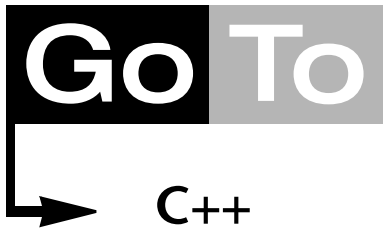
PEARSON Education

ADDISON-WESLEY

André Willms



C++ Programmierung

ADDISON-WESLEY

An imprint of Pearson Education

München • Reading, Massachusetts • Menlo Park, California
New York • Harlow, England • Don Mills, Ontario
Sydney • Mexico City • Madrid • Amsterdam

Objektorientierte Programmierung

4

Kapitelüberblick

Wir haben nun schon einige grundlegende Dinge über C++ besprochen. Bisher waren die Programme so einfach, daß sie bequem in der Hauptfunktion untergebracht werden konnten.

Damit die weitere Vorgehensweise der Problemlösung auch unter objektorientierten Gesichtspunkten ablaufen kann, ist es an der Zeit, sich mit der Philosophie der objektorientierten Programmierung zu befassen. In diesem Kapitel, welches die objektorientierte Sichtweise erst einmal von jeglicher Programmiersprache losgelöst betrachtet, geht es um folgende Themen:

- Was ist Abstraktion?
- Welche Möglichkeiten der Abstraktion gibt es?
- Was ist »prozeßorientierte Sichtweise«?
- Was ist »objektorientierte Sichtweise«?
- Was sind Klassen, Instanzen, Methoden, Attribute?
- Was ist Vererbung?
- Was ist Datenkapselung?

4.1 Einführende Gedanken

Bevor wir uns mit den Möglichkeiten der objektorientierten Programmierung[1] vertraut machen, sollten wir uns erst einmal über die Motivation Gedanken machen, die hinter der Entwicklung der OOP steht. Denn man könnte sich ja Fragen stellen wie: »Warum wurde die OOP überhaupt erfunden?« oder »Warum waren die Programmierer nicht mehr mit der alten Programmierweise zufrieden?«

Informatik

Doch zunächt wollen wir zu der Daseinsberechtigung der Informatik kommen. Was ist die Aufgabe eines Informatikers oder eines Programmierers, der sich ja die Erkenntnisse der Informatik zunutze macht? Eine komprimierte Antwort auf diese Frage wäre die folgende:

Die Aufgabe der Informatik ist die Bewältigung von Komplexität.

Komplexität

Doch wie wird die Komplexität bewältigt? Das der ganzen Informatik und darüber hinaus auch allen anderen Bereichen der Komplexitätsbewältigung zugrundeliegende Prinzip ist das der **Abstraktion**.

Das beste Mittel zur Bewältigung von Komplexität ist die Abstraktion.

1. Der Einfachheit halber wird im weiteren Verlauf der Ausdruck »objektorientierte Programmierung« durch die übliche Abkürzung »OOP« ersetzt.

Diese beiden Aussagen gemeinsam definieren dann den Aufgabenbereich der Informatik:

Die Informatik bewältigt Komplexität mit Hilfe von Abstraktion unter Einsatz von Computern.

4.1.1 Abstraktion

Was genau ist Abstraktion? Man abstrahiert zum Beispiel, wenn man Objekte unterschiedlicher Art, die aber alle in einem Teilbereich Gemeinsamkeiten haben, in einer Gruppe zusammenfaßt und diese Gruppe dann anstelle der einzelnen Objekte erwähnt oder benutzt. Jeder Mensch abstrahiert bei der täglichen Kommunikation mit seinen Mitmenschen. Man kann zwei Arten der Abstraktion grob unterscheiden:

Beginnen wir mit einem Beispiel der ersten Art. Angenommen, Sie wären am Wochenende draußen spazierengegangen. Wenn Sie dieses Erlebnis jemand anderem erzählen, werden Sie mit Sicherheit nicht sagen: »Ich bin am Wochenende durch einen wunderschönen Buchen-, Birken-, Eichen-, Eschen- und Lindenwald spazierengegangen.« Sie würden eher etwas sagen wie »Ich bin am Wochenende in einem wunderschönen Wald spazierengegangen«.

