链表

周云晓

2017.4.27

内容

链表概念

节点

链表操作

概念

例子

幼儿园的老师带领孩子出来散步，老师牵着第1个小孩的手，第1个小孩的另一只手牵着第2个孩子 ……

这就是一个“链”，最后一个孩子有一只手空着，他就是“链尾”。

概念

数组的不足

大小不可变

插入一项要移动数组中其他数据

概念

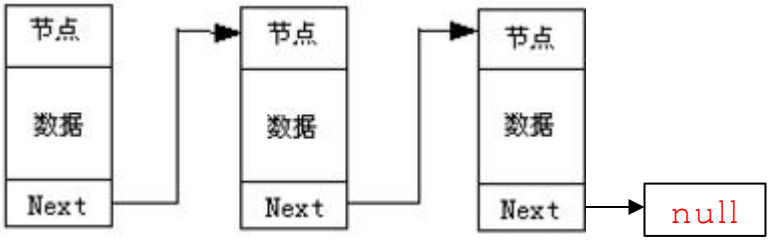
链表

动态地进行存储分配的一种结构

链表内容通常存储于内存中分散的位置

链表由节点组成

使用引用将存储数据元素的节点依次串联在一起



内容

链表概念

节点

链表操作

概念

节点 (Node)

每一个节点结构都相同

定义描述节点的类

数据域 & 链域

数据域： 存放节点的数据元素

链　域： 存放对下一个节点的引用

节点

Node类

```
class Node{
    StudentRecord data;  //数据域

    Node next;           //链域
}
//节点存放的数据类
class StudentRecord{
    public int studentID;
    public String name;
    public double gpa;

    public StudentRecord(int studentID, String name, double gpa){
        this.studentID = studentID;
        this.name = name;
        this.gpa = gpa;
    }
}
```

Node类的完善

```
class Node{
    StudentRecord data;
    Node next;

    //构造函数
      public Node(StudentRecord data){
       setData(data);
       setNext(null);
    }

    public void setData(StudentRecord data){
       this.data = data;
    }

    public void setNext(Node next){
       this.next = next;
    }
}
```

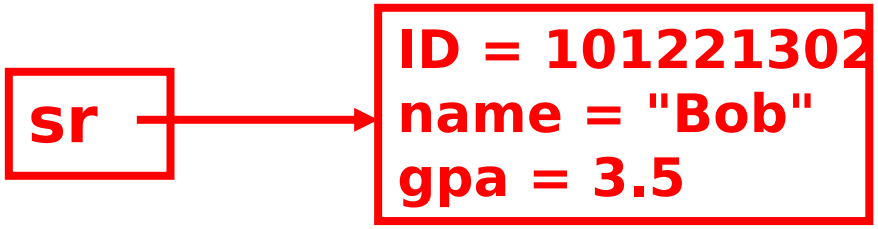
节点 -- Test

```
public static void main(String args[]) {
    StudentRecord sr =
            new StudentRecord(101221302, "Bob", 3.5);

    Node n1 = new Node(null);
    System.out.println("Empty node test\n" + n1);

    n1.setData(sr);
    System.out.println("Bob: "+n1);

    sr = new StudentRecord(101221303, "Mary", 3.7);
    Node n2 = new Node(sr);
    n1.setNext(n2);
    System.out.println("Bob: "+n1);
    System.out.println("Mary: "+n2);
}
```

