Linked List implementation

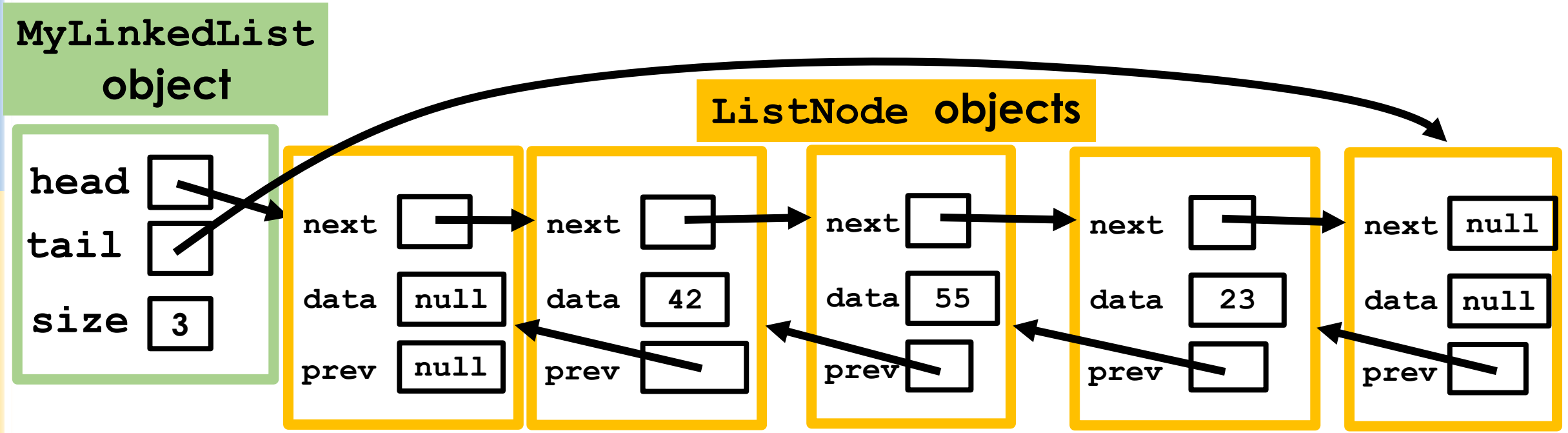
Java code details

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License by Christine Alvarado, Mia Minnes, and Leo Porter, 2015.

By the end of this video you will be able to…

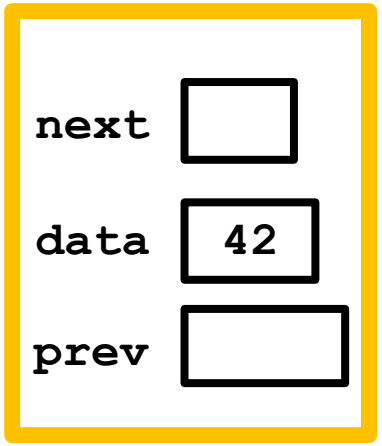
- Implement a doubly linked list with sentinel nodes in Java

Implementing a Linked List in Java



```
class ListNode<E> {
    ListNode<E> next;
    ListNode<E> prev;
    E data;

    public ListNode(E theData)
    {
        this.data = theData;
    }
}
```