Diese Form der Abstraktion faßt Individuen verschiedener Arten zu einer übergeordneten Art zusammen, die die Gemeinsamkeiten der Individuen repräsentiert. In unserem Beispiel sind die Individuen die Bäume, die unterschiedlichen Arten angehören können. Die übergeordnete Art wären die Laubbäume, deren Gemeinsamkeit es ist, Laubblätter zu besitzen.

Und nun ein Beispiel zur zweiten Art: Angenommen, Sie hätten einen Zoobesuch hinter sich, und auf der Affeninsel wäre heute der Teufel losgewesen. Wenn Sie dies einem anderen Menschen erzählen, werden Sie sich wohl kaum folgendermaßen ausdrücken: »Der Affe mit der Narbe am Arm, der Affe mit dem dunklen Fell, das Affenweibchen mit den zwei Affenjungen und der Affe mit dem langen Fell waren heute ziemlich aufgedreht.« Vielmehr würden sie etwas der folgenden Art sagen: »Die Affen auf der Affeninsel waren heute ziemlich aufgedreht.«

Diese Form der Abstraktion faßt Individuen der gleichen Art zusammen. Obwohl jeder Affe auf der Affeninsel ein eigenständiges Individuum ist und jeder sein eigenes Aussehen, seine eigenen Eigenschaften und seine eigenen Erfahrungen besitzt, fassen Sie sie zu der Gruppe »Die Affen auf der Affeninsel« zusammen, weil die Tatsache, daß sie auf der Affeninsel im Zoo hausen, all diesen Affen gemeinsam ist.

Modell

Zusätzlich zum Prozeß der Abstraktion muß die Informatik sich noch mit folgendem Problem auseinandersetzen: Wie stellt man ein reales Problem so dar, daß es vom Computer verabeitet werden kann?

Die Informatik stellt ein reales Problem oder einen realen Sachverhalt mit Modellen dar.

Dieser Modellierung eines realen Problems sehen Sie sich bei nahezu jedem Programm, das sie schreiben wollen, gegenübergestellt. Nun kann man ein Problem aber auf verschiedene Weisen betrachten. Sehen wir uns diese Betrachtungsweisen genauer an.

4.2 Die prozedurale Sichtweise

Betrachten wir zuerst die konventionelle Vorgehensweise bei der Entwicklung eines Modells, nämlich die prozeduralen Sichtweise, bei der die Prozedur oder der Prozeß im Mittelpunkt steht.

Reale Objekte

Kommen wir zur ersten Frage: Was macht ein reales Objekt aus? Grundsätzlich kann man sagen, daß ein Objekt bestimmte Eigenschaften besitzt und daß es für das Objekt typische Prozesse gibt, die diese Eigenschaften manipulieren.

Da reale Objekte gewöhnlich eine sehr große Anzahl von Eigenschaften und Prozessen haben, müssen wir zur Darstellung eines realen Objekts die Abstraktion zu Hilfe nehmen. Von den unzählig vielen Eigenschaften eines Objekts berücksichtigen wir nur die, die für unser Problem wichtig sind. Dadurch entsteht ein Modell, welches die Realität vereinfacht beschreibt.

Sie sollten sich darüber im klaren sein, daß jedes entworfene Modell eine Vereinfachung der Realität sein muß, weil Ihnen nie alle Parameter bekannt sein werden. Und selbst wenn sie Ihnen theoretisch bekannt wären, würde die Komplexität der Realität nicht zu bewältigen sein.

Kartoffeln als Beispiel

Nehmen wir zum Beispiel »Kartoffeln«. Für unseren Zweck betrachten wir als Eigenschaften einer Kartoffelpflanze folgende Parameter: *Knollenanzahl*, *Knollendicke*, *Blütenanzahl* und die *Größe* der Pflanze. Als die Kartoffelpflanze manipulierenden Prozesse betrachten wir *blühen* und *wachsen*.