节点 -- Test

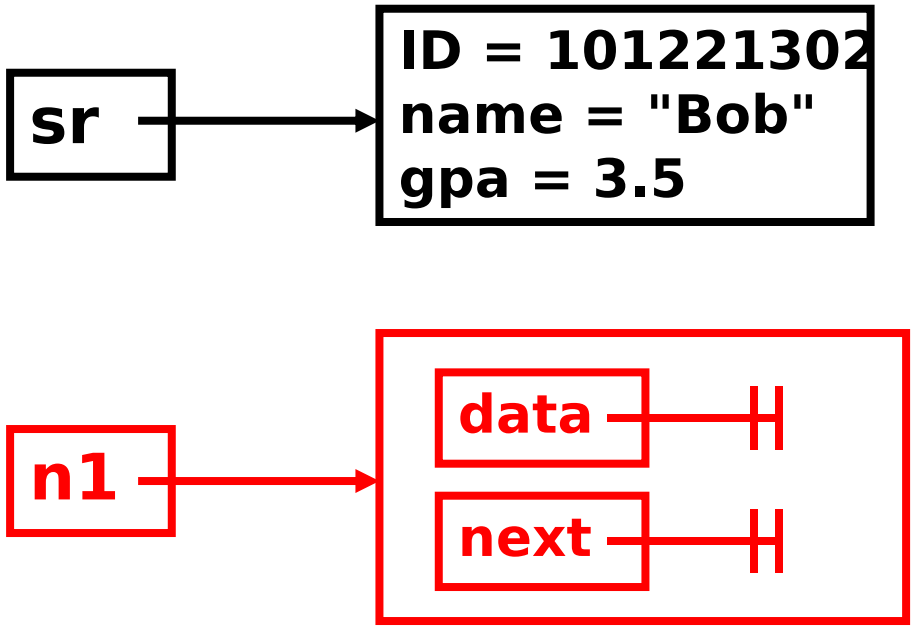
Let's see what's happening!

```
StudentRecord sr =
        new StudentRecord(101221302, "Bob", 3.5);
```



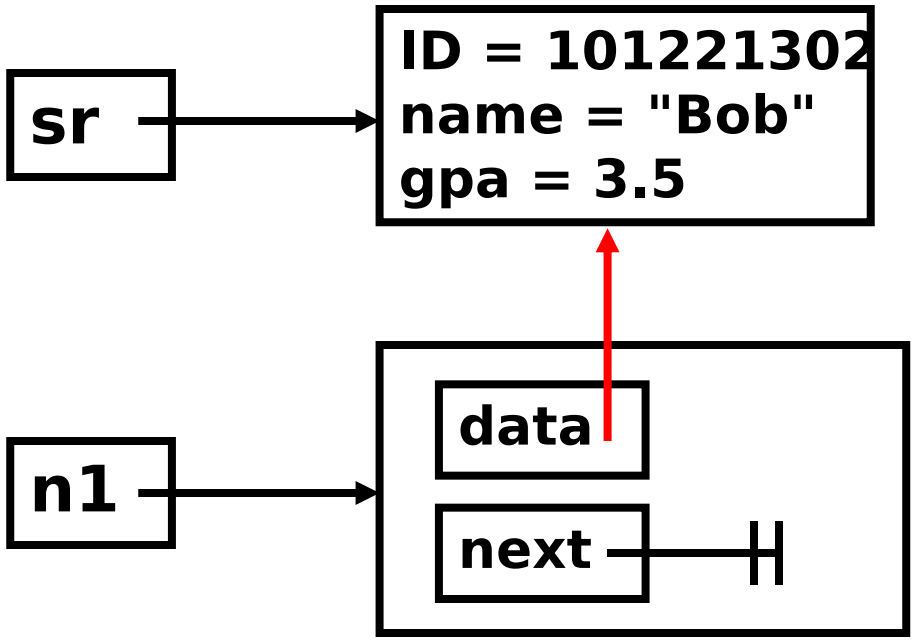
节点 -- Test

```
Node n1 = new Node(null);
```



