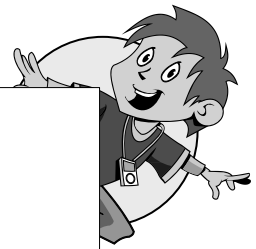


Mac- Programmierung

FÜR **KIDS**





Vorwort

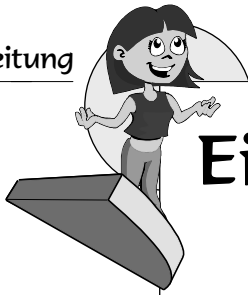
Wenn du dieses Buch gekauft hast oder es vielleicht gerade kaufen willst, hat dich möglicherweise auch die Leidenschaft des Programmierens gepackt. Eventuell bist du aber auch nur neugierig und möchtest wissen, was du mit deinem Computer alles machen kannst, wenn du nicht nur fertige Programme von anderen benutzt.

Zu behaupten, das Programmieren auf dem Mac zu erlernen wäre leicht, ist sicher nicht die ganze Wahrheit. Genau wie bei einer Fremdsprache, die man lernen will, ist auch hier aller Anfang schwer. Aber je mehr man weiß, umso leichter geht es voran – und umso mehr Spaß macht es.

Dieses Buch und die Beispiele darin sollen auch Spaß machen. Zwar wird dir vieles neu und unbekannt vorkommen, besonders dann, wenn du noch nie in deinem Leben programmiert hast, aber mit vielen kleinen Schritten wirst du immer mehr lernen und immer mehr erfahren. Doch selbst wenn du irgendwann jahrelang deine eigenen Programme geschrieben hast, wirst du immer noch etwas Neues lernen können. Das ist das Tolle am Programmieren, aber auch die Herausforderung.

Mit deinem Mac und dem Betriebssystem OS X steht dir eine Plattform zur Verfügung, auf der du viele eindrucksvolle Dinge schaffen kannst. Dinge, von denen Softwareentwickler vor wenigen Jahren noch geträumt haben. Da dir Apple alle Werkzeuge, die du zur Entwicklung eigener Programme brauchst, kostenlos zur Verfügung stellt, gibt es keinen Grund, warum nicht auch du programmieren solltest.

Nur keine Angst. Es macht Spaß!



Einleitung

Der Aufbau dieses Buchs

Die ersten Beispiele in diesem Buch geben eine kurze Einführung in das Programm Xcode und in die Programmiersprache Objective-C. Xcode ist die Entwicklungsumgebung für das OS X Betriebssystem und somit für dich das wichtigste Werkzeug, wenn du auf dem Mac programmieren möchtest. Mit Xcode werden deine Eingaben in eigenständige, lauffähige Programme umgewandelt. Objective-C ist die Programmiersprache, die in Xcode verwendet wird, und auch die Entwickler von Apple verwenden Objective-C für ihre Programme. Wenn du schon Erfahrung mit anderen Programmiersprachen wie zum Beispiel Visual Basic oder Java hast, wird dir manches bekannt vorkommen. Andere Dinge sind hingegen typisch für Objective-C und man findet sie in anderen Sprachen eher selten oder gar nicht.

Der überwiegende Teil dieses Buches beschäftigt sich aber mit der Programmierung von grafischen Benutzeroberflächen für Programme, mit Fenstern, Schaltflächen und vielen anderen Dinge, die dazugehören. Grundlage dieser Programme ist das sogenannte »Application Kit Framework«, eine von Apple zur Verfügung gestellte Sammlung von Bausteinen für grafische Anwendungen.

Besondere Vorkenntnisse für dieses Buch brauchst du nicht. Wenn du deinen Mac gut bedienen kannst, dir auch die Systemeinstellungen nicht unbekannt sind, sollte es keine Probleme geben. Die ersten Beispiele fangen leicht an und die Programme sind so einfach, dass sie noch in einem einzelnen Kapitel behandelt werden können. Später, wenn die Programme komplizierter werden, wird ein Kapitel nicht mehr ausreichen und eine Anwendung wird dich auch über einen längeren Zeitraum begleiten. Du wirst lernen, wie du deine Programme immer weiter verbesserst und verfeinerst, indem du ein Menü oder eine Symbolleiste hinzufügst oder dem Programm ein eigenes Symbol im Dock zuweist, damit es auch dort einen guten Eindruck macht.



Wie arbeitest du mit diesem Buch?