In der Realität sind die Prozesse fest an die Kartoffelpflanze gebunden. Unser Modell muß jedoch die Eigenschaften der Pflanze als Struktur zusammenfassen und jeden einzelnen Prozeß als eine Funktion definieren, die die entsprechenden Eigenschaften der Pflanze manipuliert. Der Prozeß *wachsen* verändert die Knollengröße, die Knollenanzahl und die Größe der Pflanze, wohingegen der Prozeß *blühen* nur die Blütenanzahl beeinflußt. Abbildung 4.1 stellt diesen Sachverhalt grafisch dar.

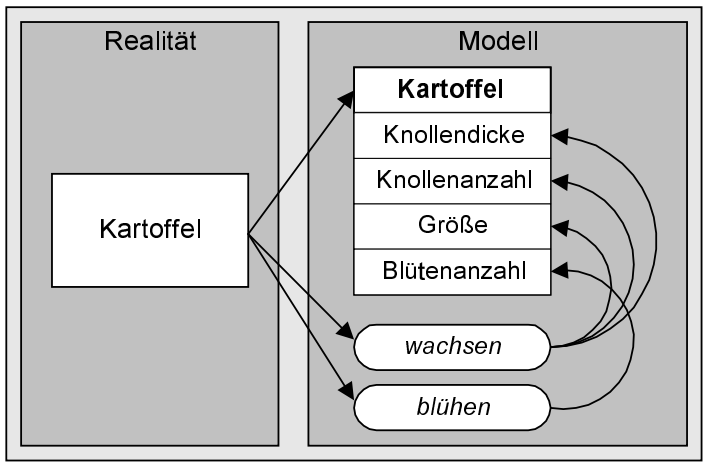


Abbildung 4.1: Prozeßorientierte Modellierung einer Kartoffelpflanze

Schweine als Beispiel

Als zweites Beispiel betrachten wir das Objekt *Schwein*. Wir geben dem Schwein die Eigenschaften *Größe*, *Gewicht* und *Sättigungsgrad*. Der Sättigungsgrad soll angeben, wie satt das Schwein oder wie stark sein Hunger ist. Als Prozesse definieren wir *wachsen*, welcher sich auf die Größe und den Sättigungsgrad auswirkt, und *bewegen*, der sich auf das Gewicht und den Sättigungsgrad auswirkt. Abbildung 4.2 stellt die Zusammenhänge grafisch dar.

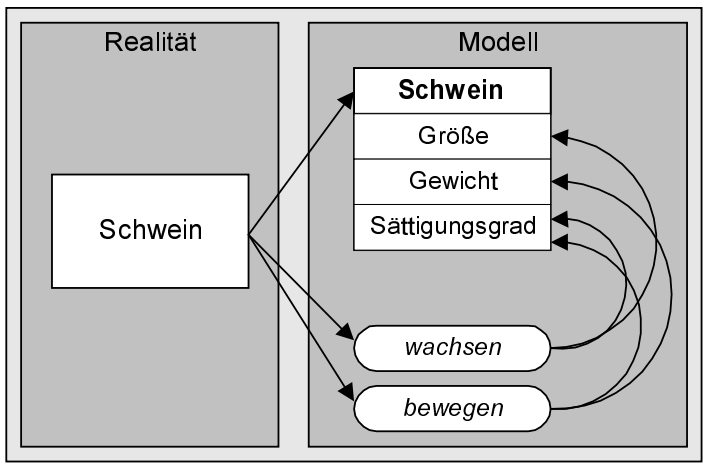


Abbildung 4.2: Prozeßorientierte Modellierung eines Schweins

Sie sehen, daß unser Modell eine sehr starke Vereinfachung der Realität ist. Diese Vereinfachung genügt unserem Vorhaben jedoch vollauf.

Der Prozeß »fressen«

Entwerfen wir nun einen Prozeß, der beide Objekte manipuliert. Dafür bietet sich *fressen* an: Das Schwein frißt eine Kartoffelknolle.