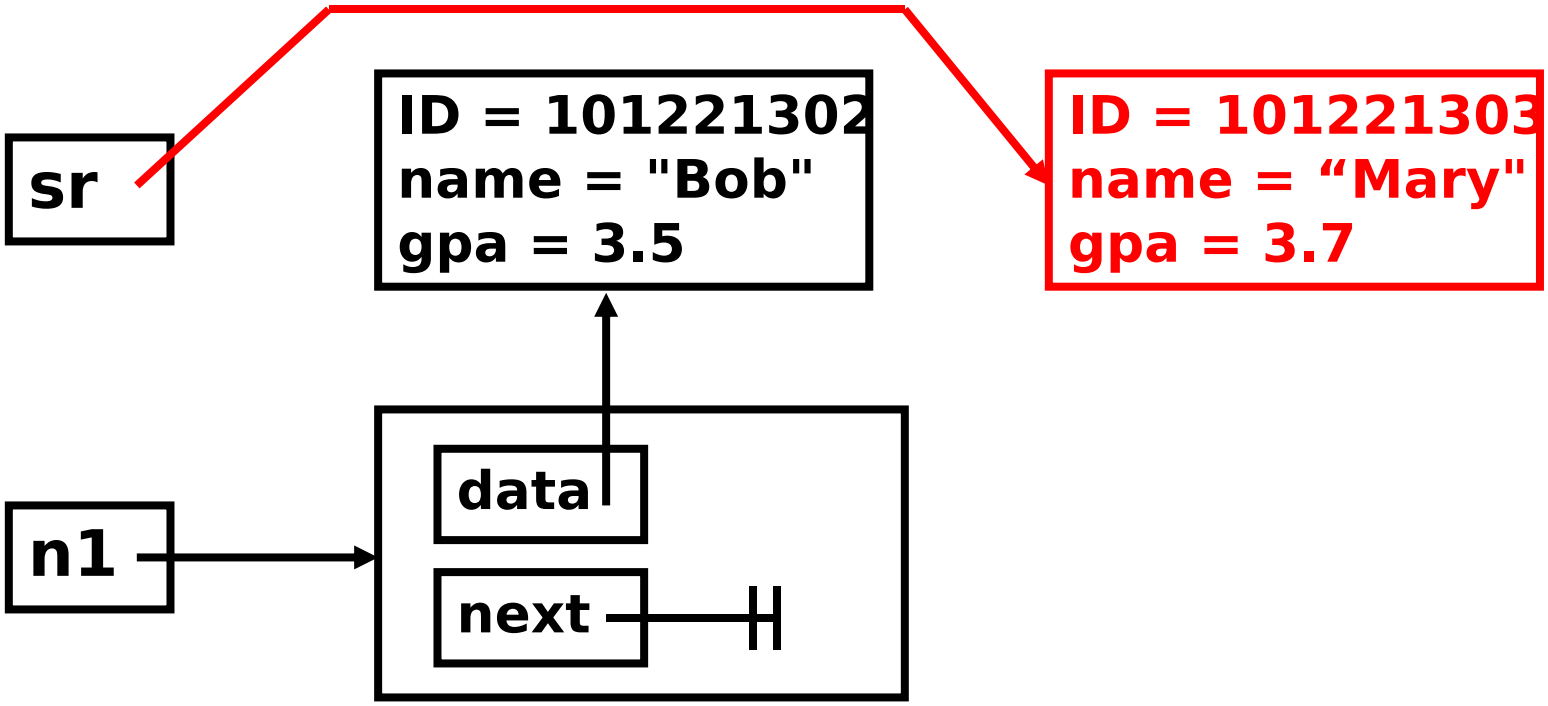
节点 -- Test

n1.setData(sr);



