Ein Buch zum Mitmachen und Verstehen

Mit Arduino-Workshop

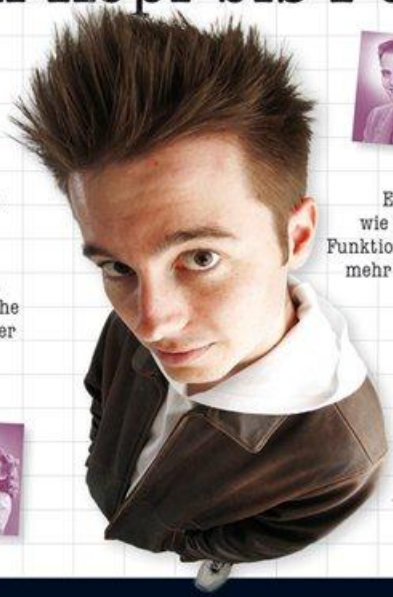
C

von Kopf bis Fuß

Entdecken Sie die Geheimnisse der C-Gurus

Vermeiden Sie peinliche Zeigerfehler

Spielen Sie mit der C-Standard-bibliothek herum



Finden Sie heraus, wie make Ihr Leben verändern kann

Erfahren Sie, wie variadische Funktionen Susi zu mehr Flexibilität verhalfen

Reanimieren Sie ein klassisches Arcade-Spiel

O'REILLY®

David Griffiths & Dawn Griffiths
Deutsche Übersetzung von Lars Schulten

Die Informationen in diesem Buch wurden mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Verlag, Autoren und Übersetzer übernehmen keine juristische Verantwortung oder irgendeine Haftung für eventuell verbliebene Fehler und deren Folgen. D.h., wenn Sie beispielsweise ein Kernkraftwerk unter Verwendung dieses Buchs betreiben möchten, tun Sie dies auf eigene Gefahr.

Alle Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt und sind möglicherweise eingetragene Warenzeichen. Der Verlag richtet sich im Wesentlichen nach den Schreibweisen der Hersteller. Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Alle Rechte vorbehalten einschließlich der Vervielfältigung, Übersetzung, Mikroverfilmung sowie Einspeicherung und Verarbeitung in elektronischen Systemen.

Kommentare und Fragen können Sie gerne an uns richten:

O'Reilly Verlag
Balthasarstr. 81
50670 Köln
E-Mail: kommentar@oreilly.de

1. Auflage 2012

Die Originalausgabe erschien 2012 unter dem Titel
Head First C bei O'Reilly Media, Inc.

Bibliografische Information Der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.d-nb.de* abrufbar.

Übersetzung und deutsche Bearbeitung: Lars Schulten, Köln
Lektorat: Alexandra Follenius, Köln
Korrektorat: Sibylle Feldmann, Düsseldorf
Satz: Ulrich Borstelmann, Dortmund
Produktion: Andrea Miß, Köln
Belichtung, Druck und buchbinderische Verarbeitung: Media-Print, Paderborn

ISBN 978-3-86899-386-8

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

Für Dennis Ritchie (1941–2011), den Vater von C.

Die Autoren von C von Kopf bis Fuß

David Griffiths

Dawn Griffiths

David Griffiths begann im zarten Alter von 12 mit dem Programmieren, nachdem er eine Dokumentation über die Arbeiten von Seymour Papert gesehen hatte. Mit 15 schrieb er eine Implementierung von Paperts Programmiersprache LOGO. Nachdem er an der Universität Mathematik studiert hatte, begann er, Code für Computer und Zeitschriftenartikel für Menschen zu schreiben. Er hat als Trainer für agile Softwareentwicklung, Entwickler und Parkplatzwächter gearbeitet, allerdings nicht in dieser Reihenfolge. Er kann in über zehn Sprachen programmieren, schreiben aber nur in einer. Wenn er nicht schreibt, programmiert oder lehrt, verbringt er seine Zeit auf Reisen mit seiner Frau – und Koautorin – Dawn.

Vor *C von Kopf bis Fuß* hat er bereits zwei andere Von Kopf bis Fuß-Bücher geschrieben: *Programmieren von Kopf bis Fuß* und *Head First Rails*.

Sie können ihm auf Twitter folgen:
http://twitter.com/dgriffiths

Dawn Griffiths begann ihre berufliche Laufbahn als Mathematikerin an einer großen britischen Universität, an der sie einen ausgezeichneten Abschluss in Mathematik machte. Anschließend begann sie ihre Karriere in der Softwareentwicklung und kann heute auf mittlerweile 15 Jahre Berufserfahrung in der IT-Industrie zurückblicken.

Bevor sie sich mit David zusammentat, um *C von Kopf bis Fuß* zu schreiben, hat Dawn bereits zwei andere Von Kopf bis Fuß Bücher geschrieben (*Statistik von Kopf bis Fuß* und *Head First 2D Geometry*) und außerdem an einem Haufen weiterer Bücher in der Reihe mitgewirkt.

Wenn Dawn nicht an Von Kopf bis Fuß-Büchern arbeitet, feilt sie an ihren Tai-Chi-Fähigkeiten, geht laufen, klöppelt oder kocht. Außerdem reist sie gern und verbringt Zeit mit ihrem Ehemann David.

Über den Übersetzer dieses Buchs

Lars Schulten ist freier Übersetzer für IT-Fachliteratur und hat für den O'Reilly Verlag schon unzählige Bücher zu ungefähr allem übersetzt, was man mit Computern so anstellen kann. Eigentlich hat er mal Philosophie studiert, aber mit Computern schlägt er sich schon seit den Zeiten herum, da Windows laufen lernte. Die Liste der Dinge, mit denen er sich beschäftigt, ist ungefähr so lang, launenhaft und heterogen wie die seiner Lieblingsessen und Lieblingsbücher.

Allein tritt er eigentlich nur auf, wenn er mal wieder versucht, den körperlichen Verfall mit sportlicher Betätigung aufzuhalten. Sonst ist er immer in Begleitung eines Buchs, seines Laptops oder Frederics unterwegs. Frederic ist zehn Jahre alt und setzt gern eine sehr kritische Miene auf, wenn Papa die Spielerei mit dem Computer als Arbeit bezeichnet.