Grundsätzlich besteht dieses Buch aus einer Menge Text mit vielen Abbildungen dazwischen. Natürlich habe ich mich bemüht, alles so aufzubereiten, dass daraus lauter leicht verdauliche Happen werden. Damit das Ganze noch genießbarer wird, gibt es zusätzlich noch einige Symbole, die ich dir hier gerne erklären möchte:

Arbeitsschritte

➤ Wenn du dieses Zeichen siehst, heißt das: Es gibt etwas zu tun. Damit kommen wir beim Programmieren Schritt für Schritt einem neuen Ziel immer näher.

Grundsätzlich lernt man besser, wenn man einen Programmtext selbst eintippt oder ändert. Aber nicht immer hat man große Lust dazu. Weil alle Projekte im Buch auch auf der CD sind, findest du dort für jedes Kapitel einen Ordner mit den Beispielen. Wenn du also das Projekt nicht selbst erstellen willst, kannst du stattdessen die zugehörige Datei laden.

Aufgaben

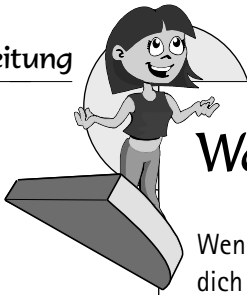
Am Ende der meisten Kapitel folgen ein paar Fragen, mit denen du kontrollieren kannst, ob du alles verstanden hast. Dann gibt es Aufgaben und kleine Übungen, um deine Programmierkenntnisse zu fordern. Die Antworten zu den Fragen findest du im Anhang des Buches und die Aufgaben als Projekte auf der CD. Aber auch zwischendurch gibt es immer mal wieder Vorschläge, was du mit dem neu erworbenen Wissen noch alles programmieren kannst.

Wichtige Stellen im Buch

Hin und wieder findest du ein solch dickes Ausrufezeichen im Buch. Das ist dann eine Stelle, an der etwas besonders Wichtiges steht.

Wenn es um eine ausführlichere Erläuterung geht, tritt Buffi in Erscheinung und bringt dir was aus seiner Kiste mit Tipps & Tricks.





Was brauchst du für dieses Buch?

Wenn du einen neuen oder nicht zu alten Mac hast, gibt es nichts, was dich vom Programmieren abhalten könnte, da du die meisten Dinge, die du benötigst, schon hast oder kostenlos bekommen kannst.

Betriebssystem

Um mit diesem Buch das Programmieren auf dem Mac zu erlernen, benötigst du auf deinem Computer eine aktuelle Version des Apple Betriebssystems OS X, jedoch mindestens die Version 10.7 »Lion«. Was für einen Computer du genau hast, spielt keine Rolle. Ein iMac ist genauso gut geeignet wie ein MacBook oder ein MacMini. Du kannst auf jedem dieser Rechner Programme für alle anderen erstellen. Das war nicht immer so. Um für den ersten Macintosh-Computer zu programmieren, benötigte man einen ganz anderen Rechner, die Lisa, ebenfalls von der Firma Apple.

Programme

Programmiert wird auf dem Mac mit dem Programm Xcode. Xcode ist eine Entwicklungsumgebung auch IDE (Integrated Development Environment) genannt. Obwohl Xcode zweifellos das wichtigste Programm für einen Entwickler ist, ist es doch nur Teil eines Paketes, das sich »Xcode Developer Tools«, übersetzt »Werkzeuge für Entwickler«, nennt. Dieses Paket enthält neben dem Programm Xcode noch weitere nützliche Anwendungen, von denen du einige in diesem Buch kennenlernen wirst.

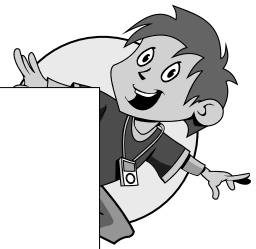
Xcode mit den Werkzeugen für Entwickler bekommst du kostenlos im Mac App Store und kannst es von dort auf deinen Computer herunterladen.

Im Internet

Auf meiner Internetseite zum Buch findest du außer den Beispielen zum Herunterladen auch Informationen zu anderen Themen, die es zwar nicht in das Buch geschafft haben, die dich aber vielleicht auch interessieren.

<http://www.cocoa-coding.de/kids/>

Sollten sich Fehler im Buch finden oder sollte Apple Anpassungen an Xcode vornehmen, werde ich versuchen, diese auf meiner Seite zu protokollieren, damit du auch weiterhin problemlos mit dem Buch arbeiten kannst. Bei Fragen zu den Beispielen oder wenn du selbst Fehler findest, kannst du dich gerne auch direkt an mich wenden, meine E-Mail-Adresse findest du ebenfalls auf der Internetseite.