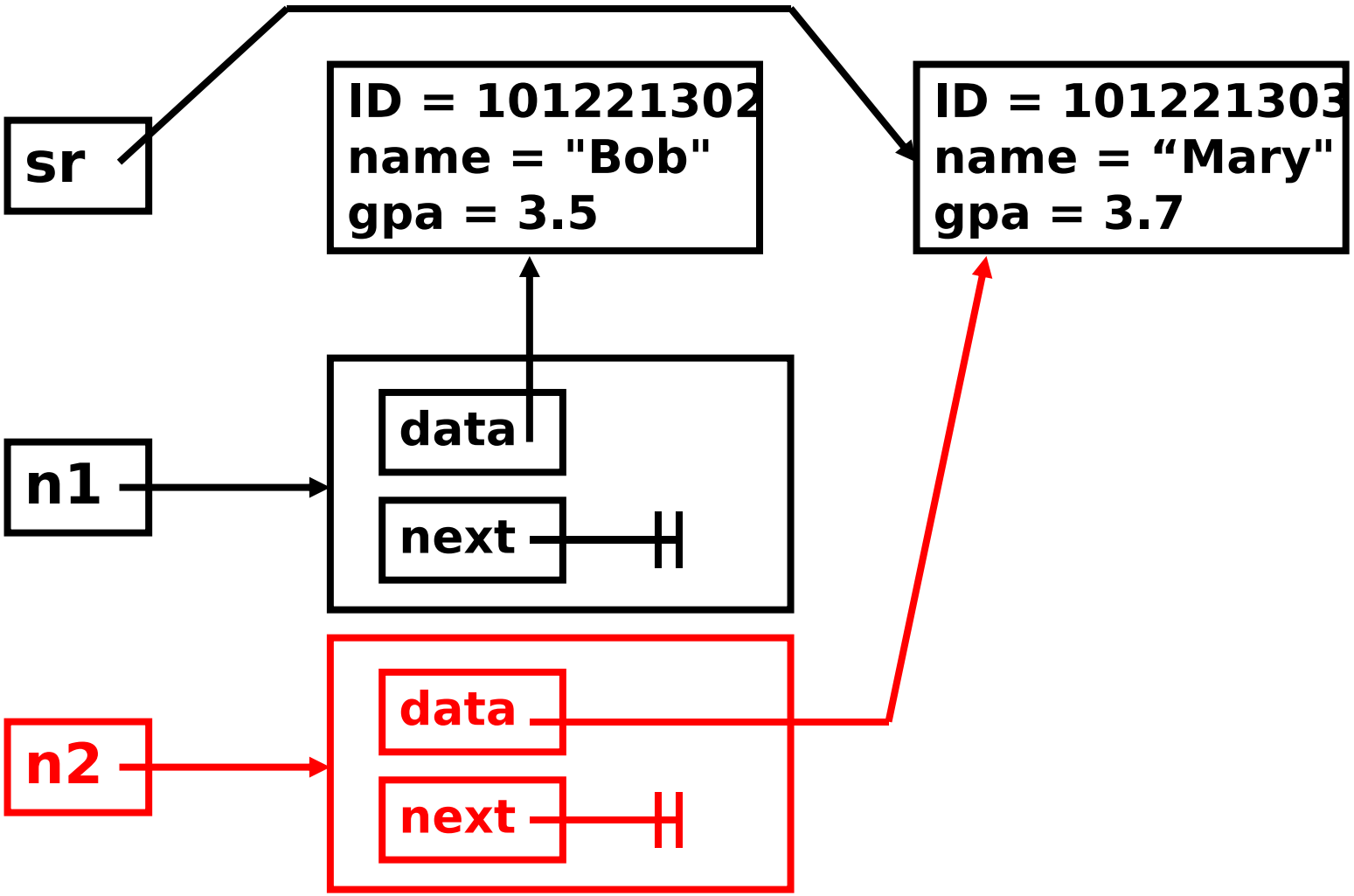
节点 -- Test

```
sr = new StudentRecord(101221303, "Mary", 3.7);
```



