Objektorientierte Programmierung

13 - Zusammenfassung und Wiederholung

Alexander Stuckenholz

Version 2022-05-04

Inhalt

HOCHSCHULE HAMM-LIPPSTADT

Die Objektorientierte Programmierung

HOCHSCHULE HAMM-LIPPSTADT

In den letzten Wochen haben wir uns mit einem neuen Programmierparadigma befasst.

- Der **Objektorientierten Programmierung**.

Im Vergleich zu reinen imperativen Programmiersprachen (z.B. C) verändert die objektorientierte Programmierung die Strukturierung von Softwaresystemen.

- Objekte bilden die Grundbausteine objektorientierter Softwaresysteme.
- Objekte verbinden Daten und Operationen zu einer Einheit.
- Objekte prägen Beziehungen untereinander aus, um gemeinsam Aufgaben zu lösen.

Die Objektorientierung macht sich vier Konzepte zu Nutze:

- Datenkapselung, Abstraktion, Vererbung und Polymorphie.

Dadurch kann im Entwurf die Kohäsion erhöht und die Kopplung reduziert werden.

- Ein wichtiger Beitrag, um wartbare und evolvierbare Software zu erhalten.

Datenkapselung

HOCHSCHULE HAMM-LIPPSTADT

Über Objektvariablen verwalten Objekte ihre eigenen Daten.

- Diese Daten repräsentieren den inneren Zustand des Objekts.
- Ein Objekt schützt diesen inneren Zustand über Methoden vor unerlaubtem Zugriff.

Dieses Vorgehen ist als **Datenkapselung** oder **Geheimnisprinzip** bekannt.

- Objekte grenzen sich gegenüber anderen Objekten ab (Module).

Dieses Vorgehen hat verschiedene Vorteile:

- Da nur die öffentliche Schnittstelle betrachtet werden muss, steigt die Übersichtlichkeit.
- Unerwünschte Interaktionen können vermieden werden (weniger Bugs).
- Ein Objekt kann durch ein Objekt mit einem anderen inneren Aufbau ersetzt werden, sofern sich die öffentliche Schnittstelle nicht ändert.

Abstraktion

HOCHSCHULE HAMM-LIPPSTADT

Programmierer lieben **Abstraktionen**.

- Eine höhere Programmiersprache wie C ist bereits eine solche Abstraktion.
- Wir müssen uns nicht mehr mit den Details eines konkreten Prozessors herumschlagen.
- In C können wir z.B. Funktionen einführen, um spezifische Fälle zu parametrisieren.

In der Objektorientierten Programmierung erhalten wir viele weitere Stilmittel, um Abstraktionen herstellen zu können.

- Mit **Klassen** und **Schnittstellen** schaffen wir neue Datentypen.

Eine Klasse dient als Vorlage für gleichartige Objekte.

- Sie definiert die Struktur und das Verhalten seiner Instanzen.

Schnittstellen hingegen definieren gemeinsame Funktionalität über mehrere Klassen hinweg.

Vererbung

Objekte können zur Laufzeit über Objektreferenzen Beziehungen untereinander aufbauen.

- Dadurch können beliebig komplexe Strukturen im Rechner abgebildet werden.

Darüber hinaus erlaubt es die **Vererbung**, eine **Ähnlichkeitsbeziehung** zwischen Klassen und Schnittstellen auszudrücken.

- In einer Vererbungshierarchie gibt das Basiselement (eine Klasse oder eine Schnittstelle) Eigenschaften an die Kindelemente weiter.

Bei Klassen sind dies Eigenschaften und Methodenimplementierungen.

- Bei Schnittstellen werden *lediglich* Methodensignaturen weiter gegeben.

Die Kindelemente können die geerbten Konzepte erweitern oder auch ändern.

Polymorphie

HOCHSCHULE HAMM-LIPPSTADT

Unter bestimmten Bedingungen können unterschiedliche Objekte als Ersatz füreinander genutzt werden.

- Eine solche *Ersetzbarkeit* kann auf zwei Arten garantiert werden:

Kindklassen bieten (mindestens) dieselben Methoden an, wie die Basisklasse.

- Die Objekte der Kindklasse können also als Ersatz der Objekte der Basisklasse genutzt werden.

Implementieren Klassen eine gemeinsame Schnittstelle, besitzen diese einen Satz gleichartiger Methoden.

- Dort, wo lediglich diese Methoden erwartet werden, können also die Objekte aller implementierender Klassen genutzt werden.

In beiden Fällen können Objekte aber unterschiedliches Verhalten zeigen.

- Dies wird als **polymorphes Verhalten** bezeichnet.

Kapitel

All diese Konzepte wurden in den letzten Kapiteln behandelt.

- Darüber hinaus wurde für C# auch gezeigt, wie diese in einer gängigen und modernen objektorientierten Programmiersprache angewandt werden.

Der Lernstoff war in die folgenden Kapitel aufgeteilt:

1. C# für C-Programmierer
2. Klassen und Objekte
3. Objekte zur Laufzeit
4. Schnittstellen und Aufzählungen
5. Objektbeziehungen
6. Vererbung und Polymorphie
7. Analyse und Analysemuster
8. Entwurf und Entwurfsmuster

Lernziele

Der erfolgreiche Teilnehmer der Veranstaltung, ist in der Lage, ...

- Klassen in C# zu implementieren.
- Klassen mit geeigneten Eigenschaften und Methoden auszustatten.
- Objekte zu erzeugen und zu verwalten.
- Objektbeziehungen aufzubauen.
- komplexe, objektorientierte Systeme so zu entwerfen, dass sie verschiedenen Qualitätskriterien genüge tragen.

Diese Fähigkeiten können natürlich nur dann entwickelt werden, wenn auch die Übungsaufgaben erfolgreich bearbeitet wurden.

Prüfung und Credits

HOCHSCHULE HAMM-LIPPSTADT

Die Prüfungsform ist eine Klausur.

- Wenn nicht anders angekündigt, wird die Klausurzeit 3h betragen.

Der Inhalt der Klausur ist eine Kombination der beiden Veranstaltungen **Objektorientierte Programmierung** und **Algorithmen und Datenstrukturen**.

- Der gesamte Lehrstoff ist klausurrelevant!
- Die Klausuraufgaben werden zu den Übungsaufgaben sehr ähnlich sein.

Bei erfolgreicher Teilnahme werde in ISD 8 ECTS Punkte vergeben.

- In ETR 6 ECTS Punkte.

Als Hilfsmittel ist eine einzelne DIN A4 Seite mit eigenen Notizen erlaubt.

- Doppelseitig beschrieben.

Zu guter Letzt...

HOCHSCHULE HAMM-LIPPSTADT

Vielen Dank für Ihre Aufmerksamkeit!

- Ich hoffe, ich konnte Ihnen etwas beibringen!
- Ich hoffe, das Ganze hat auch ein wenig Spaß gemacht!

Viel Glück bei der Klausur!

- Ich hoffe aber, dass Sie das Glück nicht brauchen!

Gute Erholung in der vorlesungsfreien Zeit!

- Wir sehen uns (möglicherweise) im nächsten Semester wieder!
- Da lernen wir (in ISD) neue tolle Dinge!

Quellen I