Die Geschichte vom Code und vom Compiler

Du hast vielleicht schon mal den Satz gehört, dass ein Computer nur 0 und 1 versteht. In Wirklichkeit ist es noch viel schlimmer, denn da ein Computer mit elektrischem Strom arbeitet, kann er nur zwischen den Zuständen »Strom fließt« und »Strom fließt nicht« unterscheiden. Kombinationen aus diesen beiden Möglichkeiten veranlassen den Rechner dann, etwas zu tun. Zum Beispiel zwei Zahlen zu addieren oder einen Text auf den Bildschirm zu schreiben.

Natürlich wäre es sehr mühsam einen Computer zu programmieren, indem man durch verschiedene Schalter elektrischen Strom zum Fließen bringt. Die ersten Computer funktionierten tatsächlich noch so, aber diese Zeiten sind glücklicherweise lange vorbei. An der grundlegenden Funktion der Computer hat sich allerdings nichts geändert. Noch immer muss elektrischer Strom gesteuert werden. Allerdings ist die Eingabe der Programme heute etwas komfortabler. Ein Programm wird nicht mehr durch verschiedene Schalter erzeugt, sondern durch Anweisungen in einer Programmiersprache. Für so eine Auflistung von Befehlen gibt es viele Namen. Man nennt sie Quelltext, Programm(code), im Englischen »Sourcecode«, oder manchmal auch nur Code.

Wie auch in einer normalen Fremdsprache setzt sich der Programmcode aus »Vokabeln« und »Grammatik« zusammen. Wenn man eine Anweisung falsch schreibt, wird sie der Computer nicht verstehen, und wenn man Anweisungen in der falschen Reihenfolge verwendet, wird nicht das passieren, was man möchte. Der Vergleich mit einer Fremdsprache geht sogar noch weiter, denn je mehr Vokabeln du kennst, umso mehr kannst du in der Sprache erreichen, und umso leichter ist der Umgang mit dieser Sprache.

Leider ist der Code, auch wenn er in einer Programmiersprache geschrieben wurde, nur ein Text, mit dem ein Computer zunächst nicht viel anfangen kann. Das Programm muss wieder in eine Serie von 0 und 1, also in Signale von elektrischem Strom, umgewandelt werden. Diese Aufgabe übernimmt ein Programm, das Compiler heißt. Dieser Compiler, zu Deutsch »Übersetzer«, wandelt den Programmcode in ein für den Computer geeignetes Format um, das man auch Maschinensprache nennt. Gleichzeitig entdeckt der Compiler bei der Übersetzung Fehler, wenn die verwendeten Anweisungen nicht denen entsprechen, die der Compiler kennt.



Ein weiteres Programm, mit dem man als Entwickler aber wenig zu tun hat, ist der Linker. Der Linker, zu Deutsch »Verknüpfen« oder »Binder«, hat die Aufgabe, die vom Compiler übersetzten Einzelteile zu einem großen Ganzen zusammenzusetzen. Das passiert in der Regel automatisch, ohne dass man selbst etwas machen muss.

In der Vergangenheit waren Compiler, Linker und der Texteditor für den Programmcode oft eigenständige Programme, die vom Entwickler einzeln bedient werden mussten. Moderne Werkzeuge wie Xcode fassen aber alle Funktionen zusammen und tragen daher die Bezeichnung »integrierte Entwicklungsumgebung«.

Unabhängig davon, wie schnell dein Rechner sein mag, wird die Arbeit von Compiler und Linker immer einige Zeit in Anspruch nehmen. Glücklicherweise nicht mehr einige Stunden, wie es in den frühen Jahren der Datenverarbeitung nicht ungewöhnlich war. Bei einem umfangreichen Programm kann das zwar durchaus einige Minuten dauern – bei den Beispielen in diesem Buch jedoch nur wenige Sekunden.

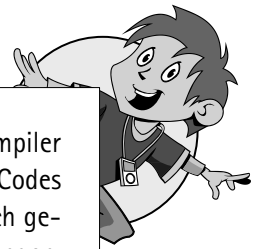
Ohne Programmiersprache geht es nicht

Falls du neugierig darauf bist, was für Programme du vielleicht in Zukunft schreiben kannst, dann sieh dir mal die Software an, die mit OS X ausgeliefert wurde. Auch iTunes und iPhoto wurden bei Apple in Objective-C erstellt und viele der dort verwendeten Bauteile dieser Programme stehen dir auch für eigene Anwendungen zur Verfügung.