In der prozeßorientierten Sichtweise wäre dies eine Funktion, der wir Verweise auf ein Schwein und eine Kartoffelpflanze übergeben und die dann die entsprechenden Manipulationen der Kartoffelpflanze vornimmt. Betroffen sind die Eigenschaften *Knollenanzahl* der Kartoffel und *Gewicht* und *Sättigungsgrad* des Schweins. Abbildung 4.3 zeigt den Prozeß *fressen* eingebettet in unser bisheriges Modell.

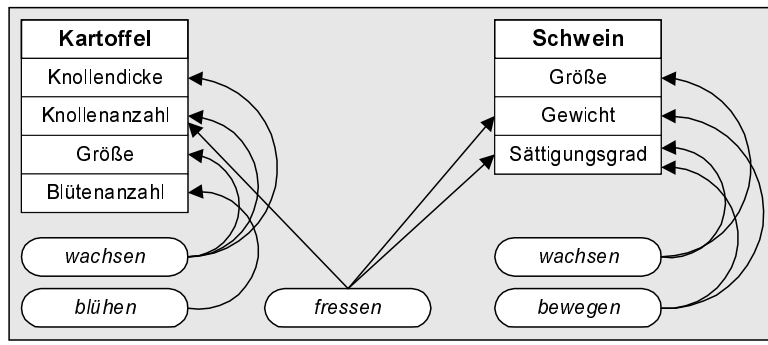


Abbildung 4.3: Prozeßorientierte Modellierung des Prozesses »fressen«

Bereits an diesem einfachen Beispiel läßt sich sehr schön erkennen, wie komplex die Beziehungen zwischen den Prozessen und den Eigenschaften der zu manipulierenden Objekte werden.

Und genau hier kommt nun der Gedanke ins Spiel, eine Sichtweise zu entwerfen, bei der die natürliche Ordnung der Zusammenhänge stärker berücksichtigt wird, um damit die Beziehungen zwischen Prozessen und Eigenschaften auf ein Minimum zu reduzieren.

4.3 Die objektorientierte Sichtweise

Rekapitulieren wir noch einmal: Die prozeßorientierte Sichtweise faßt die Eigenschaften eines Objektes in einer Datenstuktur zusammen und entwirft die Prozesse als Funktionen, die auf dieser Datenstruktur operieren, sie also manipulieren.

Nun ist man bei der objektorientierten Sichtweise davon ausgegangen, daß die auf den Objekten operierenden Prozesse nichts weiter sind als dy-

namische Eigenschaften des Objekts selbst. Dies bedeutet, daß das Objekt sowohl die Eigenschaften als auch die die Eigenschaften manipulierenden Prozesse besitzt, die sich auf diese Eigenschaften auswirken.

Diese Sichtweise ist viel natürlicher, denn Sie würden nie sagen, daß es etwas gibt, das die Kartoffel wachsen läßt. Nein, sie würden sagen, daß die Kartoffel selbst wächst. Das Kartoffelwachstum ist eine Eigenschaft der Kartoffel, genauso wie die Knollengröße. Schauen wir uns in Abbildung 4.4 die Modellierung von *Schwein* und *Kartoffel* mit Hilfe dieser Sichtweise einmal an.

