Objektorientiertes Programmieren

In diesem Artikel sollst du lernen, was Objektorientierung ist und was du darüber wissen musst. Jeder Softwareentwickler, der mit dem Programmieren Geld verdient, muss wenigstens wissen, wie man objektorientiert denkt. Es gibt auch andere Methoden, wie zum Beispiel: strukturiertes Programmieren oder funktionales Programmieren. Aber dies ist momentan eine der beliebtesten Methoden, wie man Software erstellt: die objektorientierte Programmierung.

Das Wort Objektorientierung

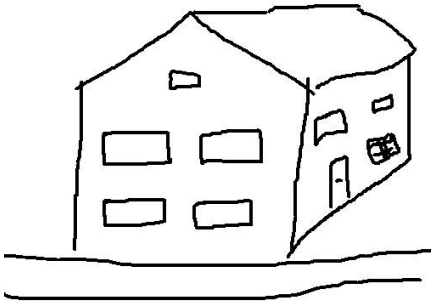
Wenn man sich das Wort Objektorientierung ansieht, merkt man schnell, dass hier zwei Wörter in einem stecken. Das Wort Objekt und das Wort Orientierung. Übersetzt heißt das: Man konzentriert sich auf Objekte. Alle Überlegungen, die der Programmierung dienen, *orientieren* sich an **Objekten**.

Was ist ein Objekt?

Schau dich mal in deinem Zimmer um. Alles, was du siehst, ist ein Objekt: die Tastatur, dein Schreibtisch, die Zimmertür, deine Deckenlampe, dein Bleistift. Das alles sind Objekte. Objekte, die man anfassen und sehen kann. Damit wir Entwickler unsere Software leicht verständlich aufbauen können, versuchen wir, diese Objekte auch in unserer Software nachzubilden. Was heißt das denn?

Dafür möchte ich ein kleines Beispiel mit dir durchgehen. Denk mal an das Haus, in dem du gerade bist. Nimm ein Blatt Papier und versuch mal, eine Skizze von diesem Haus zu zeichnen. Du brauchst die Zeichnung niemandem zeigen, aber zeichne das Haus auch, wenn du nicht gut im Zeichnen bist. Wichtig ist, dass man alle Fenster und Türen einer Hausseite sieht. Ob es aus Stein oder aus Holz ist, interessiert in der Zeichnung nicht. Sobald du das Haus gezeichnet hast, lies weiter…

Und wie sieht dein Haus aus? So etwa?



Du hast gerade objektorientiert gearbeitet! Du hast dich auf Objekte aus deiner realen Welt konzentriert und hast ein Modell davon erstellt. Deine Skizze zeigt nicht alle Details eines echten Hauses, aber jeder Mensch würde sofort erkennen, dass es sich um ein Haus handelt.

Von der realen Welt in ein Modell

Das Hausbeispiel zeigt dir, wie du bei der Objektorientierung denken musst. Um das Ganze noch ein bisschen zu vertiefen, zeige ich dir noch ein Beispiel. Nehmen wir ein Auto. Nein, du musst jetzt kein Auto zeichnen. Aber denk mal an ein

rotes Auto und versuch mal, an alle Details zu denken, die in so einem Auto stecken. Das Aussehen ist noch relativ einfach: Es hat 4 Räder, eine Karosserie, ein paar Fenster, Türen, Außenspiegel sowie Vorder- und Rückleuchten. Aber das ist ja eigentlich noch nicht alles. Wenn man genauer hinguckt, sieht man alleine bei einem einzelnen Rad: Reifen, Felge, Radschrauben, Antrieb, Bremsscheibe und einen Bremssattel. Und schaut man noch genauer hin, also zum Beispiel auf eine Radschraube, hat man auch hier wieder viele Details wie: Schraubenlänge, Schraubendicke, Gewinde, Mutter und das Material. Und wir reden hier nur von einer Art Schraube am ganzen Auto! Dabei haben wir noch nicht einmal über Hersteller und Modell des Wagens gesprochen! Puh… Ok, worauf will ich hinaus?

Man kann sich bei der Objektorientierung in Details verlieren. Und wenn wir Software so detailliert schreiben würden, wie wir gerade das Auto analysiert haben, dann würden unsere Programme nie fertig werden. Aus diesem Grund konzentrieren wir uns nur auf die Details, die uns in der Software interessieren. Um auf das Beispiel zurückzukommen: Wir erstellen ein kleines Modell von unserem Auto. Und dabei konzentrieren wir uns nur auf die Details, die für uns wichtig sind.

