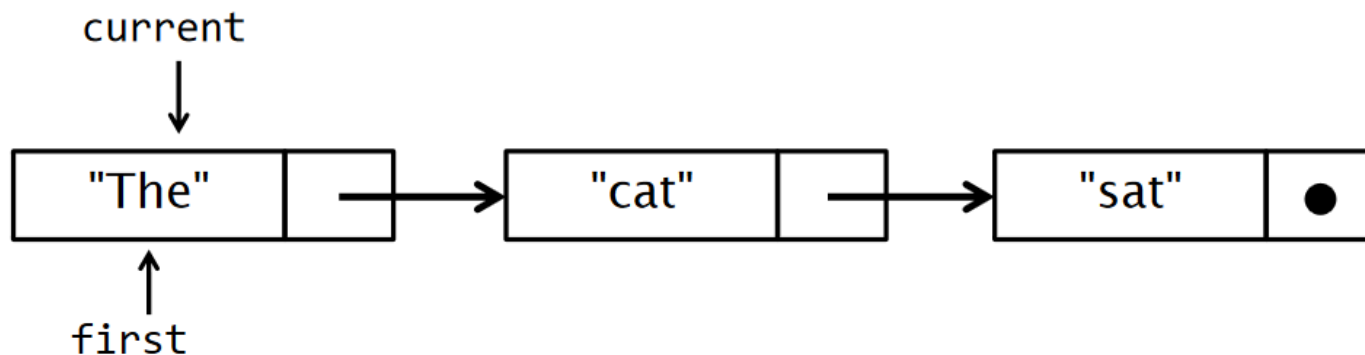
```
for (Node current = first; current != null; current = current.next)  
    System.out.println(current.item);
```

shorthand version



Traversing a List

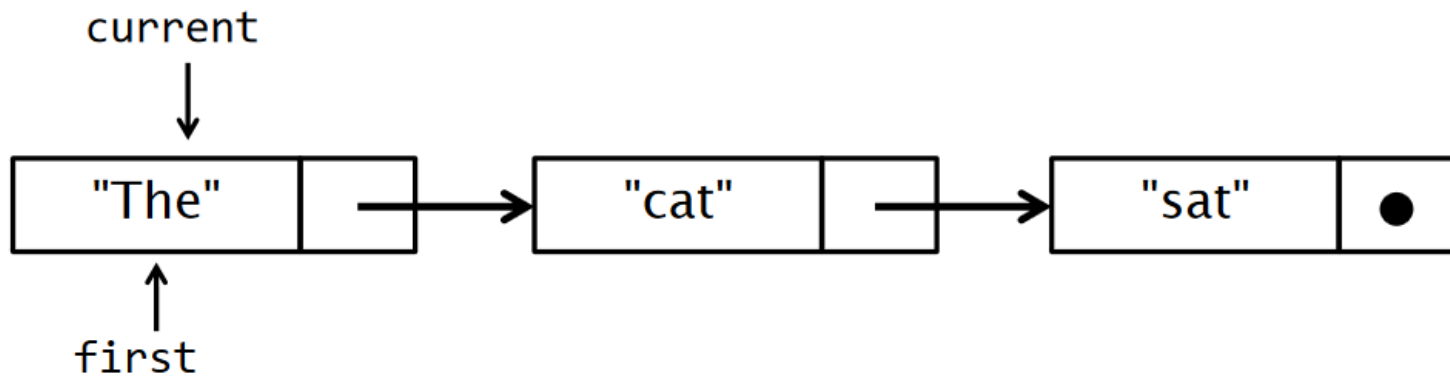
```
→ Node current = first;  
while (current != null)  
{  
    System.out.println(current.item);  
    current = current.next;  
}
```



Traversing a List

```
Node current = first;  
while (current != null)  
{  
    ➡ System.out.println(current.item);  
    current = current.next;  
}
```