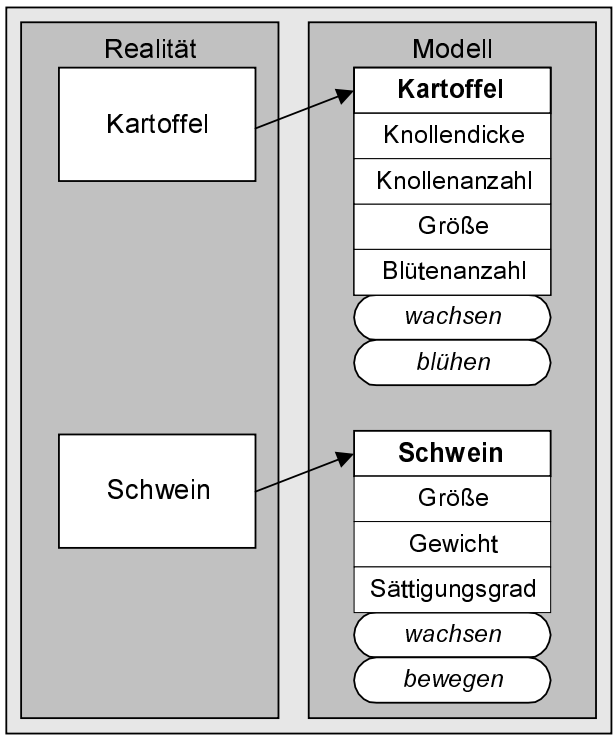


Abbildung 4.4: Objektorientierte Modellierung von »Kartoffel« und »Schwein«

Datenkapselung

Sie sehen die Vereinfachung auf den ersten Blick: Die Prozesse sind Eigenschaften des Objekts geworden. Dadurch findet die Manipulation der Eigenschaften durch die Prozesse »innerhalb« des Objekts statt. Ein Außenstehender wird nicht mehr mit der konkreten Manipulation konfrontiert, sondern sieht nur noch die manipulierenden Prozesse. Diese Form des »innerhalb Ablaufens« nennt man **Kapselung**.

Attribute

Kommen wir nun erst einmal zu ein paar Begriffsdefinitionen. Die Eigenschaften eines Objekts nennt man in der OOP **Attribute**.

Die Eigenschaften eines Objekts nennt man Attribute.

Und die zu einem Objekt gehörenden Prozesse heißen **Methoden**.

Methoden

Die Prozesse eines Objekts nennt man Methoden.

Die optimale Form der Datenkapselung ist gegeben, wenn der Benutzer eines Objekts keinen direkten Zugriff auf die Attribute mehr hat, sondern jede Art der Manipulation und jede Abfrage über Methoden des Objekts geschehen.

Die optimale Form der Datenkapselung ist gegeben, wenn jeglicher Zugriff auf Attribute eines Objekts nur noch über Methoden des Objekts möglich ist.

Schauen wir nun noch analog zur prozeßorientierten Sichtweise den Prozeß *fressen* in der objektorientierten Sichtweise an. Wie würden Sie die Methode *fressen* im Sprachgebrauch benutzen? Sie würden vermutlich sagen: »Das Schwein frißt die Kartoffel«. Also ist *fressen* eine Methode von *Schwein*, die Auswirkungen auf das Objekt *Kartoffel* hat. Abbildung 4.5 zeigt dies.

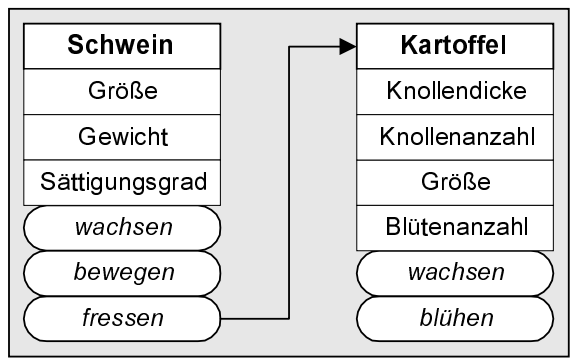


Abbildung 4.5: Objektorientierte Modellierung des Prozesses »fressen«

Wegen der Datenkapselung manipuliert die Methode *fressen* nicht direkt das Attribut *Knollenanzahl*, sondern wendet sich an das Objekt, welches eine Methode zur Manipulation von *Knollenanzahl* bereitstellen muß.

4.4 Objekte, Instanzen, Klassen

Kommen wir nun wieder zu ein paar Begriffsdefinitionen. Die Gesamtheit von Objekten einer Art bezeichnet man als **Klasse**.

Klassen

Eine Klasse faßt Objekte der gleichen Art zusammen.

Ein Objekt einer bestimmten Klasse ist eine **Instanz** dieser Klasse. Das Erzeugen einer Instanz von einer Klasse nennt man **Instanziierung**.

Instanzen

Objekte gleicher Art sind Instanzen der gleichen Klasse.