Verwandte Bücher von O'Reilly

C – kurz & gut

C++ – kurz & gut

Weitere Bücher aus O'Reillys *Von Kopf bis Fuß*-Reihe

C# von Kopf bis Fuß

Datenanalyse von Kopf bis Fuß

Entwurfsmuster von Kopf bis Fuß

HTML mit CSS & XHTML von Kopf bis Fuß

HTML5-Programmierung von Kopf bis Fuß

Java von Kopf bis Fuß

JavaScript von Kopf bis Fuß

jQuery von Kopf bis Fuß

Mobiles Web von Kopf bis Fuß

Netzwerke von Kopf bis Fuß

Objektorientierte Analyse & Design von Kopf bis Fuß

PHP & MySQL von Kopf bis Fuß

Programmieren von Kopf bis Fuß

Python von Kopf bis Fuß

Servlets & JSP von Kopf bis Fuß

Softwareentwicklung von Kopf bis Fuß

SQL von Kopf bis Fuß

Statistik von Kopf bis Fuß

Webdesign von Kopf bis Fuß

Head First Algebra

Head First Physics

Head First Ajax

Head First Rails

Head First PMP

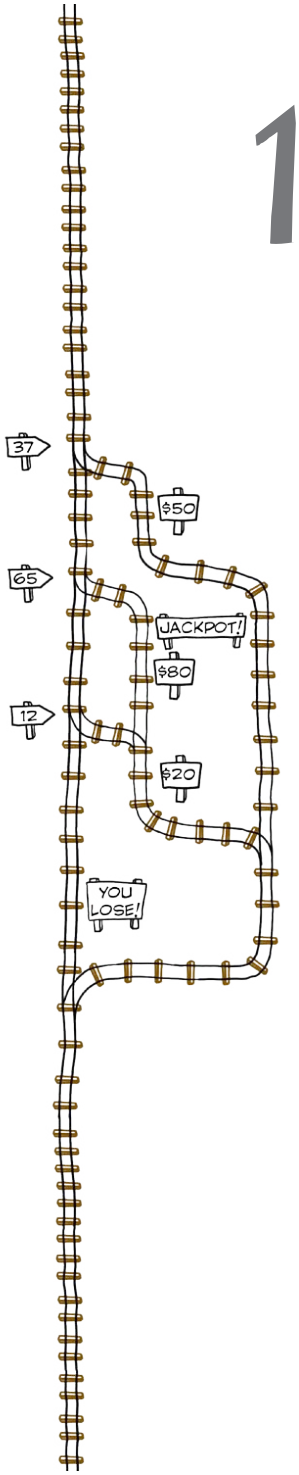
Head First EJB

Der Inhalt (im Überblick)

Der Inhalt (jetzt ausführlich)

Einführung

Ihr Gehirn und C. *Sie* versuchen, etwas zu *lernen*, und Ihr *Hirn* tut sein Bestes, damit das Gelernte nicht *hängen bleibt*. Es denkt nämlich: »Wir sollten lieber ordentlich Platz für wichtigere Dinge lassen, z. B. für das Wissen darüber, welche Tiere einem gefährlich werden könnten oder dass es eine ganz schlechte Idee ist, nackt Snowboard zu fahren.« Tja, *wie* schaffen wir es nun, Ihr Gehirn davon zu überzeugen, dass Ihr Leben davon abhängt, etwas über C zu wissen?



Erste Schritte mit C

1 Eintauchen

Wollen Sie dem Rechner auf die Finger schauen?

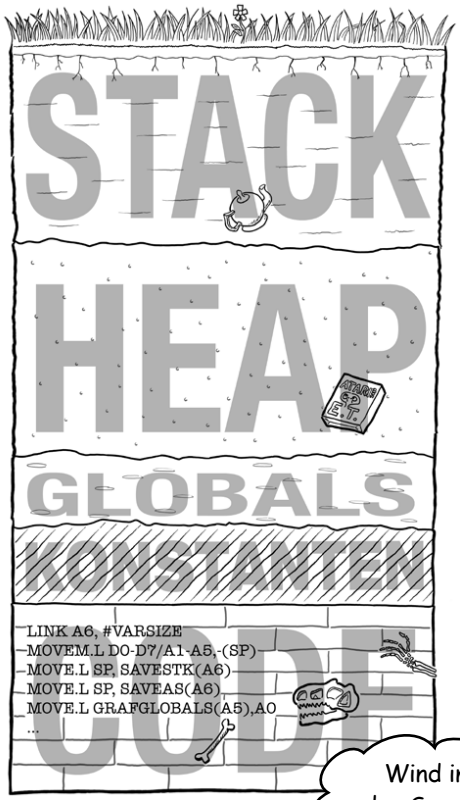
Müssen Sie **Hochleistungscode** für ein neues Spiel schreiben? Einen **Arduino**-Controller programmieren? Oder diese raffinierte **externe Bibliothek** in Ihrer iPhone-App einsetzen? Wenn das der Fall ist, wird C Ihr bester Freund werden. C arbeitet auf einer **viel elementareren Ebene** als die meisten anderen Programmiersprachen. Wenn Sie C verstanden haben, werden Sie auch viel besser verstehen, **was eigentlich im Herzen der Maschine vor sich geht**. C kann Ihnen sogar helfen, andere Sprachen besser zu verstehen. Zögern Sie nicht! Machen Sie Ihren Compiler bereit – auf dass Sie schon bald loslegen können.

Speicher und Zeiger

2 Worauf zeigst du?

Wenn Sie C wirklich beherrschen wollen, müssen Sie verstehen, wie C mit Speicher umgeht.

C bietet Ihnen ziemlich umfangreiche Möglichkeiten, zu *steuern*, wie Ihr Programm den **Speicher des Systems** nutzt. In diesem Kapitel werden wir den Vorhang lüften und Ihnen zeigen, was passiert, wenn Sie **Variablen lesen und schreiben**. Sie werden erfahren, **wie Arrays funktionieren**, wie man einige **garstige Speicherprobleme vermeidet,** und natürlich auch einsehen lernen, dass der Weg zum gewieften C-Programmierer nur über **die Beherrschung von Zeigern und der Speicheradressierung** führt.



2.5

Strings

Stringtheorie

Strings muss man nicht nur lesen.

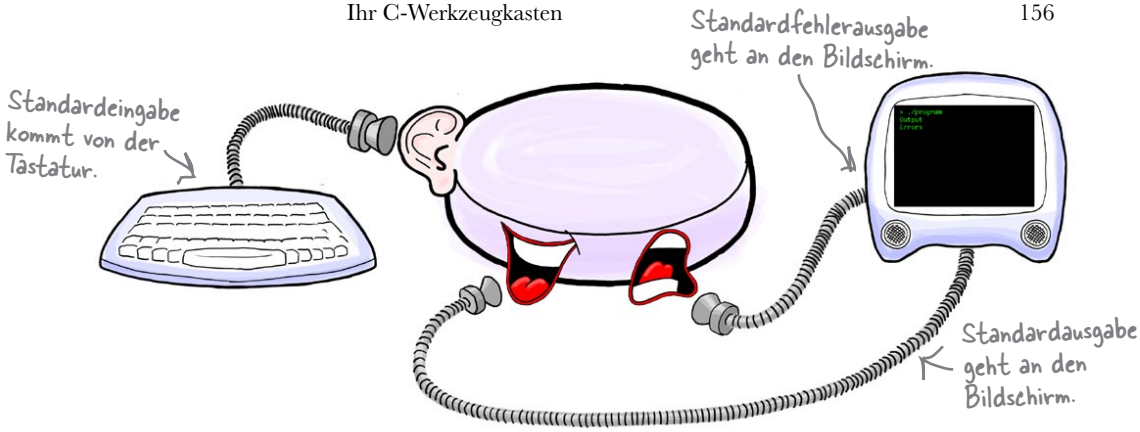
Sie haben erfahren, dass Strings in C eigentlich `char`-*Arrays* sind, aber was C Sie mit ihnen *anstellen* lässt, das wissen Sie noch nicht. Dazu müssen wir uns ***string.h*** zuwenden. *string.h* ist ein Teil der C-Standardbibliothek, der sich ganz der **Stringmanipulation** widmet. Wenn Sie Strings ***verketten***, einen String in einen anderen ***kopieren*** oder zwei Strings ***vergleichen*** wollen, können die Funktionen in *string.h* nützlich sein. In diesem Kapitel werden Sie sehen, wie Sie ein **Array mit Strings** erstellen, bevor wir uns genauer ansehen werden, wie man mit der Funktion **`strstr()` *in Strings sucht***.