Wir nehmen wieder das große Auto, aber erstellen nun ein Modell davon. Also stell dir doch mal das große Auto als kleines Spielzeugauto vor. Schau mal, wie viele Details plötzlich

verschwinden. Klar: Wir haben immer noch 4 Räder, aber um Schrauben brauchen wir uns jetzt nicht mehr kümmern. Sehr wahrscheinlich wird in dem kleinen Spielzeug auch kein funktionsfähiger Motor verbaut sein. Beim Spielzeugmodell des Autos geht es nur darum, dass es so aussieht wie das Original - und dass es 4 drehbare Räder hat, damit man damit spielen kann. Und so macht man es auch in der Software. Man kümmert sich nur um die Details, die für das Funktionieren deines Programmes wichtig sind.

Nur die Details sind wichtig, die du wirklich brauchst

Halten wir fest: Wir haben Objekte (Autos), die so viele Details (Räder) haben können, dass die einzelnen Details (das Rad) wiederum Objekte sein könnten. Allerdings interessieren uns nicht alle Details. Dass ein Rad mit Schrauben (wiederum Objekte) befestigt wird, ist bei einem Spielzeugauto völlig egal. Also können wir diese Objekte einfach ignorieren. Aber woher weißt du, welche Details für was überhaupt wichtig sind? Wer sagt dir, dass die Radschraube nicht wichtig für dein Programm ist? Es hängt davon ab, was du später mit diesem Objekt machen möchtest. Dein großes Auto ist natürlich sehr komplex, denn es muss den täglichen Straßenverkehr meistern. Möchtest du nur ein Spielzeugmodell davon haben, fallen viele Details weg. Selbst wenn man es in derselben Größe bauen würde wie das Original, würde es den Anforderungen der Straße nicht gerecht werden. Ohne Motor müsstest du es schieben. Nehmen wir ein anderes Beispiel vom Auto, dein kleines Spielzeugmodell reicht dir nicht aus. Du möchtest dieses Spielzeugmodell fernsteuern… Und schon kommen neue Details dazu. Du brauchst immer noch 4 Räder, und das Auto sollte ebenfalls aussehen wie das Original. Was brauchst du nun an Details damit aus deinem Spielzeug ein ferngesteuertes Auto wird? Mach dir mal kurz Gedanken darüber, dann lies weiter.

Details beschreiben

Um aus deinem Spielzeug ein ferngesteuertes Auto zu machen, brauchst du nicht nur die Fernbedienung: Dein Modell braucht einen Motor, der die 4 Räder drehen kann. Eine Lenkung, damit das Auto nach links oder rechts gesteuert werden kann. Eine Art Funkmodul, welches die Befehle der Fernbedienung annimmt. Grob beschrieben würde dies so passen. Wenn du an eine ähnliche Lösung gedacht hast, bist du goldrichtig. Auch hier kann man sich wieder in Details verlieren. Welche Art von Motor und Antrieb wird gebraucht? Woher bekommt der Motor seine Energie? Akku oder Benzin? So tief wollen wir nicht gehen. Es geht einfach darum, dass du lernst, wie man Objekte ansieht und analysiert - und zwar immer bis auf die Ebene, die notwendig ist. Wenn du nur mit dem ferngesteuerten Auto rumfahren willst, ist dir nur wichtig, dass der Akku voll ist, dass das Auto funktioniert und du die Fernbedienung benutzen kannst. Wenn du der Modellbauer bist, dann interessierst du dich wiederum für die Details in dem Modell. Wenn du Hersteller für ferngesteuerte Autos bist, dann interessieren dich sogar noch viel mehr Details. Wichtig ist für dich, dass du lernst, Objekte zu analysieren, und dass du deren Details herausfinden kannst, bis zu der Ebene, die dich interessiert bzw. die notwendig ist. Wenn es für dich notwendig ist, in den Motor deines Autos zu sehen, benötigst du dafür viel Detailwissen. Details an Objekten nennt man auch Eigenschaften. Denn nicht immer ist ein Detail auch noch ein Objekt. Dein Auto hat 4 Räder, ein Rad ist ein Objekt. Allerdings gibt es 4 davon. 4 Räder sind eine Eigenschaft deines Autos. Und dies ist eine Eigenschaft, die auf (fast) alle Autos zutrifft.

Objekte, Eigenschaften und Funktionen

Die Eigenschaft, dass ein Auto 4 Räder hat, ist an sich ja kein Objekt. Sondern es gibt 4 Objekte, die zusammen diese Eigenschaft bilden. Aber es gibt noch eine kleine Besonderheit in den Details deines Autos. Denn deine Objekte sind zwar keine Lebewesen, aber sie können etwas tun. Ein Auto kann fahren. Und auch hier gelten wieder die gleichen Regeln für die Details. Ob dich interessiert, was alles im Auto passiert, damit es fährt, hängt wieder davon ab, was dein Objekt letztendlich machen soll. Ein Spielzeugauto muss angeschoben werden, es kann nicht fahren. Das ferngesteuerte und das echte Auto können aber alleine fahren. Sie können auch beschleunigen, lenken und auch bremsen. Die Türen kann man öffnen, schließen. Fenster auf- und zumachen usw. Also auch hier nicht in den Details verlieren. Das, was ein Objekt also machen kann, nennt sich in der Softwareentwicklung Funktion oder Methode. Denn in deinem Programm würdest du deinem Objekt Befehle geben und damit genau diese Funktionen benutzen. Ich erkläre dir das Ganze gleich anhand eines Rennspiels. Dort wirst du auch ein Auto brauchen. Kommt gleich 😉

Was du dir hier also merkst, ist Folgendes: Objekte haben Eigenschaften und Methoden. Und Objekte können aus anderen Objekten bestehen.