Erinnern Sie sich noch an die Abstraktion der zweiten Art mit den Affen auf der Affeninsel? Mit unseren neuen Begriffen gesprochen, wären die »Affen auf der Affeninsel« eine Klasse und jeder einzelne Affe eine Instanz dieser Klasse.

Genauso sind die einzelnen Schweine eine Instanz der Klasse *Schwein*. Die Attribute der einzelnen Schweine sind durch die Klasse vorgegeben. Lediglich die Werte der Attribute können und werden individuell verschieden sein. Abbildung 4.6 zeigt diese Zusammenhänge.

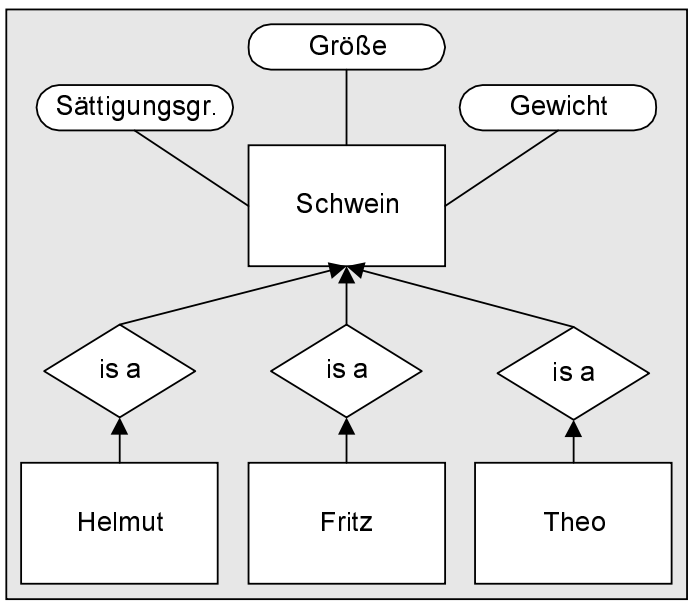


Abbildung 4.6: Die statischen Beziehungen zwischen »Objekt«, »Instanz« und »Attribut«

Die Objekte *Helmut*, *Fritz* und *Theo* sind eine Instanz der Klasse *Schwein*. Dies wird im Diagramm üblicherweise mit dem »is a«-Symbol gekennzeichnet, was auf deutsch soviel wie »ist ein(e)« bedeutet. Die Klasse *Schwein* hat die Attribute *Sättigungsgrad*, *Größe* und *Gewicht*. Darum besitzen die Objekte *Helmut*, *Theo* und *Fritz* als Instanz dieser Klasse diese Attribute ebenfalls.

4.5 Vererbung

Kommen wir nun noch auf die erste Abstraktionsart zu sprechen, die am Anfang des Kapitels vorgestellt wurde. Es ging dabei um das Zusammenfassen von Objekten verschiedener Arten zu einer übergeordneten Art, die die Gemeinsamkeiten der Objekte beinhaltet.

Vererbung

Dieses Abstrahieren entspräche in der OOP einer übergeordneten Klasse, die die Gemeinsamkeiten anderer Klassen beinhaltet. Im allgemeinen wird in der OOP jedoch zuerst eine Hauptklasse entworfen, von der dann andere Klassen abgeleitet werden. Dieses Ableiten nennt man **Vererbung**. Schauen wir uns dazu Abbildung 4.7 an.