Kleine Werkzeuge erstellen

3 Eine Sache tun, und das gut

Alle Betriebssysteme beinhalten kleine Werkzeuge.

Kleine in C geschriebe Werkzeuge erledigen **kleine spezielle Aufgaben**, schreiben oder lesen Dateien oder filtern Daten. Wenn Sie komplexere Aufgaben bewältigen müssen, können Sie *mehrere Werkzeuge hintereinanderschalten*. Aber wie werden diese kleinen Werkzeuge erstellt? In diesem Kapitel werden wir uns die Bausteine der Erstellung kleiner Werkzeuge anschauen. Sie werden lernen, wie man **Kommandozeilenoptionen** steuert, wie man **Informationsströme** verarbeitet und **Umleitungen** einsetzt, um im Handumdrehen Werkzeuge zu bauen.

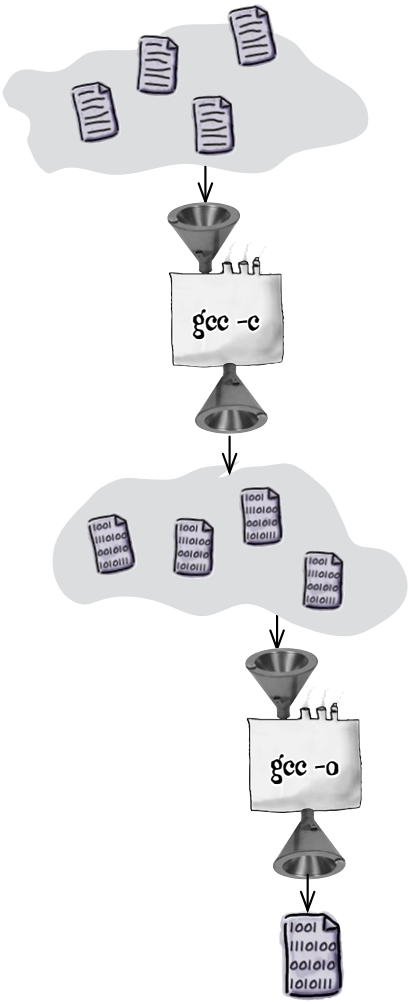


Mehrere Quelldateien

4 Zerlegen und zusammenbauen

Wenn Sie ein großes Programm erstellen, heißt das nicht, dass Sie auch eine große Quelldatei haben wollen.

Können Sie sich vorstellen, wie schwierig und zeitaufwendig die Wartung einer einzigen Quelldatei bei umfangreichen Programmen werden kann? In diesem Kapitel werden Sie erfahren, wie Ihnen C ermöglicht, Quellcode in **kleine, handhabbare Happen** zu zerlegen und diese dann zu **einem großen Programm** zusammenzusetzen. Auf dem Weg dorthin werden Sie etwas mehr über die **Feinheiten von Datentypen** erfahren und werden jemandem über den Weg laufen, der einer Ihrer besten Freunde werden wird: **`make`**.



1. C-Workshop

Arduino

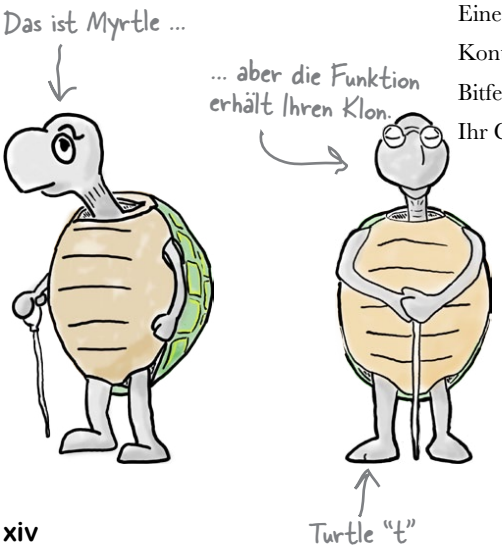
Hatten Sie sich schon einmal gewünscht, dass Ihre Pflanzen Ihnen sagen könnten, wann sie Wasser brauchen? Mit einem Arduino geht das! In diesem Workshop werden Sie eine Arduino-gesteuerte und in C programmierte Pflanzenüberwachung konstruieren.

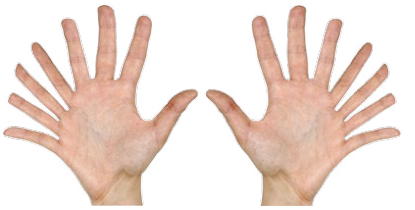
Structs, Unions und Bitfelder

5 Eigene Strukturen

Die meisten Dinge im Leben sind komplexer als einfache Zahlen.

Bislang haben wir uns die elementaren Datentypen der Programmiersprache C angesehen, aber was ist, wenn Zahlen und einfache Zeichenfolgen nicht mehr ausreichen? Was ist, wenn Sie **Dinge aus dem wahren Leben modellieren wollen**? `structs` ermöglichen Ihnen, **die Komplexität der realen Welt zu modellieren**, indem Sie eigene Strukturen gestalten. In diesem Kapitel werden Sie lernen, wie Sie die **elementaren Datentypen** zu Strukturen **kombinieren** und mit Unions die **Unwägbarkeiten des Lebens in den Griff bekommen**. Und wenn Ihnen ein einfaches Ja oder Nein ausreicht, können ***Bitfelder*** genau das Richtige für Sie sein.