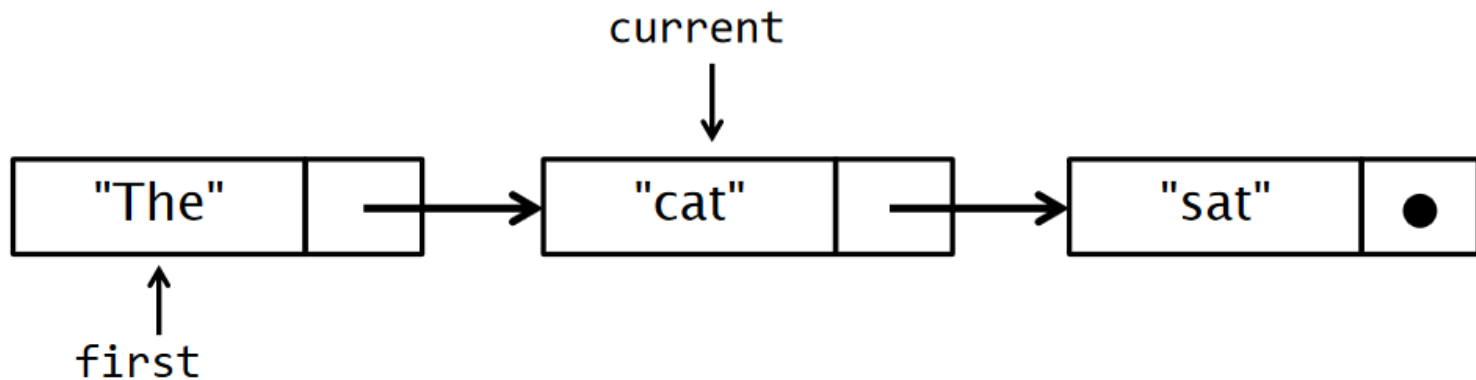
The



Traversing a List

```
Node current = first;  
while (current != null)  
{  
    System.out.println(current.item);  
    → current = current.next;  
}
```

The



Playing with a Linked List

- What things might we want to do with a list?
 - Construct a node
 - Add a node to the end
 - Insert a node at a certain position
 - Remove a node from a position
 - Print out the list of nodes

Playing with a Linked List

- What things might we want to do with a list?
 - Construct a node

Data Structures

- A data structure is a group of data elements grouped together under one name
 - Not quite the same thing as a data type in Java
- Use struct to define a structure in C++

```
struct type_name {  
    member_type1 member_name1;  
    member_type2 member_name2;  
    member_type3 member_name3;  
    .  
    .  
} object_names;
```

```
struct product {  
    int weight;  
    double price;  
} ;  
  
product apple;  
product banana, melon;
```

```
struct product {  
    int weight;  
    double price;  
} apple, banana, melon;
```

Pointers to Structures

- The arrow operator -> is used to access structures that have member elements

```
struct movies_t {  
    string title;  
    int year;  
};
```

```
movies_t amovie;  
movies_t * pmovie;
```

```
pmovie = &amovie;
```

```
pmovie->title
```

is equivalent to:

```
(*pmovie).title
```

```
*pmovie.title
```

is equivalent to:

```
*(pmovie.title)
```

Expression	What is evaluated	Equivalent
a.b	Member b of object a	
a->b	Member b of object pointed to by a	(*a).b
*a.b	Value pointed to by member b of object a	*(a.b)

new and new[]

- new is followed by a data type specifier and if there are multiple elements needed, brackets are used, to specify an array

```
pointer = new type  
pointer = new type [number_of_elements]
```

- For example:

```
int * foo;  
foo = new int [5];
```

- In this example, a pointer to an integer is created, and then a block of memory is allocated to store 5 of them

Playing with a Linked List

- What things might we want to do with a list?
 - Add a node to the end

Playing with a Linked List

- What things might we want to do with a list?
 - Insert a node at a certain position

Playing with a Linked List

- What things might we want to do with a list?
 - Remove a node from a position

delete and delete[]

- C++ does not handle garbage collection for you
 - You need to determine when a particular data item is no longer needed and then remove it
 - Use delete and delete[] to do this

```
delete pointer;  
delete[] pointer;
```

- The “thing” deleted should be either something that was created with new or new[] before, or it should be a null pointer (in which case nothing happens)

Playing with a Linked List

- What things might we want to do with a list?
 - Print out the list of nodes

Summary

- Coming Up:
 - C++ Classes
 - Special Members
 - Friendship
- But first...
 - A review of linked lists