Objective-C ist eine objektorientierte Programmiersprache. Was das genau bedeutet, wirst du in späteren Kapiteln noch genauer erfahren. Eines der Hauptmerkmale einer solchen Sprache ist die hohe Wiederverwendbarkeit von Programmteilen. Wenn du also einen Programmteil, wie zum Beispiel eine komplizierte Berechnung, erfolgreich programmiert hast, kannst du ihn später sehr leicht in anderen Programmen wiederverwenden.

Bevor du allerdings anfangen kannst zu programmieren, solltest du zumindest einige der wichtigsten Grammatikregeln dieser Programmiersprache kennen.

In Objective-C folgt am Ende jeder Anweisung ein Semikolon (;). Dieses Zeichen signalisiert dem Compiler eindeutig, dass der Befehl dort zu Ende ist und dass die folgende Anweisung ein neuer Befehl sein soll. Das ist nötig, da nicht alle Anweisungen gleich lang und gleich aufgebaut sind. Setzt



man aber ein Semikolon zu früh oder vergisst man es, wird der Compiler das in den meisten Fällen bemerken und bei der Übersetzung des Codes bemängeln. Man kann allerdings auch Pech haben und durch ein falsch gesetztes Semikolon entstehen andere, aber trotzdem gültige, Anweisungen. Das ist einer der Gründe, warum der Compiler Fehler im Programmcode zwar erkennt, aber nicht selbstständig korrigiert.

Während des Programmierens gibt es immer wieder Dinge, die für den Menschen und nicht für den Computer wichtig sind. Die Formatierung und das Layout des Programmcodes sind das beste Beispiel. So ist es möglich, mehrere Anweisungen einfach hintereinander zu schreiben. Natürlich immer mit einem Semikolon getrennt.

```
Anweisung1;Anweisung2;Anweisung3;
```

Das würde problemlos funktionieren, besser ist es aber, jede Anweisung in eine neue Zeile zu schreiben.

```
Anweisung1;  
Anweisung2;  
Anweisung3;
```

Die Formatierung des Codes ist für den Compiler natürlich vollkommen uninteressant. Am Semikolon erkennt er genau, wann eine Anweisung zu Ende ist. Ob die Befehle in einer Zeile hintereinander oder in mehreren Zeilen untereinander stehen, macht also für ihn überhaupt keinen Unterschied. Aber auch der Programmierer sollte in der Lage sein, das Programm zu lesen und seine Funktion nachvollziehen zu können. Das geht viel besser, wenn der Programmcode gut formatiert ist. Später wirst du Anweisungen kennenlernen, die ineinander verschachtelt sind, und dann ist leicht lesbarer Code wirklich sehr wichtig. Es gibt nämlich auch Befehle, die aus mehr als einer Zeile bestehen.

```
Anweisung;  
{  
    Anweisung;  
}
```

So ein Konstrukt aus geschweiften Klammern nennt man Block und man benutzt Blöcke immer dann, wenn man Befehle mehrfach oder nur unter bestimmten Bedingungen ausführen will. Auch hier sollte man den Code gut formatieren und die Anweisungen im Block einrücken. Die Tabulatortaste ist dabei eine große Hilfe. Es gibt aber auch Einstellungen in Xcode, die dich bei der Formatierung des Codes unterstützen. Außerdem kannst du Kommentare in deinen Quelltext schreiben, um dir direkt bei den Anweisungen kleine Notizen zu machen.



Das alles klingt für dich vielleicht sehr theoretisch und es wurde hier auch schon einiges vorweggenommen, was später noch sehr detailliert erklärt wird. Lass dich davon am besten nicht beeindrucken. Wenn du die ersten Programme selbst geschrieben hast, wirst du bestimmt vieles besser verstehen. Lies diese Einleitung einfach später noch einmal und alles wird dir ganz leicht und verständlich vorkommen. Programmieren lernt man nicht in der Theorie, sondern nur, indem man wirklich Programme schreibt. Deshalb geht es im nächsten Kapitel auch sofort richtig los. Dein erstes eigenes Projekt wartet!