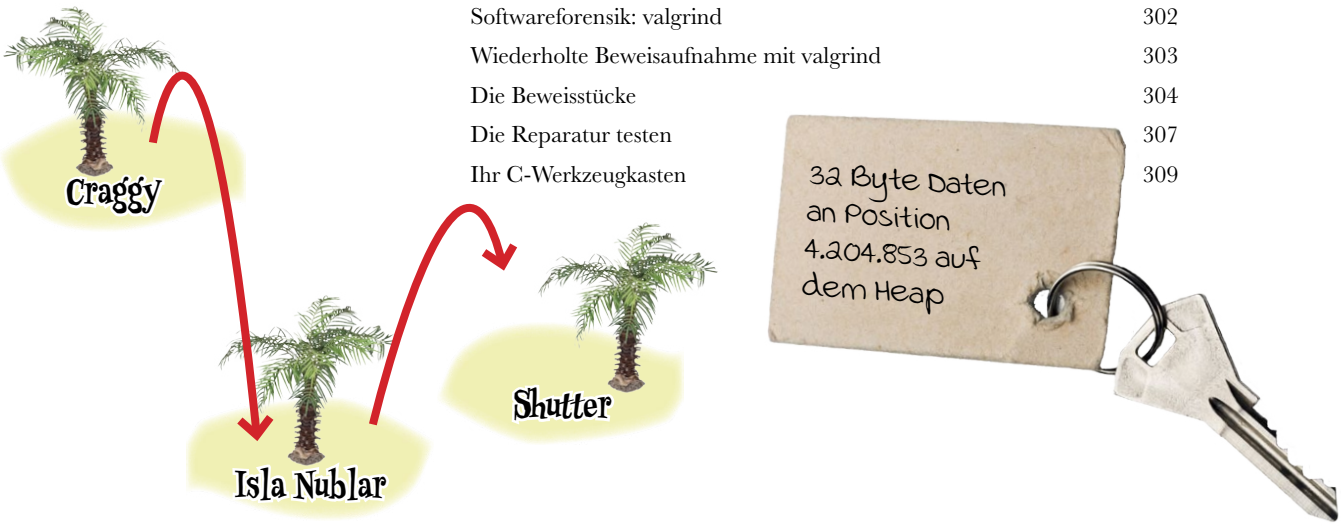


Datenstrukturen und dynamischer Speicher

6 Brücken bauen

Manchmal reicht ein Struct einfach nicht aus.

Zur Modellierung komplexer Datenanforderungen muss man häufig Structs **verbinden**. In diesem Kapitel werden Sie erfahren, wie Sie Struct-**Zeiger** nutzen können, um eigene Datentypen zu **großen, komplexen Datenstrukturen** zusammenzuknüpfen. Sie werden *Schlüsselprinzipien* erforschen, indem Sie **verkettete Listen** erstellen. Sie werden auch erfahren, wie Sie Ihre Datenstrukturen dazu bringen, flexible Datenmengen zu bewältigen, indem Sie **dynamisch Speicher auf dem Heap allozieren** und wieder freigeben, wenn Sie ihn nicht mehr benötigen. Und dann werden wir Ihnen ein Werkzeug vorstellen, das Sie unterstützen kann, wenn Sie Probleme bei der Haushaltsführung haben: **`valgrind`**.

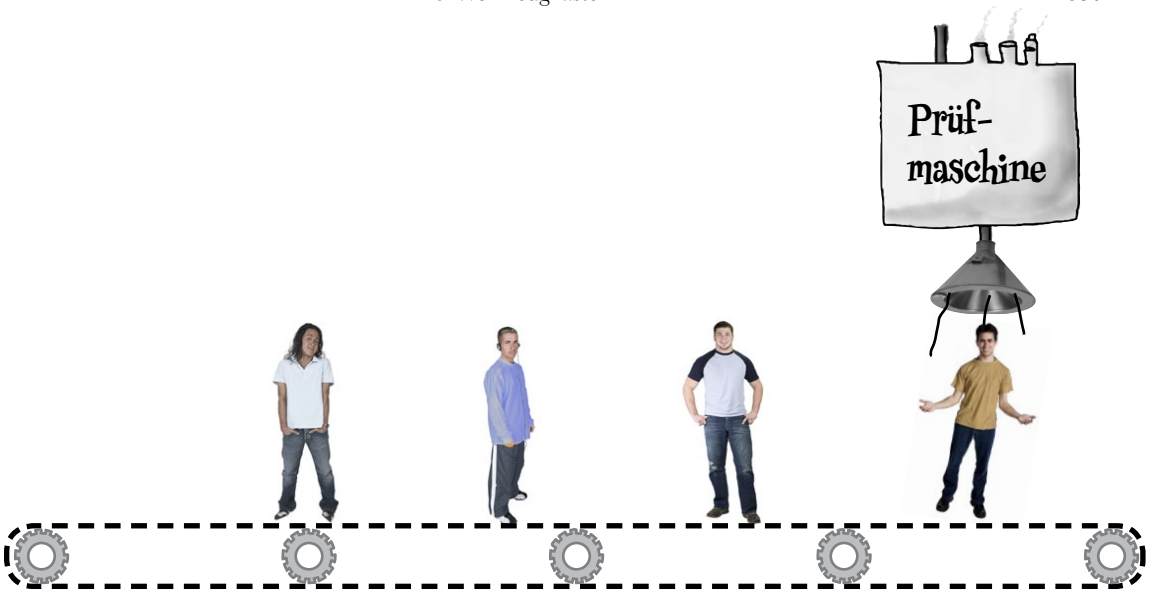


Fortgeschrittene Funktionen

7 Ihre Funktionen auf Vordermann bringen

Einfache Funktionen sind schön, aber reichen manchmal nicht aus.

Bislang haben Sie sich auf die Grundlagen beschränkt, aber was ist, wenn Sie mehr *Macht* und *Flexibilität* benötigen, um Ihr Ziel zu erreichen? In diesem Kapitel werden Sie erfahren, wie Sie **den IQ Ihres Codes aufpeppen können**, indem Sie **Funktionen als Parameter übergeben**. Sie werden erfahren, wie man **Dinge mit Vergleichsfunktionen sortiert**. Und schließlich werden Sie entdecken, wie Sie Ihren Code mit **variadischen Funktionen** *anpassungsfähiger* machen.



Statische und dynamische Bibliotheken

8 Code-Wiederverwendung

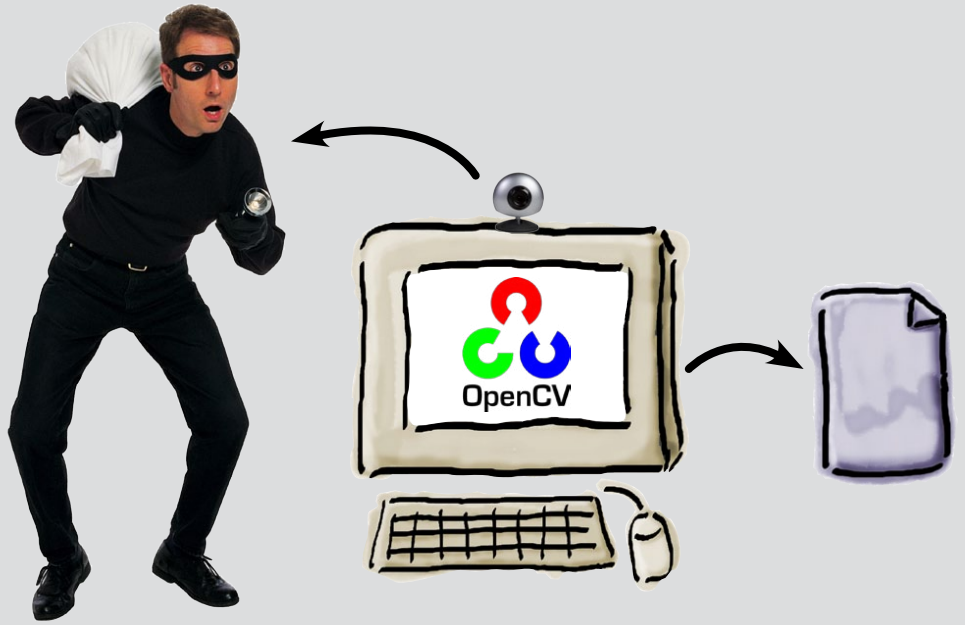
Die Macht der Standardbibliothek haben Sie bereits kennengelernt.

