Hands-On: Linked Nodes

This activity focuses on the fundamental mechanics of using singly- and doubly-linked nodes as containers for physical storage. You should study the instructional resources on linked nodes before attempting this activity. After completing this hands-on activity you should:

- Understand creation and linking of singly-linked nodes.
- Understand creation and linking of doubly-linked nodes.

You will need the following source code files to complete this activity.

- `SinglyLinked.java`
- `DoublyLinked.java`

Note: This activity utilizes jGRASP Viewers, which are available in jGRASP, IntelliJ, and Eclipse.

Singly-linked nodes

1. Open `SinglyLinked.java` in jGRASP and compile it.
2. Set a breakpoint on line 18: `client.basicExamples();`
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `basicExamples()`.
5. Single-step to line 54: `n = new Node(1);`
6. Step over this statement and then open a viewer on `n`.
7. Step over each remaining statement in `basicExamples`, making sure you understand the effect of each statement. (You may want to step in to the calls to `length` and `contains`.)

Close the viewer and end program execution.

1. Clear any previous breakpoints you set in `SinglyLinked.java`.
2. Set a breakpoint on line 19: `client.add()`.
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `add()`.

5. Single-step to line 154: `System.out.println(toString(n));`
6. Open a viewer on `n` .
7. Click on the *Interactions* tab in the jGRASP Desktop.
8. Use *Interactions* to practice inserting the node referenced by `temp` at various locations in the pointer chain. (You'll have to repeat these steps each time you want to practice an insertion.)

Close the viewer and end program execution.

1. Clear any previous breakpoints you set in `SinglyLinked.java` .
2. Set a breakpoint on line 20: `client.delete()` .
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `delete()` .
5. Single-step to line 169: `System.out.println(toString(n));`
6. Open a viewer on `n` .
7. Click on the *Interactions* tab in the jGRASP Desktop.
8. Use *Interactions* to practice deleting various nodes in the pointer chain. (You may have to repeat these steps when you want to practice multiple deletions.)

Close the viewer and end program execution.

Doubly-linked nodes

1. Open `DoublyLinked.java` in jGRASP and compile it.
2. Set a breakpoint on line 18: `client.basicExamples();`
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `basicExamples()` .
5. Single-step to line 52: `m = new Node(2);`
6. Open a viewer on `n` .
7. Step over each remaining statement in `basicExamples` , making sure you understand the effect of each statement. (You may want to step in to the calls to `length` and `contains` .)

Close the viewer and end program execution.

1. Clear any previous breakpoints you set in `DoublyLinked.java` .
2. Set a breakpoint on line 19: `client.add()` .
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `add()` .
5. Single-step to line 139: `System.out.println(toString(n));`
6. Open a viewer on `n` .
7. Click on the *Interactions* tab in the jGRASP Desktop.
8. Use *Interactions* to practice inserting the node referenced by `temp` at various locations in the pointer chain. (You'll have to repeat these steps each time you want to practice an insertion.)

Close the viewer and end program execution.

1. Clear any previous breakpoints you set in `DoublyLinked.java` .
2. Set a breakpoint on line 20: `client.delete()` .
3. Start the debugger and wait until execution is paused at the breakpoint.
4. Step in to the call to `delete()` .
5. Single-step to line 157: `System.out.println(toString(n));`
6. Open a viewer on `n` .
7. Click on the *Interactions* tab in the jGRASP Desktop.
8. Use *Interactions* to practice deleting various nodes in the pointer chain. (You may have to repeat these steps when you want to practice multiple deletions.)

Close the viewer and end program execution.

Submission

The submission page for this activity asks you to apply your understanding of linked nodes to a problem and then submit for a grade.