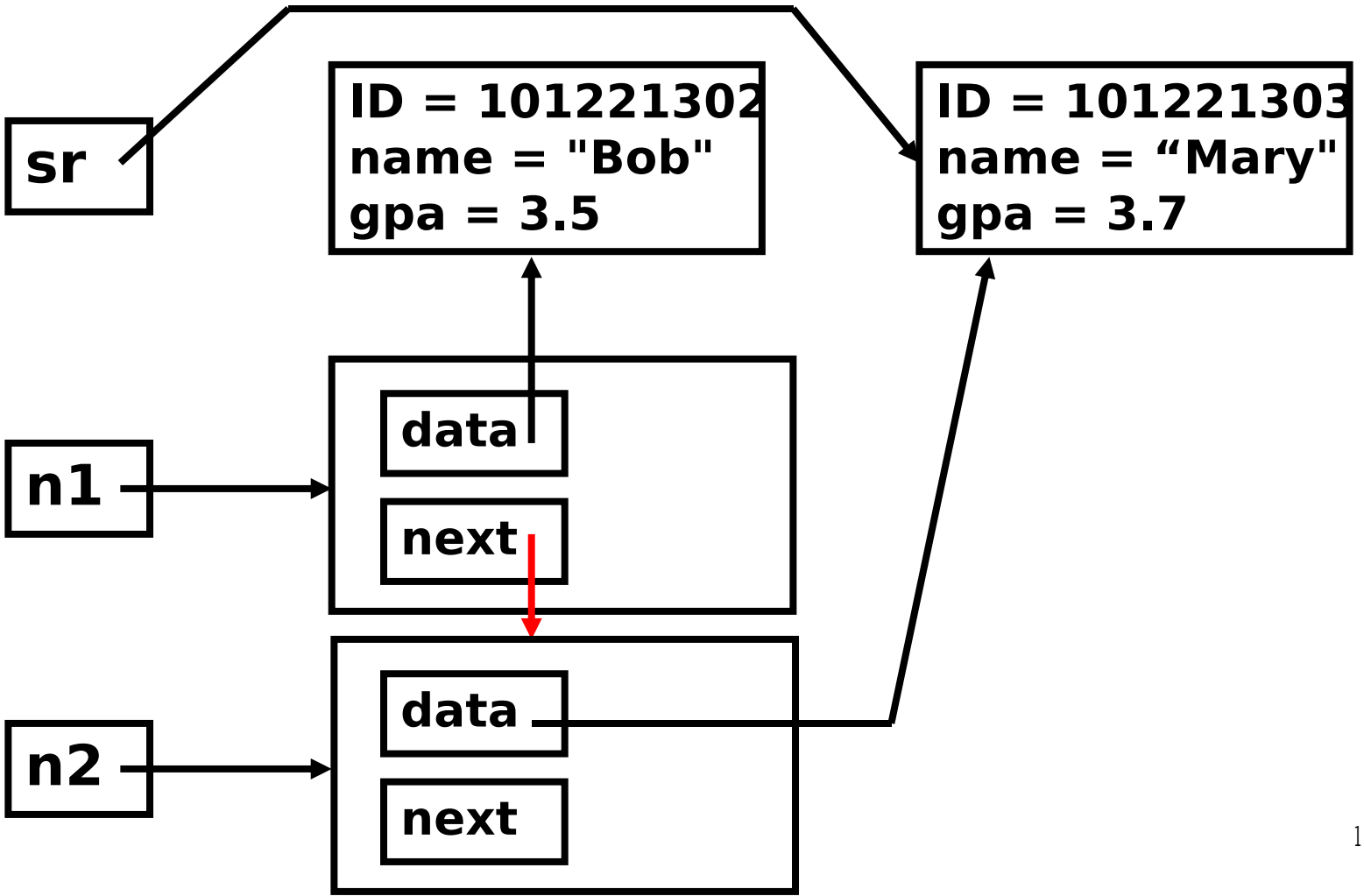
节点 -- Test

Node n2 = new Node(sr);



节点 -- Test

n1.setNext(n2);