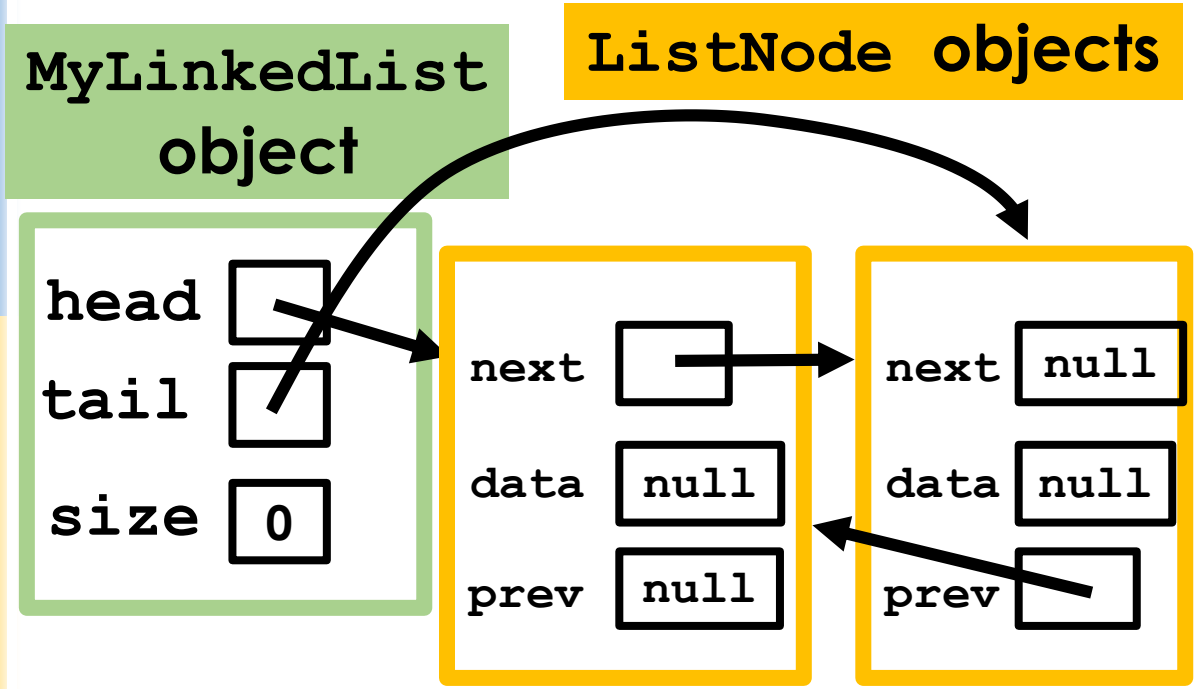
class ListNode



Recursive data type!

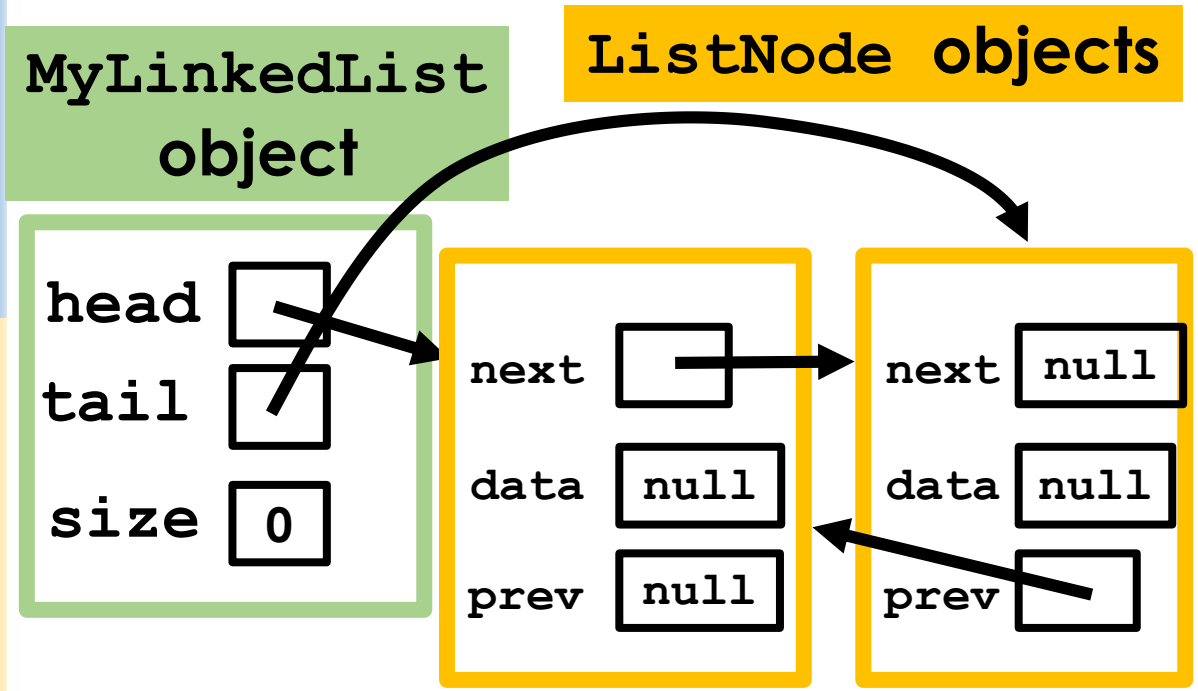
No type parameter in the constructor header