Jetzt ist es an der Zeit, dass Sie diese Macht für Ihren *eigenen* Code einsetzen. In diesem Kapitel werden Sie lernen, wie Sie Ihre **eigenen Bibliotheken** erstellen und den **gleichen Code in mehreren Programmen wiederverwenden**. Außerdem werden Sie erfahren, wie Ihnen **dynamische Bibliotheken** ermöglichen, Ihren Code zur Laufzeit zu teilen. Sie werden in die Geheimnisse der *Programmiergurus* eingeweiht werden. Und wenn Sie das Ende dieses Kapitels erreicht haben, werden Sie Code schreiben können, der gut skaliert und leicht und effizient zu warten ist.

2. C-Workshop

OpenCV

Stellen Sie sich vor, Ihr Computer könnte ein Auge auf Ihr Haus werfen, während Sie unterwegs sind, und Sie informieren, wenn sich dort unbefugt jemand herumtreibt. Genau das ist mit einer an Ihren Rechner angeschlossenen oder in ihn eingebauten Webcam und den ausgefeilten Techniken von OpenCV möglich!

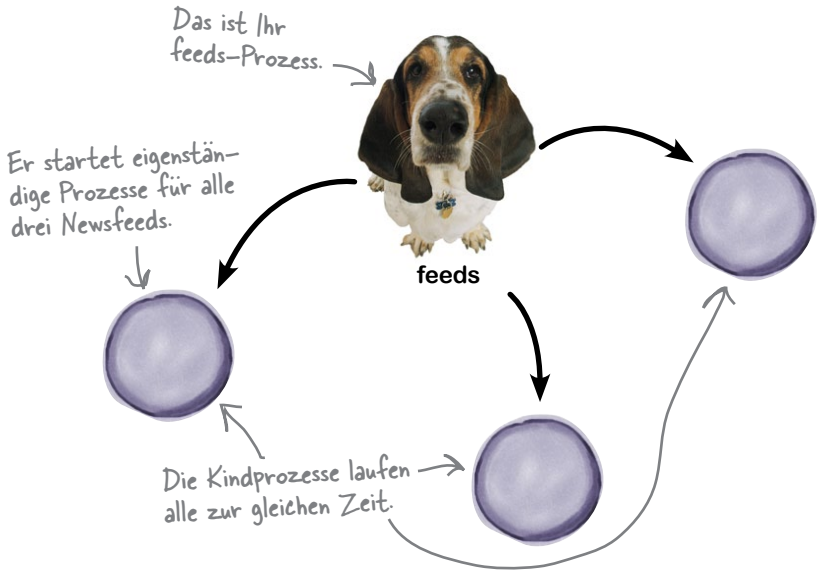


Prozesse und Systemaufrufe

9 Grenzverletzungen

Es wird Zeit, über den Tellerrand hinauszublicken.

Sie haben bereits erfahren, dass Sie komplexe Anwendungen aufbauen können, indem Sie auf der Kommandozeile kleine Werkzeuge miteinander verbinden. Aber was ist, *wenn Sie andere Programme* aus Ihrem Code heraus nutzen wollen? In diesem Kapitel werden Sie lernen, wie Sie **Systemeinrichtungen** nutzen, um ***Prozesse*** zu erstellen und zu kontrollieren. Das wird Ihren Programmen Zugriff auf *E-Mail*, das *Web* und Unmengen anderer *Werkzeuge* geben, die Sie installiert haben. Wenn Sie das Ende dieses Kapitels erreicht haben, werden Sie die Macht haben, die **Grenzen von C** zu verlassen.

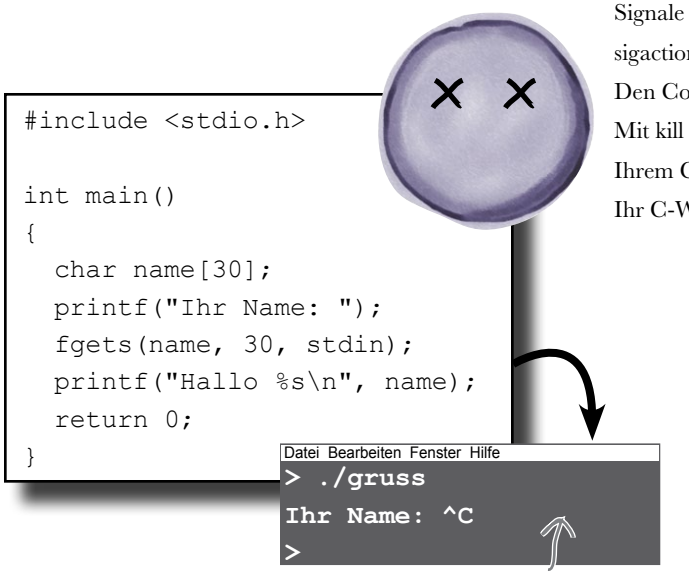


Interprozesskommunikation

10 Ein nettes Gespräch

Prozesse machen ist nur die halbe Miete.

Was ist, wenn Sie den Prozess *steuern* wollen, nachdem er gestartet ist? Was, wenn Sie ihm *Daten senden* wollen? Oder seine *Ausgabe lesen*? Die **Interprozesskommunikation** ermöglicht Prozessen, gemeinsame Sache zu machen. Wir werden Ihnen zeigen, wie Sie die **Macht** Ihres Codes multiplizieren, indem Sie ihn mit anderen Programmen auf Ihrem Systemen **reden lassen**.