内容

链表概念

节点

链表操作

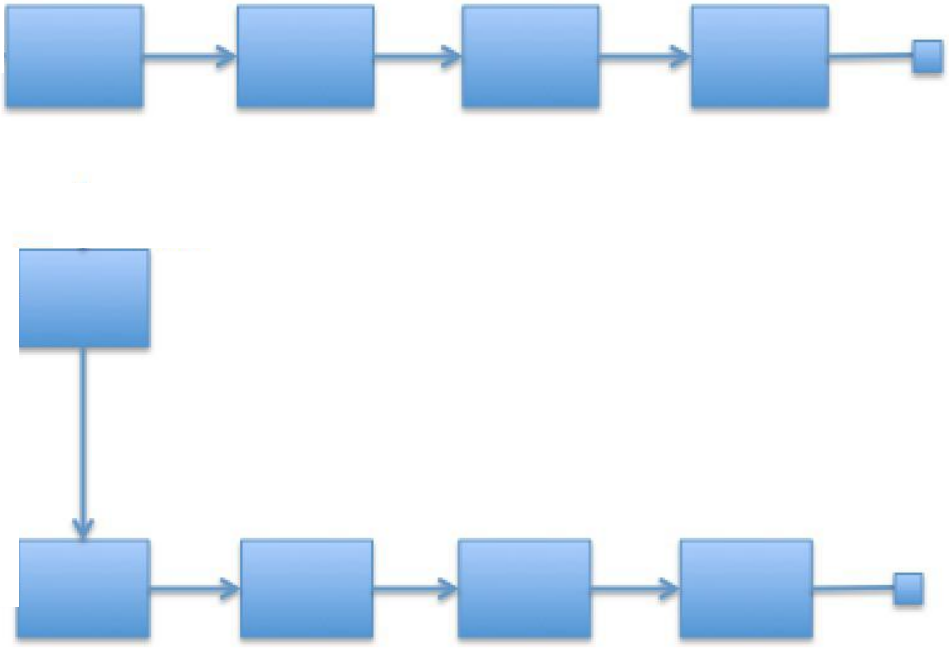
链表的操作 -- 构建

创建链表

```
public class LinkedList {

    private Node head;

    public LinkedList(){
        setHead(null);
    }

    private void setHead(Node head){
        this.head = head;
    }

    private Node getHead(){
        return head;
    }
}
```

