Anhängen eines neuen Listenknotens

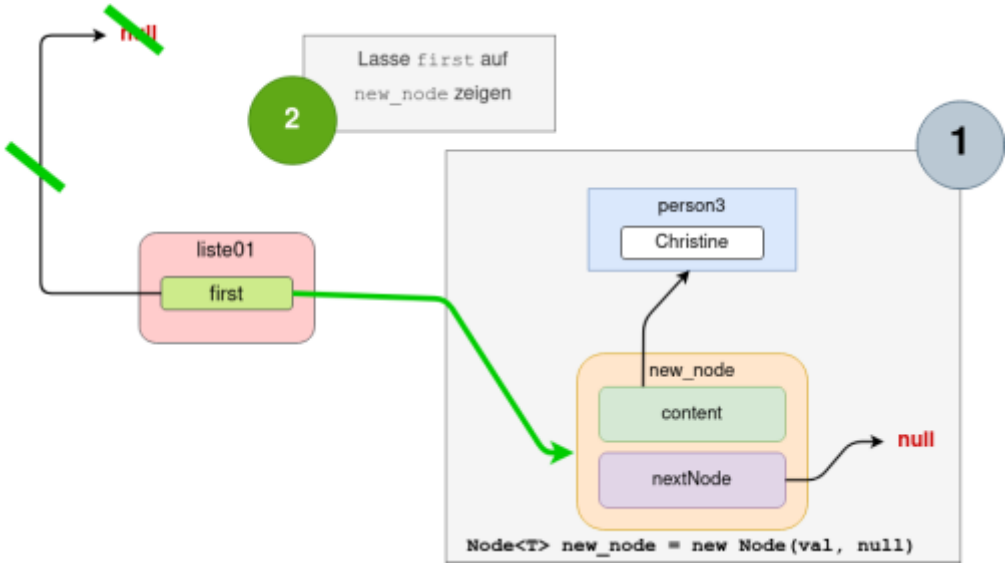
Beim Anhängen eines weiteren Knotens an die Liste sind zwei Fälle zu unterscheiden:

1. Die Liste ist leer
2. Die Liste ist nicht leer

Liste leer

Wenn die Liste Leer ist, ist der Vorgang schnell beschrieben, eine Veranschaulichung findet sich im Diagramm unten.

- Erzeuge eine neuen Knoten mit dem Nachfolger "null"
- Setze das Attribut first der Liste auf den neuen Knoten