Implementing a Linked List in Java

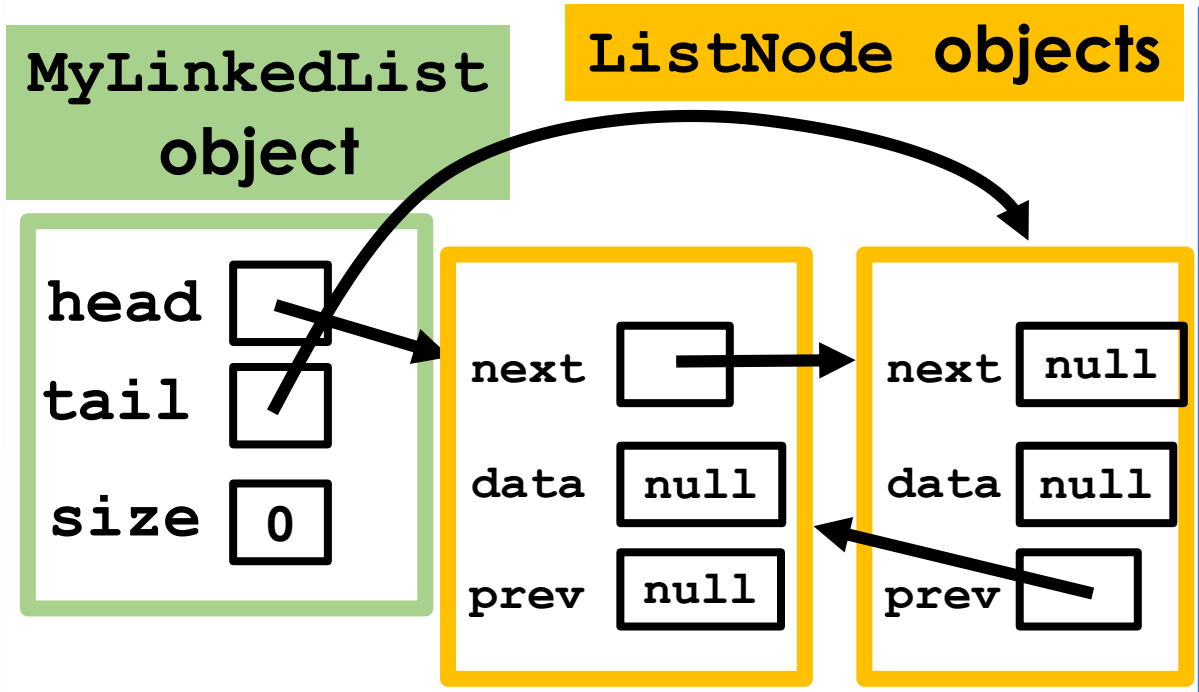



```
public class MyLinkedList<E>
{
   private _______ head;
   private _______ tail;
   private int size;
```

What goes in the blank for the type of `head` and `tail` in `MyLinkedList<E>`?

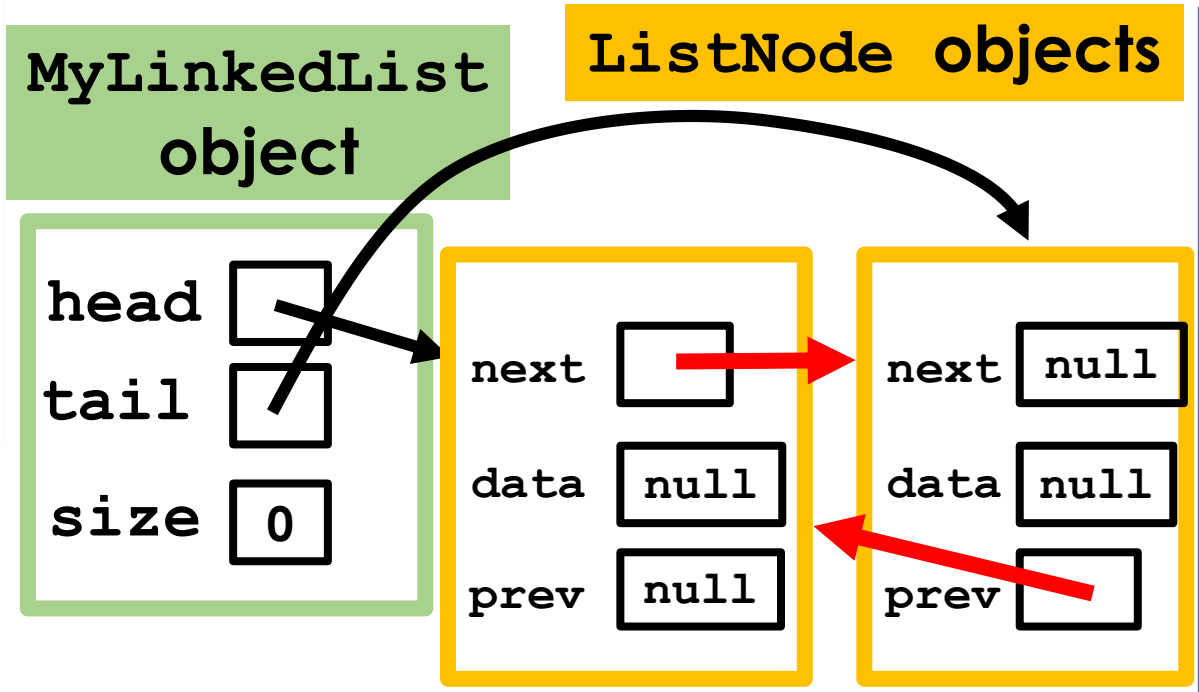



```
public class MyLinkedList<E>
{
   private ListNode<E> head;
   private ListNode<E> tail;
   private int size;
```