Die Sache mit der Tastatur

Manche der Zeichen, die man zum Programmieren ständig benötigt, sind leider auf der deutschen Tastatur nicht abgebildet. Man kann sie zwar problemlos eingeben, oft ist aber etwas Eingewöhnung nötig, bis man sich gemerkt hat, wo diese Zeichen verborgen sind. Hauptsächlich handelt es sich dabei um Klammern.

Die eckigen Klammern [] erreichst du mit der `Alt`-Taste und den Zahlentasten `5` und `6`.

Die geschweiften Klammern { } erreichst du mit der `Alt`-Taste und den Zahlentasten `8` und `9`.

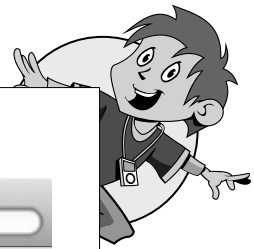
Das funktioniert aber nur mit den Zahlentasten auf dem Haupttastenblock. Mit den Zahlentasten des Nummernblocks funktioniert es nicht.

Die Sache mit der Maus

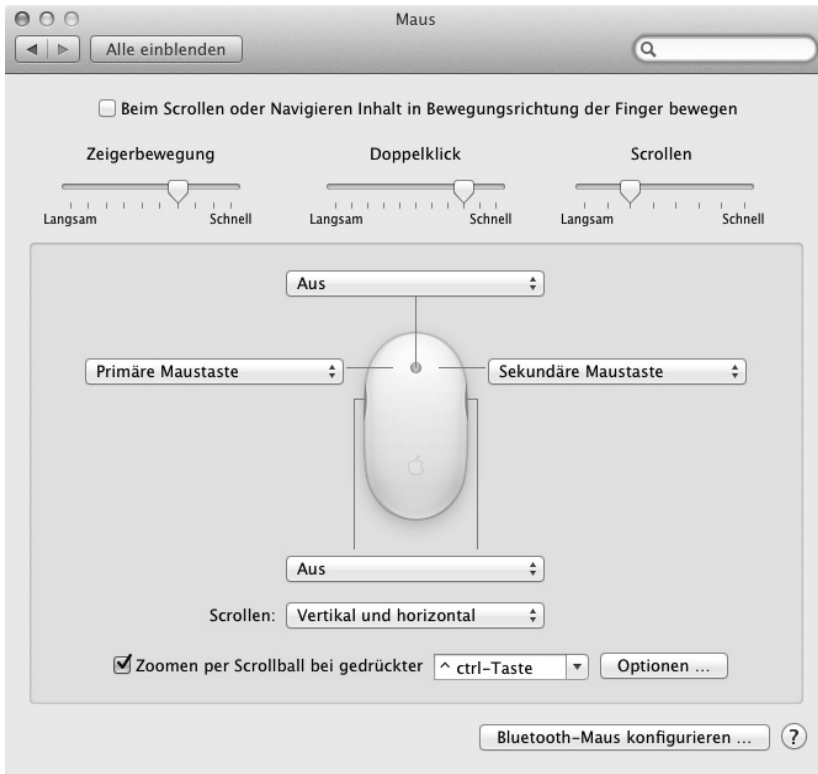
In der Vergangenheit hatten Mäuse der Mac Computer manchmal nur eine Taste, und obwohl die Apple Mouse und auch die Magic Mouse inzwischen sogar mehr als zwei Tasten haben, muss man die zusätzlichen Funktionen erst selbst konfigurieren.

In Xcode, dem Programmierwerkzeug auf dem Mac, gibt es sehr viele Befehle, die man durch einen Klick auf die sekundäre Maustaste ausführen kann. Deshalb ist es sinnvoll, die Maus so anzupassen, dass sie wirklich mit mehreren Maustasten arbeitet. Dies geschieht in den Systemeinstellungen von OS X und sieht je nach verwendeter Maus etwas anders aus.

Die Sache mit der Maus



Die Systemeinstellungen für die Apple Mouse:



Die gleichen Einstellungen für die kabellose Magic Mouse:





Wenn dir aber die klassische Arbeitsweise lieber ist, kannst du auch weiterhin mit nur einer Maustaste arbeiten. Für die alternativen Funktionen musst du beim Anklicken immer zusätzlich die `Ctrl`-Taste gedrückt halten. Immer wenn in diesem Buch von der sekundären oder »rechten« Maustaste die Rede ist, kannst du das Gleiche auch bei gedrückter `Ctrl`-Taste mit der primären Maustaste erreichen.