

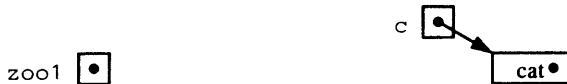
Figure 4.5 Detailed structures of SLL objects and SLLNode objects.

Note the terminology used here. When we talk in general terms about linked-list data structures and algorithms, we talk about *nodes* connected by *links*. When we talk about their implementation in Java, we represent nodes by *objects* connected by *references* to other objects. (In fact, every Java object is accessed through a reference.)

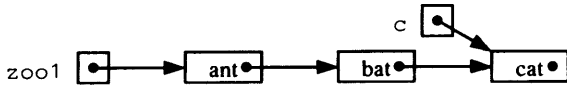
**EXAMPLE 4.1 SLL construction**

The following Java code and diagrams illustrate construction of an SLL:

```
SLLNode c = new SLLNode("cat", null),
SLL zoo1 = new SLL();
```



```
zoo1.first = new SLLNode("ant",
    new SLLNode("bat", c));
```



**EXAMPLE 4.2 SLL traversal**

The following Java method traverses an SLL from its first node to its last node:

```
public void printFirstToLast () {
// Print all the elements in this SLL, in first-to-last order.
    for (SLLNode curr = first; curr != null;
        curr = curr.succ)
        System.out.print(curr.element + " ");
}
```

This would be an instance method of the SLL class.