

# Arbeitsunterlagen zu Prin im BG Praktische Informatik

Thomas Maul

V 0.1.0 - im Aufbau

Stand: 7. November 2025

## Inhaltsverzeichnis

<b>I. OOP</b>	<b>2</b>
<b>1. Verkettete Liste</b>	<b>2</b>
<b>2. Einfügen in die verkettete Liste</b>	<b>3</b>
2.1. Einfügen vorne . . . . .	3

# Teil I.

# OOP - Objektorientierte Programmierung

## 1. Verkettete Liste

Das Abspeichern von Daten in einem Array ist möglich, aber mit einigen Einschränkungen verbunden. Das Array muss beim Design des Programms festgelegt werden. Eine Größenänderung ist nachträglich nicht möglich<sup>1</sup>

Eine verkettete Liste bietet die Möglichkeit beliebig viele Einträge zu verwalten. Alle Einträge werden als Pointer auf dem Heap abgelegt. Die physikalische Reihenfolge ist für die logische Reihenfolge unerheblich.

In der Liste sind die Daten abgespeichert. Zusätzlich verweist ein Eintrag der Liste auf den nächsten Eintrag. Der letzte Eintrag hat als Nachfolger (nächster Eintrag) nullptr eingetragen.

Um die Liste zu verwalten, benötige ich (mindestens) zwei Klassen. Eine Klasse (im Beispiel ModelData) verwaltet die Liste. Sie ist auch die Schnittstelle im Programm zu den anderen Klassen (wenn vorhanden). Entry soll die Klasse sein, die die Daten beinhaltet. Diese Klasse erhält zusätzlich zu den Attributen der Daten (siehe Bild 1).

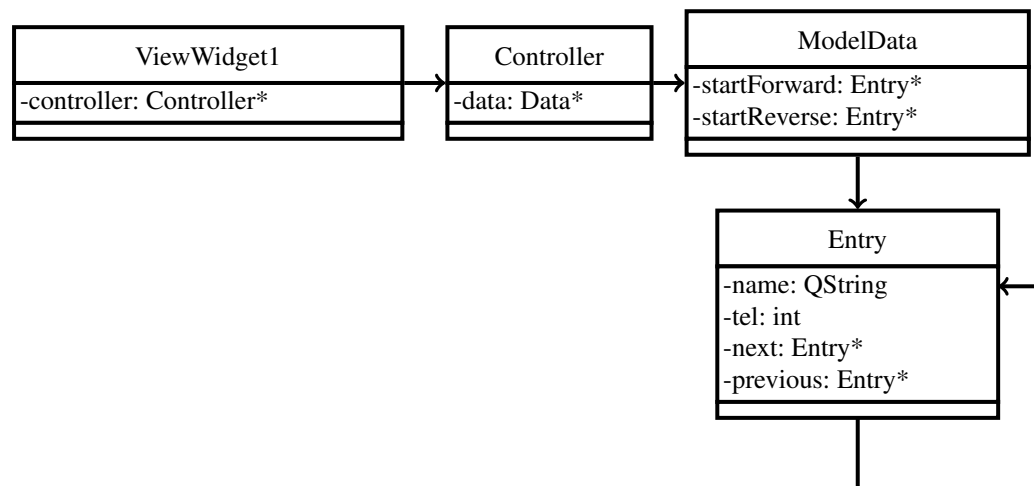


Abbildung 1: Klassendiagramm zu Programm, das die verkettete Liste verwaltet

Bild 2 zeigt das Programm zur Laufzeit in Form eines sogenannten Objektdiagramms. Hier mit einer leeren Liste - ohne Objekte zur Ablage von Daten in der Liste. Jedes Rechteck stellt ein Objekt dar. Der Name des Objekts steht in der ersten Zeile vor dem ,: ' dahinter folgt der Name der Klasse. Im Diagramm sind Objekte der Liste als Zahlen (1, 2, 3, ...) eingetragen. Unterhalb des Namens des Objekts stehen die Attribute mit dem aktuellen Wert. Das Objektdiagramm zeigt einen Zustand der Attribute und Objekte zu einem bestimmten Zeitpunkt.

In Bild 3 sind in der Liste drei Einträge vorhanden. Das Attribut startF stellt den Beginn der Liste dar. Die Adressen hier sind fiktiv. startR ist ein Zeiger auf das Ende der Liste. Dieser ist befüllt und wird nur bei einer doppelt verketteten Liste benötigt. Das Attribut next in den Objekten von Entry beinhaltet die Adresse, unter der das nachfolgende Element der Liste auf dem Heap gespeichert ist. Wenn kein Element folgt (hier bei Objekt 3) wird für next der Wert nullptr (0x00<sup>2</sup>) gespeichert.