Die Gedanken, die du dir beim Modell des Autos gemacht hast, wirst du dir in Zukunft immer bei der Programmierung machen. Wir Softwareentwickler denken immer in Objekten; aus diesem Grund heißt es Objektorientierung. Man orientiert sich an Objekten. Alles, was irgendwie komplex ist, ist ein Objekt. Ich hoffe, dieser Begriff ist dir spätestens jetzt nicht mehr fremd.

Objekte, die man nicht anfassen kann

Wir Softwareentwickler beschreiben alles gerne in Objekten aus der realen Welt. Das beste Beispiel dafür ist die E-Mail. Mail ist zu Deutsch ein Brief, das E steht für „electronic“, also elektronisch. Eine E-Mail ist also nichts anderes als ein elektronischer Brief. Ob hier wohl jemand objektorientiert gedacht hat? Einen Brief kann man anfassen, eine E-Mail naja erst wenn man sie ausdruckt, aber darum geht es hier nicht.

Ich weiß, der Artikel ist schwer genug, allerdings ist dieser Abschnitt der wichtigste für die Softwareentwicklung. Nicht alle Objekte, die man in der Softwareentwicklung findet, kann man auch in der Realwelt anfassen. Du wirst immer wieder Objekte finden, die etwas komplizierter aufgebaut sind, und es dauert etwas, bis man Sie versteht. Zum Beispiel ein Mausklick. Ist dies für dich ein Objekt? Eigentlich nicht, es ist eine Art Ereignis, und das kann man nicht anfassen, das passiert einfach. Allerdings kann man mit Ereignissen arbeiten. Wenn dein Wecker morgens klingelt, ist das Klingeln zum Beispiel ein Ereignis - und jeder weiß, was er dann tun muss: Man muss aufstehen. Auch wenn man das Ereignis nicht direkt anfassen kann, würde man beim Programmieren dieses als Objekt verwenden. Denn auch dein Ereignis „Wecker klingelt“ hat viele Details, die man beachten kann. Erstens wäre da die Uhrzeit, wann das Ereignis stattfindet. Und ohne Wecker kein Klingeln; also auch hier wieder ein Objekt. Dieses Objekt (den Wecker) könnte man wiederum anfassen. Wir Softwareentwickler nennen alles, was irgendwie eine Struktur oder Details aufweist, Objekt, unabhängig davon, ob es sichtbar oder unsichtbar ist. Ich habe es dir versprochen, lass uns mal anfangen, ein Rennspiel objektorientiert zu sehen. Wenn du es wirklich lernen willst, überleg dir doch mal, welche Objekte in einem Rennen so auftauchen (auch nicht sichtbare). Und versuche dabei natürlich, daran zu denken, dass wir hier ein Computerspiel schreiben wollen. Also ist es wichtig, an die Objekte zu denken, die du bei einem Computerspiel so brauchst. Nimm dir am besten mal 10 Minuten Zeit dafür.

Wenn du absolut keine Idee hast, schau dir mal ein paar Ausschnitte aus Formel-1-Rennen an oder versuche mal, ein Rennen auf einem Blatt Papier zu zeichnen. Wie würde dein Spiel auf dem Bildschirm aussehen?