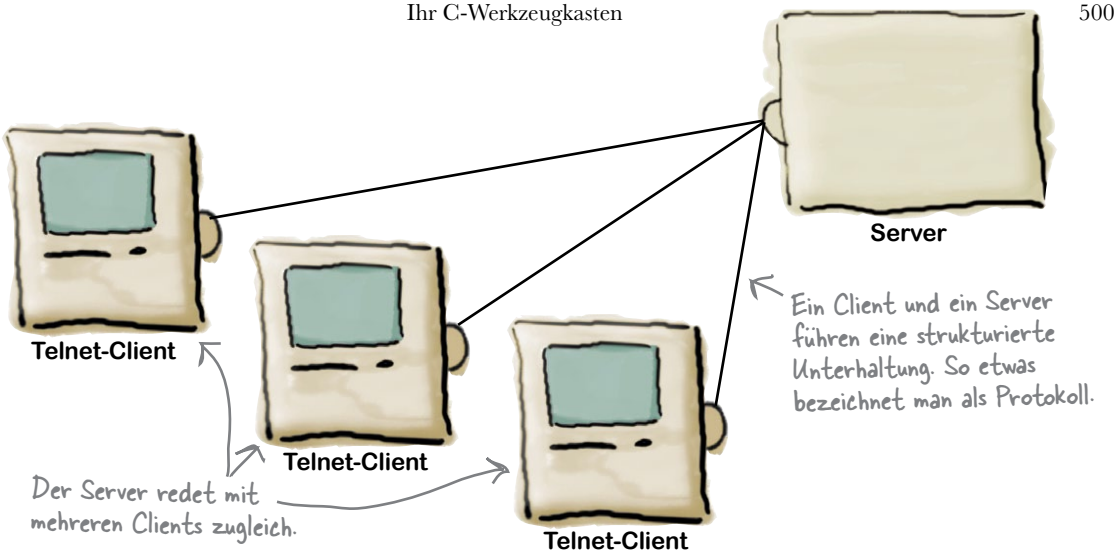
Wenn Sie Strg-C drücken, stellt das Programm die Arbeit ein. Aber warum?

Sockets und Netzwerke

11 127.0.0.1 ist ein toller Ort

Programme auf unterschiedlichen Systemen müssen miteinander reden.

Sie haben gelernt, wie Sie mithilfe von I/O-Operationen mit Dateien reden und wie Prozesse auf dem gleichen System miteinander kommunizieren können. Jetzt werden Sie nach *dem Rest der Welt greifen* und erfahren, wie man C-Programme schreibt, die **über das Netzwerk** und **quer über die ganze *Welt*** mit anderen Programmen reden können. Wenn Sie das Ende des Kapitels erreicht haben, werden Sie dazu in der Lage sein, **Programme zu schreiben, die sich wie Server verhalten, und Programme, die sich wie Clients verhalten**.



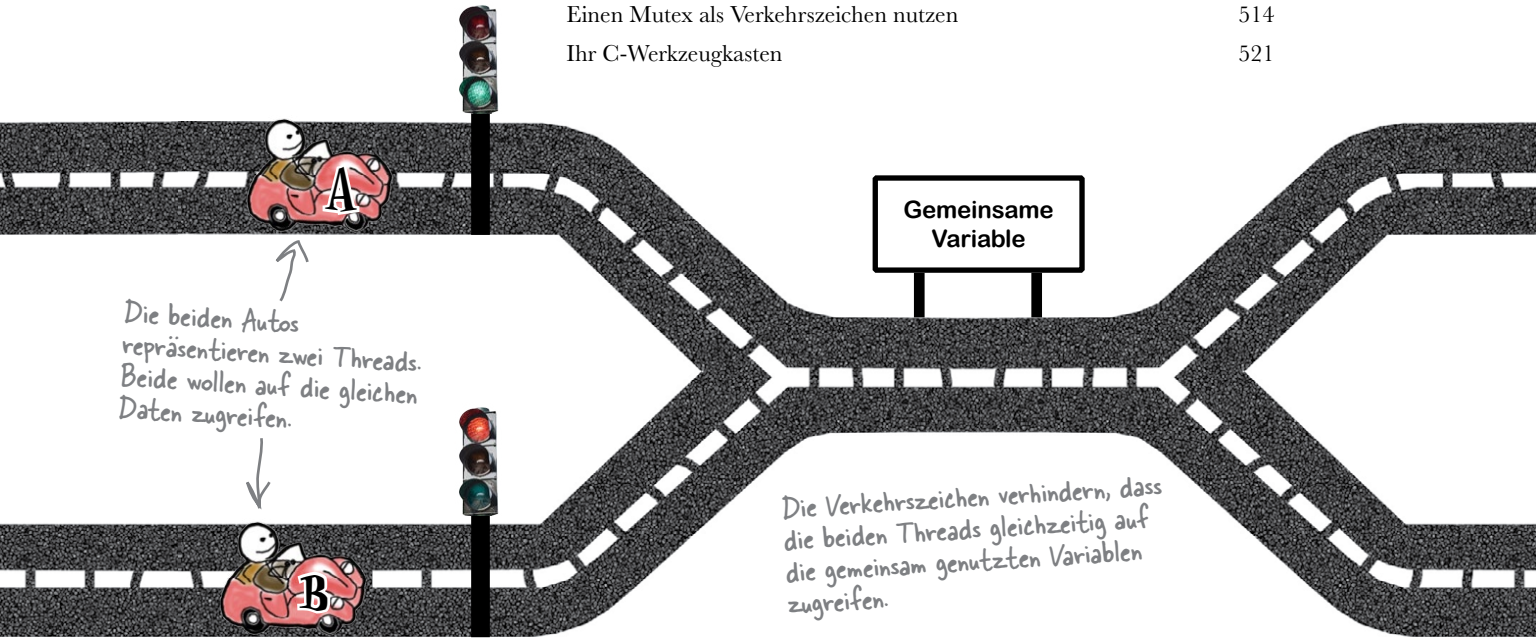
12

Threads

Parallelwelten

Häufig müssen Programme mehrere Dinge gleichzeitig tun.

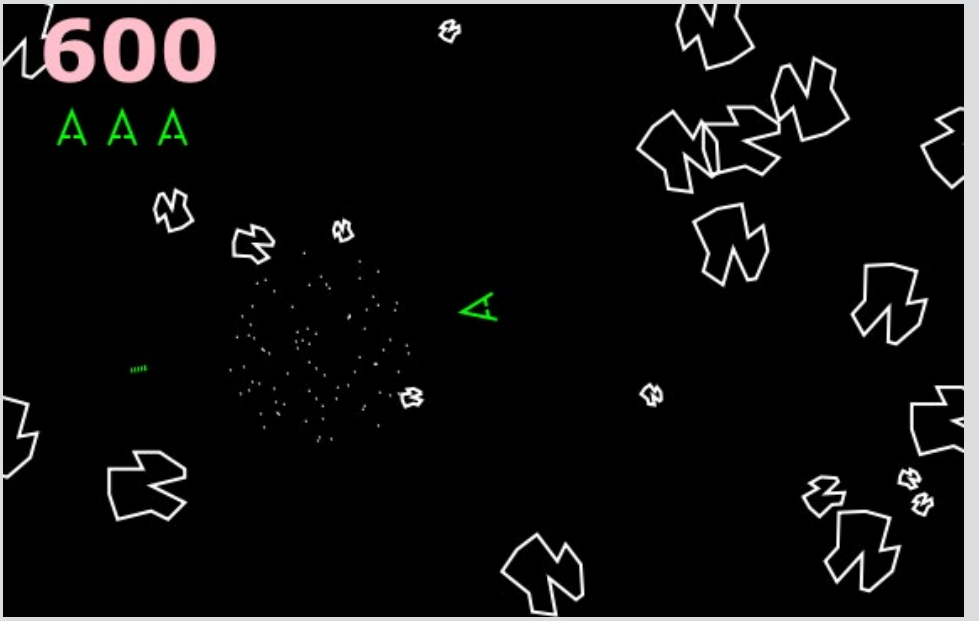
Mit POSIX-Threads können Sie Ihren Code reaktionsfähiger machen, indem **Sie verschiedene Codeteile abspalten und parallel laufen lassen**. Doch aufgepasst! Threads sind mächtige Werkzeuge, und Sie sollten tunlichst vermeiden, dass sie sich in die Quere kommen. In diesem Kapitel werden Sie lernen, wie Sie die **Verkehrsschilder** und **Straßenmarkierungen** einrichten, die ***Codeunfälle verhindern***. Am Ende werden Sie wissen, wie man **POSIX-Threads erstellt** und wie man **Synchronisationsmechanismen nutzt**, um ***die Integrität wichtiger Daten zu sichern***.



