Fachhochschule Nordwestschweiz
Hochschule für Technik

Software Architecture

Exercise – 4+1 View Model (Linked List)

BSc

Ingo Arnold

Exercise Opening
Overview

In this exercise Kruchten's 4 + 1 view model is utilized to abstract from a linked list code snippet in Java.

Exercise Agenda

- 4+1 View Model of a Linked List

4+1 View Model
Java source code

Take the following Linked List implementation as an example. Represent the architecture of this software in both the logical, development, and physical view.

As a use case, consider this scenario: elements “1”, “2”, and “3” are *added* to *List<String>*. Next the *contains* method is called with a parameter “2”. Finally the *plot* method is called.

```
public class Node <T> {
	private T element;
	private Node<T> next;

	public Node (T element, Node<T> next) {
		this.element = element; this.next = next;
	}
	public T getElement() {
		return element;
	}
	public Node<T> getNext() {
		return next;
	}
	public void setNext(Node<T> next) {
		this.next = next;
	}
}
```

4+1 View Model
Java source code

Take the following Linked List implementation as an example. Represent the architecture of this software in both the logical, development, and physical view.

```
public class List <T> {
	private Node<T> first = null;
	private Node<T> current = null;

	public void add(T element) {
		Node<T> n = new Node<T> (element, null);

		if(first == null) {
			first = current = n;
		} else {
			current.setNext(n); current = n;
		}
	}
	public boolean contains(T element) {
		Node<T> n = first;
		while (n!=null && !n.getElement().equals(element)) { n = n.getNext(); }
		return n != null;
	}
	public String plot() {
		String result = new String(); Node<T> n = first;
		while(n!=null) { result += n.getElement().toString() + " "; n = n.getNext(); }
		return result;
	}
}
```

4+1 View Model

Java source code

Take the following Linked List implementation as an example. Represent the architecture of this software in both the logical, development, and physical view.

```
public class Client {
	public static void main(String[] args) {
		List<String> list = new List<String>();

		list.add("1");
		list.add("2");
		list.add("3");

		System.out.println(list.contains("2"));
		System.out.println(list.plot());
	}
}
```

4+1 View Model
View Model Applied

How is this represented in the development view?

4+1 View Model
View Model Applied

How is this represented in the logical view?

4+1 View Model
View Model Applied

How is this represented in the physical view?

Questions