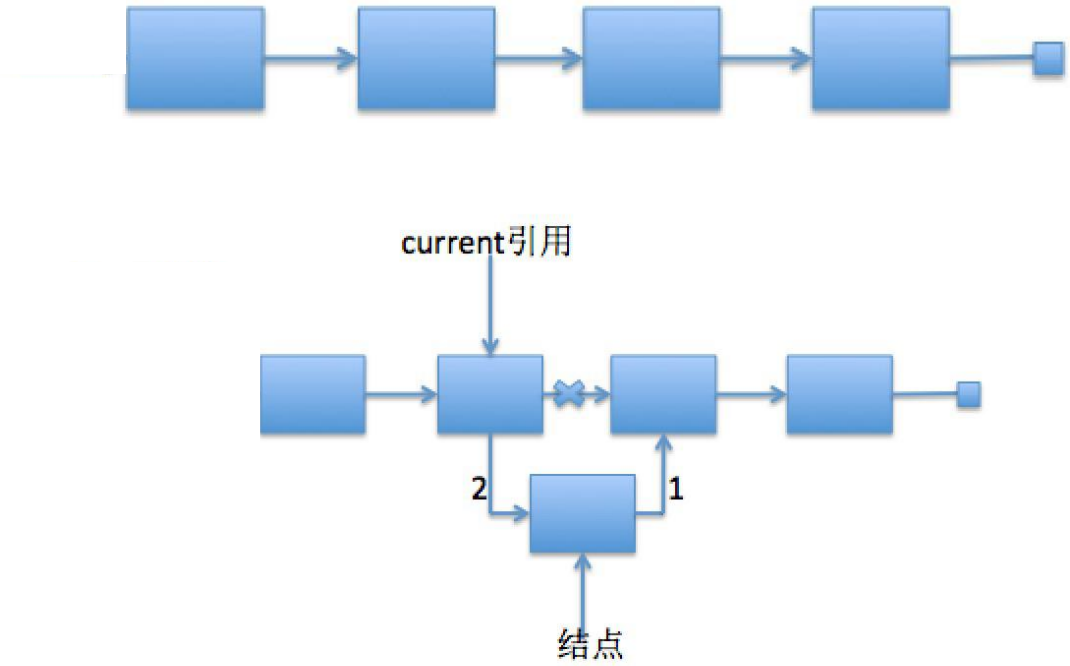
链表的操作 -- 插入

在首部插入节点



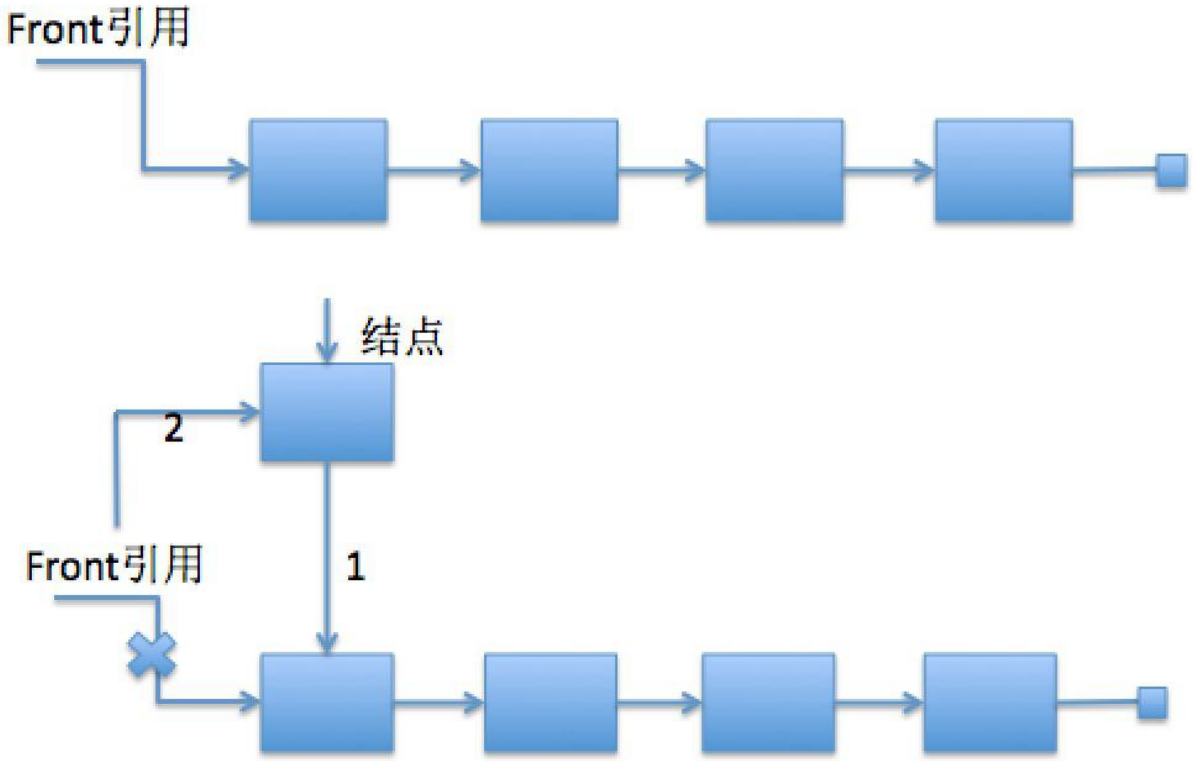
链表的操作 -- 插入

在其他部分插入节点



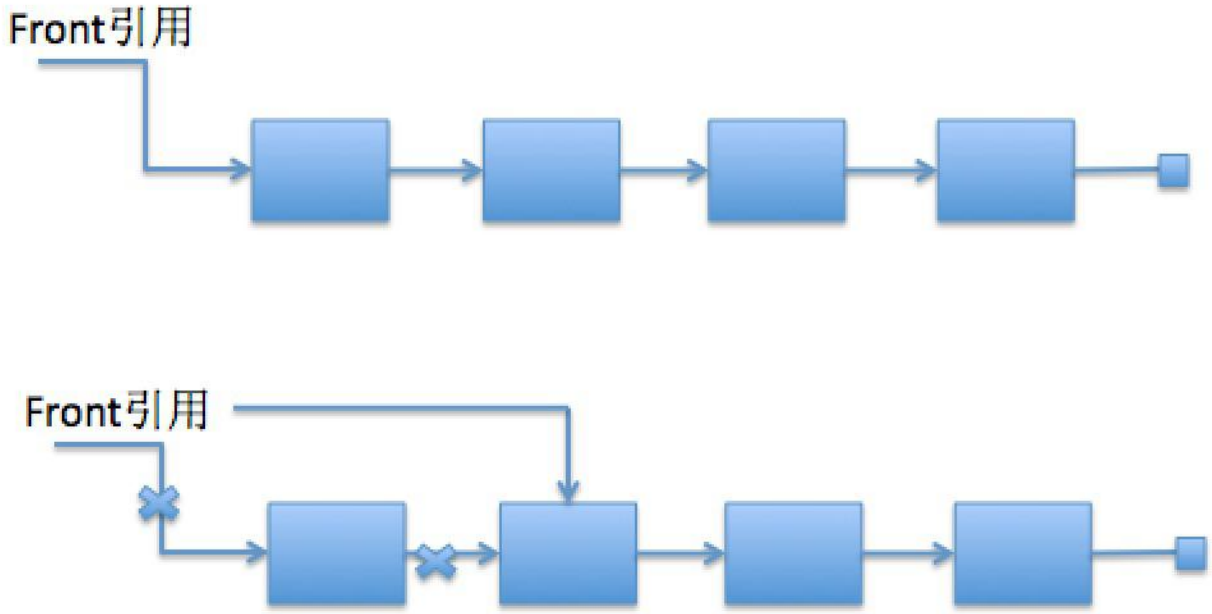
链表的操作 -- 插入

在首部插入节点 -- 添加哨兵