3. C-Workshop

Blasteroids

Einer der wichtigsten Gründe dafür, dass die Leute in C programmieren lernen wollen, ist, dass sie dann Spiele schreiben können. In diesem Workshop werden wir einem der beliebtesten und langlebigsten Videospiele aller Zeiten Tribut zollen. Schreiben wir Blasteroids!

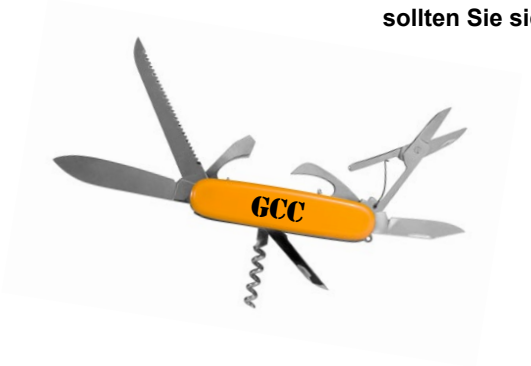


Anhang A: Was übrig bleibt

Die Top Ten der Dinge (die wir nicht behandelt haben)

Wir haben viel behandelt, und doch bleibt manches offen.

Da sind noch ein paar Dinge, die Sie wahrscheinlich wissen sollten. Wir würden uns nicht wohlfühlen, wenn wir sie nicht erwähnten, auch wenn wir sie nur kurz anreißen können – weil wir Ihnen ein Buch geben wollten, das Sie auch ohne ausgiebiges Muskeltraining noch transportieren können. Bevor Sie das Buch zu den Akten legen, **sollten Sie sich also noch diese Kleinigkeiten ansehen**.

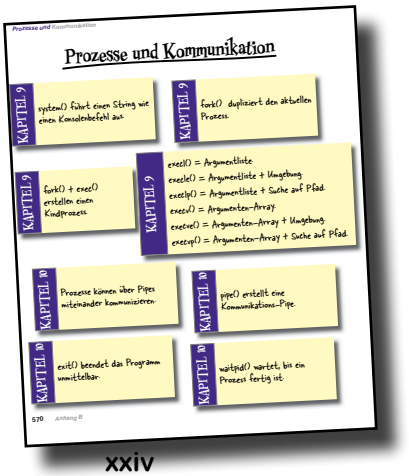


Anhang B: C-Themen

Zusammenfassungen im Überblick

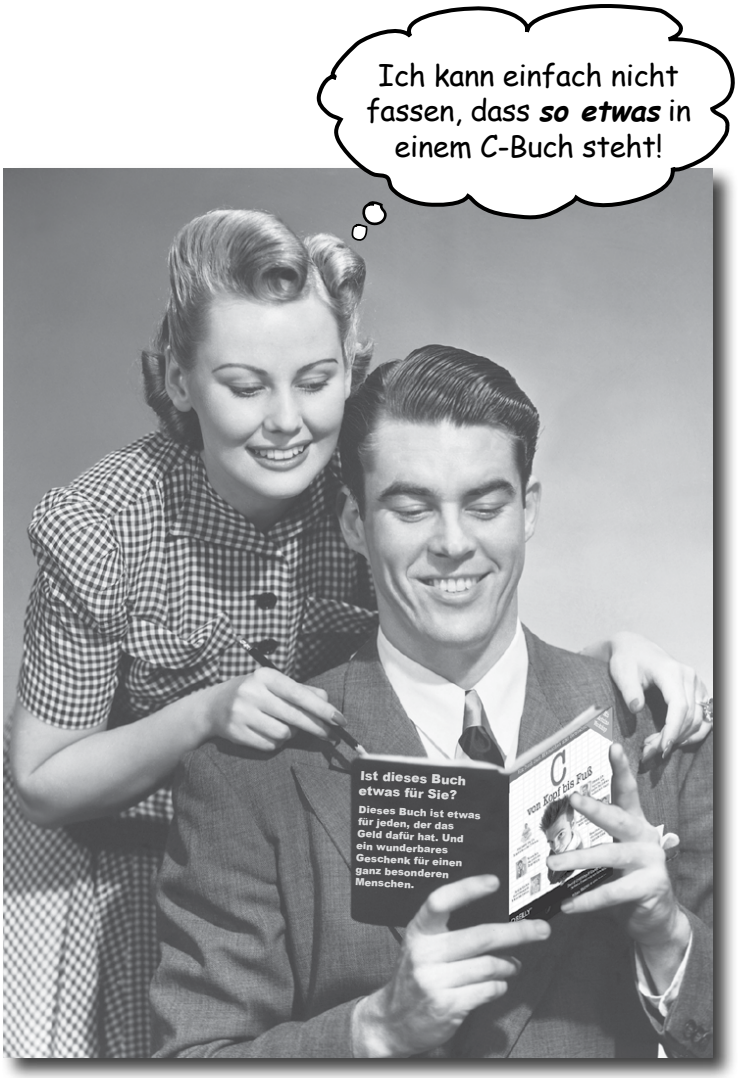
Träumen Sie auch davon, das gesamte C-Wissen wäre an einem einzigen Ort vorzufinden?

Das ist eine Zusammenfassung aller C-Themen und Prinzipien, die wir in diesem Buch behandelt haben. Werfen Sie einen Blick darauf und schauen Sie, ob Sie sich alle eingeprägt haben. Neben den Merksätzen steht jeweils das Kapitel, aus dem sie kommen, damit Sie leicht an die entsprechende Stelle zurückkehren können, um Ihrem Gedächtnis auf die Sprünge zu helfen. Vielleicht sollten Sie diese Seiten sogar herausschneiden und bei sich an die Wand hängen.