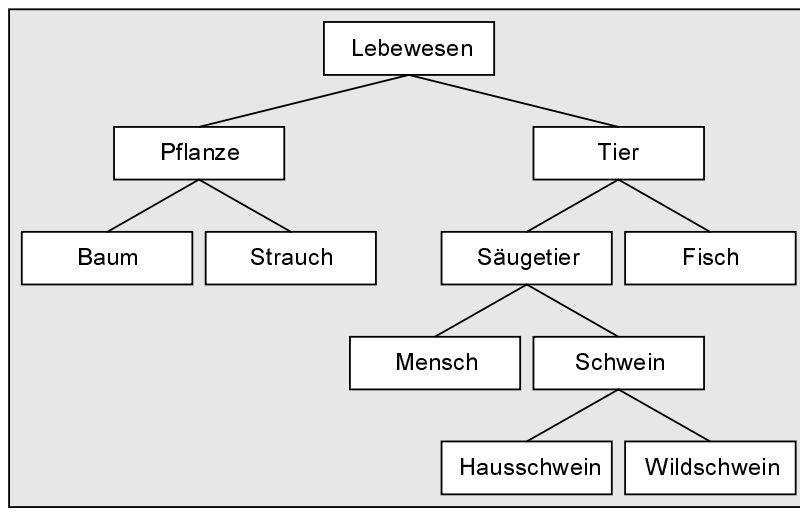


Abbildung 4.7: Beziehungen zwischen Klassen

Spezialisierung, Generalisierung

Wir sehen in Abbildung 4.7 sehr schön die einzelnen Vorgänge der Vererbung. Die Abbildung zeigt zwei Vorgänge – die Generalisierung und die Spezialisierung. Geht aus einer Klasse eine andere Klasse hervor, die nur einen Teilbereich der Ursprungsklasse abdeckt (z.B. vom Tier zum Säuge-

tier), dann nennt man dies **Spezialisierung**. Der umgekehrte Vorgang heißt **Generalisierung**. Üblicherweise wird der Vorgang der Spezialisierung als **Vererbung** bezeichnet. Schauen wir uns dazu Abbildung 4.8 an.

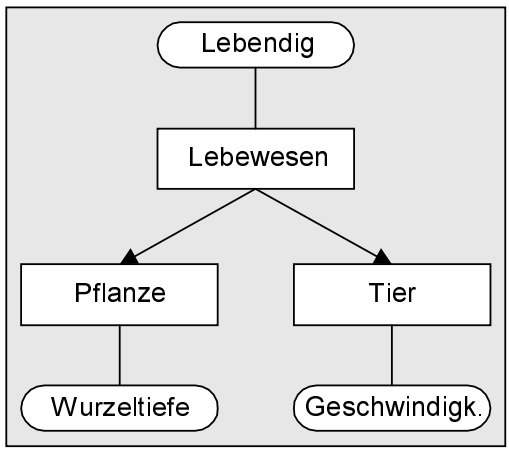


Abbildung 4.8: Der Vorgang der Vererbung

Wenn eine Klasse von einer anderen erbt, nennt man die Klasse, von der geerbt wurde, Basisklasse der erbenden Klasse.

Die erbende Klasse ist eine von der Basisklasse abgeleitete Klasse.

In Abbildung 4.8 bedeuten die Pfeile damit »ist Basisklasse von«. *Lebewesen* ist Basisklasse von *Pflanze* und von *Tier*. *Lebewesen* hat als Attribut *Lebendig*, welches aussagt, ob das entsprechende Lebewesen noch lebt oder schon tot ist. Da *Pflanze* und *Tier* von *Lebewesen* erben, besitzen sie automatisch das Attribut *Lebendig*. Darüber hinaus wird die abgeleitete Klasse *Pflanze* noch um das Attribut *Wurzeltiefe* und die abgeleitete Klasse *Tier* noch um das Attribut *Geschwindigkeit* erweitert. Die Attribute der einzelnen Klassen werden noch einmal in Abbildung 4.9 gezeigt.

Abbildung 4.9: Die Attribute der Klassen »Lebewesen«, »Pflanze« und »Tier«

Wichtig ist, daß zusätzlich zu den Attributen auch eventuell vorhandene Methoden vererbt werden.

Bei der Vererbung werden sowohl die Attribute als auch die Methoden vererbt.

Mehrfachvererbung

Eine weitere Form der Vererbung ist die sogenannte **Mehrfachvererbung**, bei der eine Klasse von mehreren Klassen erbt. Zum Beispiel könnte eine Klasse *Hund* sowohl von der Klasse *Säugetier* als auch von der Klasse *Vierbeiner* erben.

Wir wissen nun genug von der objektorientierten Sichtweise, so daß wir langsam damit beginnen können, dieses Wissen mit Hilfe von C++ umzusetzen.