Objektorientiert denken: Wir modellieren ein Rennspiel

Lass uns einmal schauen, wie ein Rennen so aufgebaut ist. Okay, wir haben eine Rennstrecke, und diese hat Geraden und Kurven. Der Belag der Strecke interessiert uns erstmal nicht. Für jede Gerade und Kurve gibt es aber ein paar Details, die interessant sind, z.B. die Länge der Gerade und der Winkel der Kurve. Zusammengesetzt ergeben diese die Rennstrecke. Dann haben wir die Autos. Jedes Auto muss fahren und lenken können. Wie das Auto aussieht, hängt allerdings davon ab, wie das Spiel aufgebaut ist. Haben wir eine Autorennbahn von Carrera im Spiel, dann sind es kleine Automodelle. Wollen wir ein 3D-Spiel machen, dann brauchen wir viele Details über das Aussehen. Haben wir ein 2D-Rennspiel, bei dem wir das Spielfeld von oben sehen, brauchen wir das Aussehen des Autos von oben. Hier hängen die Details wieder einmal davon ab, wie das Ergebnis aussehen soll. Zurück um Rennen: Das Rennen läuft über ein paar Runden, und die Autos fahren über die Rennstrecke. Das Auto, das zuerst über die Ziellinie fährt, hat gewonnen. Auch in diesem Satz verstecken sich Objekte. Die Ziellinie ist definitiv wichtig, um herauszufinden, wer der Gewinner ist. Dies wäre an sich ja nicht das Auto, sondern der Fahrer des Autos. Da wir hier ein Videospiel schreiben wollen, haben wir auch keine echten Fahrer, sondern Computerbenutzer. Hier verschwimmt nun langsam das echte Rennen zum Videospiel. Immer wieder stoßen wir mit unseren Objekten an Details, die zum Computer gehören. Also fangen wir an, auch die Objekte zu finden, die wir in einem Rennspiel erwarten. Das heißt: Die Rennstrecke muss irgendwie auf dem Bildschirm gezeichnet werden. So auch die Autos. Die Autos müssen sich über den Bildschirm und die Rennstrecke bewegen. Wie hängt wiederum davon ab, ob es ein 2D-Spiel oder 3D-Spiel ist. Die Autos müssen beschleunigen, bremsen und lenken können. Dafür ist in unseren Computer kein Gaspedal zuständig, sondern die Tastatur oder ein Gamepad. Also überlegen wir weiter: Um das Auto zu beschleunigen, muss der Nutzer eine Taste auf der Tastatur drücken. Was würdest du sagen? Ist dies ein Detail, das wichtig für dein Computerspiel ist? Genau. Du merkst, dass dein Auto-Objekt sich ziemlich von dem echten Auto unterscheidet. Es ist nicht wichtig, dass es 4 Räder hat. Hier ist wichtig, welche Tasten der Tastatur es beschleunigen. Was passiert eigentlich bei der Beschleunigung? Die Geschwindigkeit wird erhöht. Diese Information könnte wichtig sein in einem Rennspiel, zum Beispiel, wenn man die Geschwindigkeit anzeigen möchte. Das sind jetzt nur ein paar Details, die zum Rennspiel gehören. Ich wollte dir hier zeigen, wie viel dazugehört, wenn man anfängt, ein Computerspiel objektorientiert zu modellieren.

Wichtig in dieser Übung ist, dass du lernst, in Beispielen zu denken, um deine Objekte, Eigenschaften und Methoden zu erhalten. Erst wenn du ein gutes Beispiel durchdenkst, wirst du sehen, welche Objekte es gibt, wie sie zusammenhängen und welche davon du wirklich benötigst. Mit dem Objekt modellierst du deine Software. Beim Rennspiel merkst du, dass man sich selbst dort schnell in

Details verlieren kann. Wenn du dich also schon einmal gefragt hast, warum man nicht in einem Rennspiel einfach aus dem Auto aussteigen kann, dann versuche doch mal, mit Objekten zu modellieren, was eigentlich alles passieren muss. Man sollte sich vorher überlegen, was das Programm eigentlich können soll, damit man die Objekte genau darauf ausrichten kann.

Objekte im Computer

Wir haben jetzt über Objekte in der echten Welt gesprochen und bereits ein paar Ideen für Objekte in einem Rennspiel gesehen. Aber wie kommen unsere Objekte nun in den Computer? Dafür gibt es in jeder Programmiersprache bestimmte Sprachelemente. In **Javascript** zum Beispiel heißen diese Dinge tatsächlich Objekte. Allerdings bestehen Objekte im Programmcode aus Variablen. Für einfache Objekte kann man Variablen nehmen. Und unser Computer kennt ein paar Variablentypen. Damit kann man Text und Zahlen abbilden. Wenn wir also unsere Geschwindigkeit im Objekt notieren wollen, nehmen wir eine Zahlvariable mit dem Namen „geschwindigkeit“ (weil es eine Zahl speichern soll). Wenn wir die Maximalgeschwindigkeit nehmen, ist dies auch eine Zahl. Wenn wir das Auto aber auf dem Bildschirm darstellen wollen, brauchen wir ein Objekt, das uns hilft, auf den Bildschirm etwas zeichnen zu können. Die kleinste Einheit im Computer ist die Variable. Erst wenn deine Variable zu komplex wird, brauchst du mehrere Variablen und kannst diese dann zu Objekten zusammenfassen.

Fazit

In diesem Artikel hast du gelernt, was Objektorientierung ist. Alles, was Details hat und irgendwie strukturiert ist, ist ein Objekt. Egal, ob man es anfassen kann oder nicht. Worauf man genau achten muss bei der Programmierung, sollte hier nur angerissen werden. Mich würde jetzt noch interessieren, welche Objekte du bei deinem Rennspiel so gefunden hast.