<sup>1</sup>Bei einem dynamisch angelegten Array (als Pointer, mit new) ist die Auswahl der Größe zur Laufzeit möglich.)

<sup>2</sup>0x bedeutet, dass die Zahl hexadezimal dargestellt wird

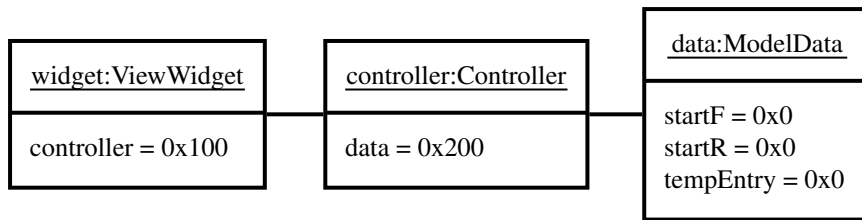


Abbildung 2: Objektdiagramm, leere Liste

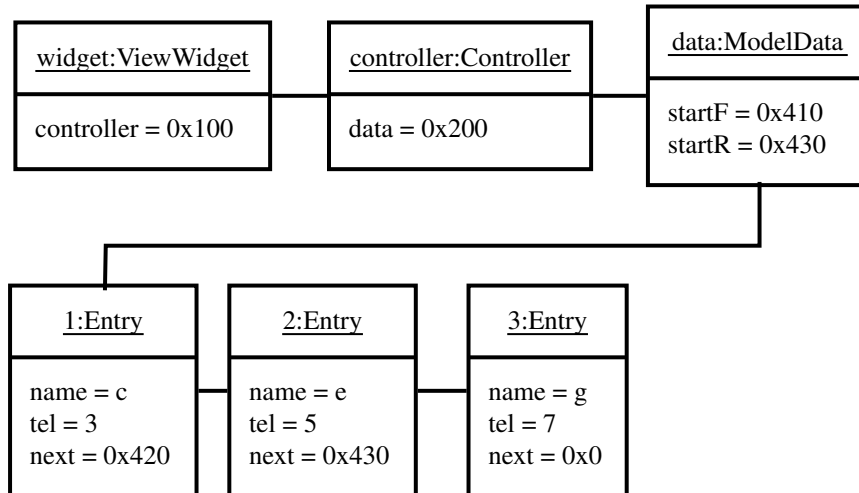


Abbildung 3: Objektdiagramm, drei Einträge in verkettete Liste

## 2. Einfügen in die verkettete Liste

In den folgenden Abschnitten betrachte ich das Einfügen in eine Liste, die in einer Richtung verkettet ist. Diese Form wird einfach verkettete Liste genannt. Einfach bezieht sich dabei auf die mögliche Richtung der Navigation.

In späteren Abschnitten befasse ich mich dann mit einer sogenannten doppelt verketteten Liste. Diese Form kann in zwei Richtungen (von vorne nach hinten und von hinten nach vorne durchlaufen werden).

### 2.1. Einfügen vorne

Wenn neue Einträge hinzugefügt werden sollen, müsste man bei einem Array alle Einträge, die nach der entsprechenden Position folgen, um eine Stelle nach rechts verschieben. Dies würde durch Kopieren der Daten erfolgen (vom Ende rückwärts bis zur betroffenen Stelle). Anschließend kann in dem freien Feld das Datum eingetragen werden.

Bei einer Liste wird ein neues Objekt mit `new Entry` angelegt (Bild 4).

Anschließend werden (Reihenfolge der Arbeitsschritte beachten!) die Zeiger aktualisiert (Bilder 5 und 6). In Bild 5 ist der Nachfolger von Objekt 4 auf Objekt 1 verlinkt. Somit kann ich das Objekt 1 über `startF` und zusätzlich über den temporären Zeiger `tempEntry` erreichen.

In Bild 6 ist dann der Zeiger `startF` angepasst, der `tempEntry` ist wieder auf `0x00` gesetzt. Damit ist das Einfügen am Anfang der Liste abgeschlossen.