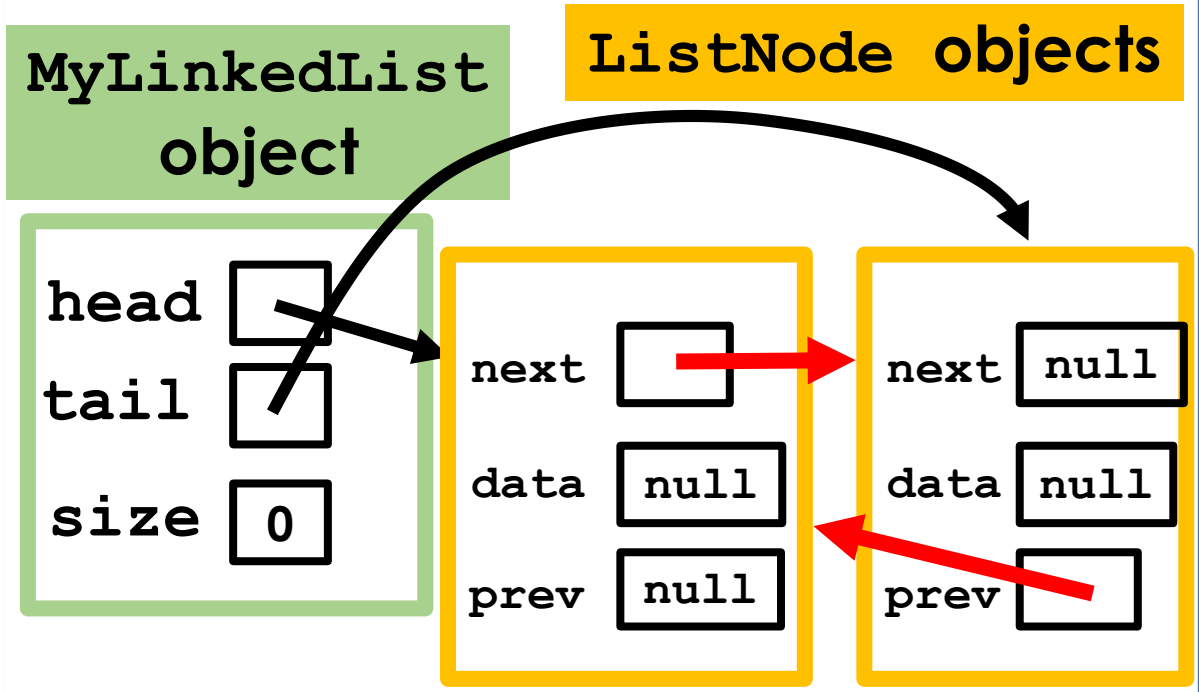



```
public class MyLinkedList<E>
{
   private ListNode<E> head;
   private ListNode<E> tail;
   private int size;

   public MyLinkedList() {
      size = 0;
      head = new ListNode<E>(null);
      tail = new ListNode<E>(null);
   }
```

Does this constructor correctly create the diagram as shown above?




```
public class MyLinkedList<E>
{
   private ListNode<E> head;
   private ListNode<E> tail;
   private int size;

   public MyLinkedList() {
      size = 0;
      head = new ListNode<E>(null);
      tail = new ListNode<E>(null);
   }
```



You will implement:
size, get, set, add, remove

```
public class MyLinkedList<E>
{
   private ListNode<E> head;
   private ListNode<E> tail;
   private int size;

   public MyLinkedList() {
      size = 0;
      head = new ListNode<E>(null);
      tail = new ListNode<E>(null);
      head.next = tail;
      tail.prev = head;
   }
```