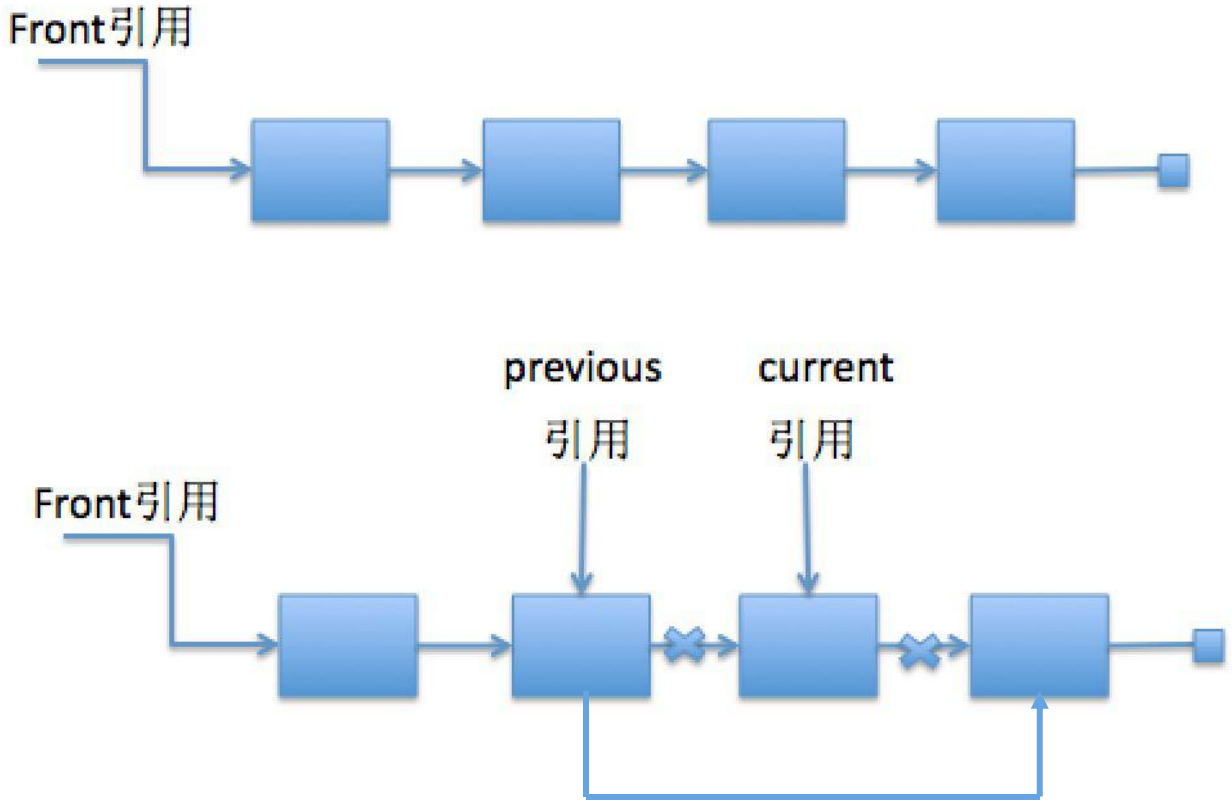
链表的操作 -- 删除

删除第一个节点



链表的操作 -- 删除

删除内部节点



拓展

JDK提供了链表数据结构：

java.util.LinkedList

https://docs.oracle.com/javase/7/docs/api/java/util/LinkedList.html