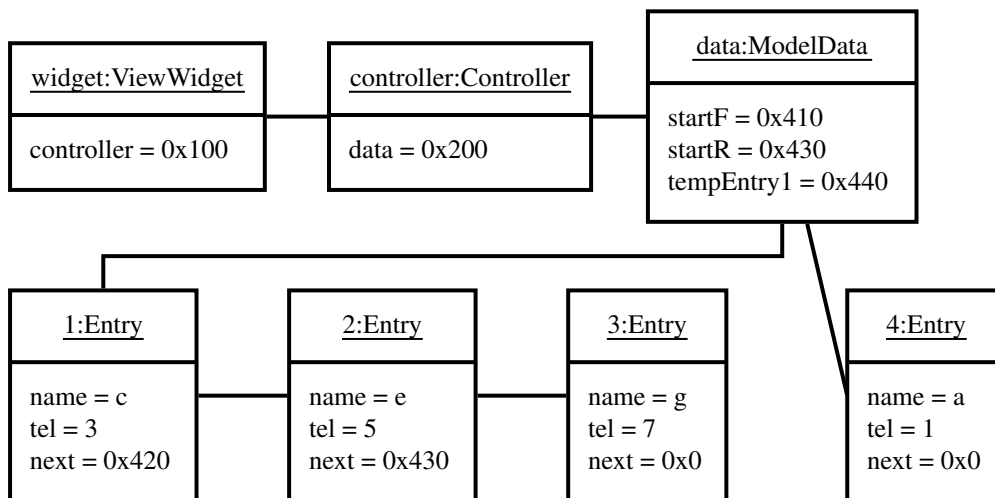


Abbildung 4: Objektdiagramm, ein Objekt vorne Einfügen, neues Objekt

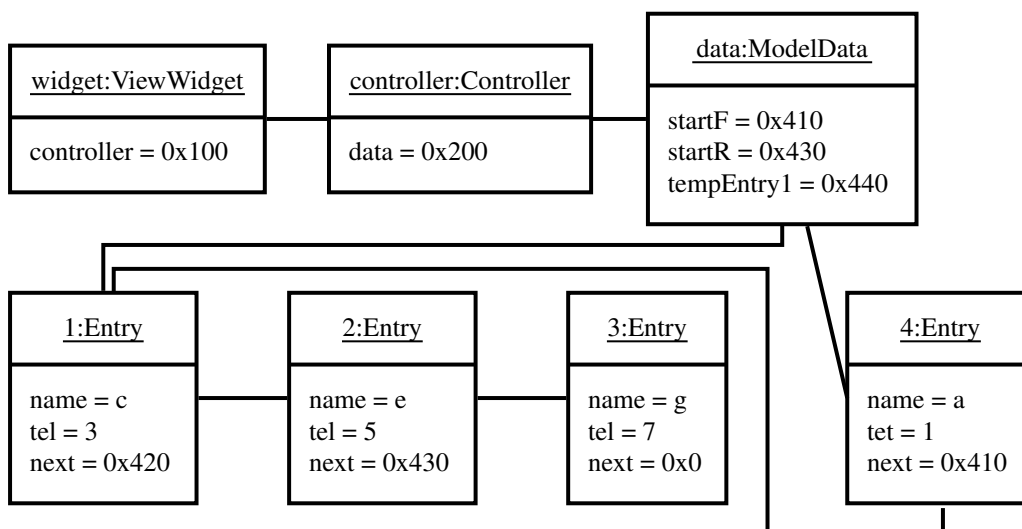


Abbildung 5: Objektdiagramm, ein Objekt vorne Einfügen, neues Objekt

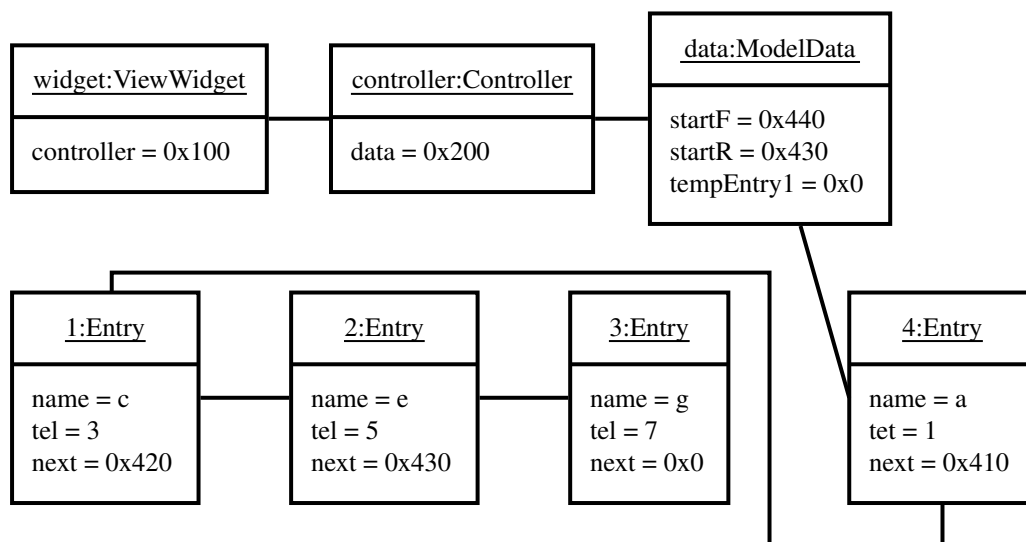


Abbildung 6: Objektdiagramm, ein Objekt vorne Einfügen, neues Objekt