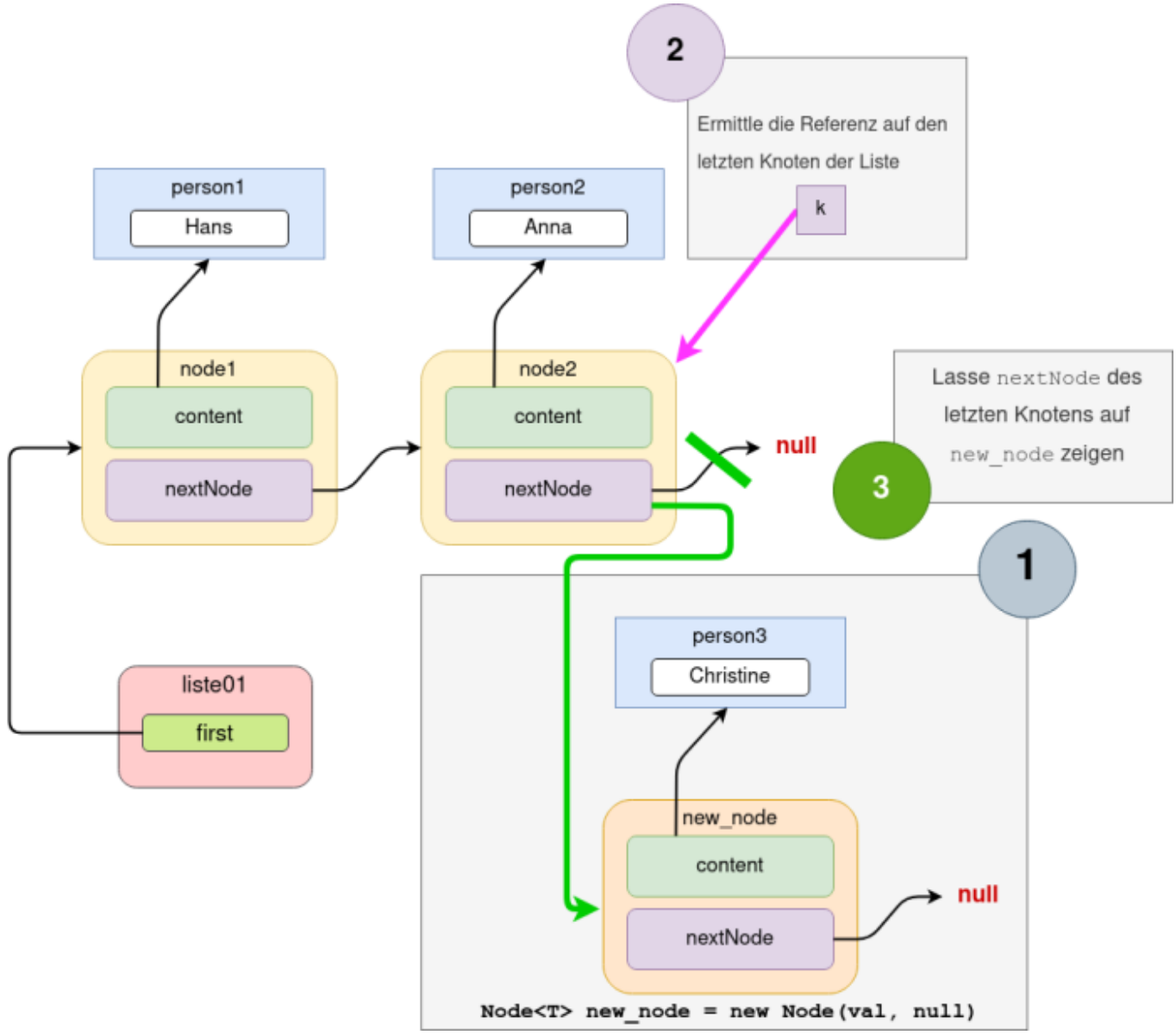


(A1) "append" für die leere Liste

- Implementiere in der Methode `append` den Teil des Codes, der an eine **leere** Liste einen neuen Knoten anhängt. Die Methode `isEmpty()` ist hilfreich.
- Teste deine Methode von Hand, indem du eine Liste für Integer Zahlen erzeugst und einen Knoten in die leere Liste einfügst.
- Überprüfe auch, dass deine zuvor implementierte Methode `isEmpty()` auch bei einer nicht leeren Liste einen korrekten Rückgabewert liefert.

Liste nicht leer

Das folgende Objektdiagramm veranschaulicht die Schritte, die beim Anhängen eines neuen Knotens an eine nicht leere Liste auszuführen sind.



(A2)

Schreibe als Merksatz stichwortartig nieder, was beim Anhängen eines neuen Knotens alles passieren muss.

Bewegen in der Liste

Ein konkretes Problem stellt sich noch in Schritt 2 des Ablaufs zum Einfügen eines neuen Knotens: Wie können wir uns durch die Liste bewegen, wenn das Listenobjekt selbst nur die Referenz auf den ersten Knoten kennt?

Hier gehen wir ähnlich vor wie bei der Erhöhung einer Zählvariablen:

Zähler	Liste

// Neuer Zähler
int i=0;
// Zählen bis 100
while (i < 100) {
  i++;
}
// Ende der Zahlenreihe erreicht,
i = 100
	// Knotenzeiger erzeugen, auf first setzen
	Node<T> n = first;
	// weitergehen bis zum Ende
	while(n.getNext()!= null) {
	  n = n.getNext();
	}
	// Jetzt zeigt n auf den letzten Knoten.

(A3) "append" für die nicht leere Liste

Implementiere den zweiten Fall der Methode append entsprechend deiner Vorüberlegungen. Teste deine Methode von Hand, indem du eine Liste für Integer Zahlen erzeugst und 3 Knoten mit aufsteigenden Werten einfügst. Wenn du das Listenobjekt inspizierst, solltest du deine Werte in der korrekten Reihenfolge vorfinden.

Lösungsvorschlag zur Methode "append"

```
    /**
     * Hängt einen neuen Wert hinten an die Liste an.
     * @param val Der anzuhängende Wert
     */
    public void append(T val) {

        // Auf jeden Fall: Neuen Knoten erzeugen
        Node<T> new_node = new Node(val, null);

        // Fall 1: Liste ist leer
        if (this.isEmpty()) {
            first = new_node;
        } else {
            //Fall 2: Liste ist nicht leer
            // Durch die Liste zum letzten Element wandern
            Node<T> n = first;
            while(n.getNext() != null) {
                n = n.getNext();
            }
            n.setNext(new_node);
        }

    }
```

<<< Zurück zur Übersicht: Liste in Java **oder** Weiter zur Länge und Wert auslesen >>>