Wie man dieses Buch benutzt

Einführung



In diesem Abschnitt beantworten wir die brennende Frage:
»Und? Warum STEHT so was in einem C-Buch?«

Für wen ist dieses Buch?

Wenn Sie alle folgenden Fragen mit »Ja« beantworten können ...

1. Beherrschen Sie bereits eine andere Programmiersprache?
2. Möchten Sie programmieren lernen, damit Sie der neue Stern am Softwarehimmel sind, die Millionen scheffeln und sich bald auf Ihrer eigenen Insel zur Ruhe setzen können?

Gut, das ist vielleicht etwas übertrieben, aber wir alle haben ja mal klein angefangen, oder?

3. Ziehen Sie es vor, tatsächlich etwas zu tun? Die Dinge anzuwenden, die Sie gelernt haben? Ist das besser, als stundenlang jemandem zuzuhören, der sich in einem endlosen Vortrag über irgendwas ergeht?

... dann ist dieses Buch etwas für Sie.

Wer sollte eher die Finger von diesem Buch lassen?

Wenn Sie eine dieser Fragen mit »Ja« beantworten müssen ...

1. Suchen Sie nach einer knappen Einführung in oder eine Referenz zu C?
2. Würden Sie sich lieber von 15 kreischenden Affen die Zehennägel ziehen lassen, als einmal etwas Neues auszuprobieren? Sind Sie der Ansicht, dass ein Buch zu C *wirklich alles* behandeln muss und richtig gut eigentlich nur sein kann, wenn es den Leser zu Tode langweilt?

... dann ist dieses Buch **nicht** das richtige für Sie.

[Anmerkung aus dem Marketing: Dieses Buch ist etwas für jeden, der eine Kreditkarte besitzt. Auch Barzahlung ist möglich.]

Wir wissen, was Sie gerade denken.

»Wie kann *das* ein ernsthaftes Buch zu C sein?«

»Was sollen all die Abbildungen?«

»Kann ich auf diese Weise wirklich *lernen*?«

Und wir wissen, was Ihr *Gehirn* gerade denkt.

Ihr Gehirn lechzt nach Neuem. Es ist ständig dabei, Ihre Umgebung abzusuchen, und es *wartet* auf etwas Ungewöhnliches. So ist es nun einmal gebaut, und es hilft Ihnen zu überleben.

Also, was macht Ihr Gehirn mit all den gewöhnlichen, normalen Routinesachen, denen Sie begegnen? Es tut alles in seiner Macht stehende, damit es dadurch nicht bei seiner *eigentlichen* Arbeit gestört wird: Dinge zu erfassen, die wirklich *wichtig* sind. Es gibt sich nicht damit ab, die langweiligen Sachen zu speichern, sondern lässt diese gar nicht erst durch den »Dies-ist-offensichtlich-nicht-wichtig«-Filter.

Woher *weiß* Ihr Gehirn denn, was wichtig ist? Nehmen Sie an, Sie machen einen Tagesausflug und ein Tiger springt vor Ihnen aus dem Gebüsch: Was passiert dabei in Ihrem Kopf und Ihrem Körper?

Neuronen feuern. Gefühle werden angekurbelt. *Chemische Substanzen durchfluten Sie.*

Und so weiß Ihr Gehirn:

Dies muss wichtig sein! Vergiss es nicht!

Aber nun stellen Sie sich vor, Sie sind zu Hause oder in einer Bibliothek. In einer sicheren, warmen, tigerfreien Zone. Sie lernen. Bereiten sich auf eine Prüfung vor. Oder Sie versuchen, irgendein schwieriges Thema zu lernen, von dem Ihr Chef glaubt, Sie bräuchten dafür eine Woche oder höchstens zehn Tage.

Da ist nur ein Problem: Ihr Gehirn versucht, Ihnen einen großen Gefallen zu tun. Es versucht, dafür zu sorgen, dass diese *offensichtlich* unwichtigen Inhalte nicht knappe Ressourcen verstopfen. Ressourcen, die besser dafür verwendet würden, die wirklich *wichtigen* Dinge zu speichern. Wie Tiger. Wie die Gefahren des Feuers. Die späte Einsicht, dass Sie nie diese »Party-Bilder« auf Ihre Facebook-Seite hätten stellen dürfen. Und es gibt keine einfache Möglichkeit, Ihrem Gehirn zu sagen: »Hey, Gehirn, vielen Dank, aber egal, wie langweilig dieses Buch auch ist und wie klein der Ausschlag auf meiner emotionalen Richterskala gerade ist, ich *will* wirklich, dass du diesen Kram behältst.«

Wir stellen uns unseren Leser als einen aktiv Lernenden vor.

Also, was ist nötig, damit Sie etwas *lernen*? Erst einmal müssen Sie es *aufnehmen* und dann dafür sorgen, dass Sie es nicht wieder *vergessen*. Es geht nicht darum, Fakten in Ihren Kopf zu schieben. Nach den neuesten Forschungsergebnissen der Kognitionswissenschaft, der Neurobiologie und der Lernpsychologie gehört zum *Lernen* viel mehr als nur Text auf einer Seite. Wir wissen, was Ihr Gehirn anmacht.

Einige der Lernprinzipien dieser Buchreihe:

Bilder einsetzen. An Bilder kann man sich viel besser erinnern als an Worte allein und lernt so viel effektiver (bis zu 89 % Verbesserung bei Abrufbarkeit- und Lerntransferstudien). Außerdem werden die Dinge dadurch verständlicher. **Text in oder neben die Grafiken setzen,** auf die sie sich beziehen, anstatt darunter oder auf eine andere Seite. Die Leser werden auf den Bildinhalt bezogene Probleme dann mit *doppelt* so hoher Wahrscheinlichkeit lösen können.

Verwenden Sie einen gesprächsorientierten Stil mit persönlicher Ansprache. Nach neueren Untersuchungen haben Studenten nach dem Lernen bei Tests bis zu 40 % besser abgeschnitten, wenn der Inhalt den Leser direkt in der ersten Person und im lockeren Stil angesprochen hat statt in einem formalen Ton. Halten Sie keinen Vortrag, sondern erzählen Sie Geschichten. Benutzen Sie eine zwanglose Sprache. Nehmen Sie sich selbst nicht zu ernst. Würden *Sie* einer anregenden Unterhaltung beim Abendessen mehr Aufmerksamkeit schenken oder einem Vortrag?

Bringen Sie den Lernenden dazu, intensiver nachzudenken. Mit anderen Worten: Falls Sie nicht aktiv Ihre Neuronen strapazieren, passiert in Ihrem Gehirn nicht viel. Ein Leser muss motiviert, begeistert und neugierig sein und angeregt werden, Probleme zu lösen, Schlüsse zu ziehen und sich neues Wissen anzueignen. Und dafür brauchen Sie Herausforderungen, Übungen, zum Nachdenken anregende Fragen und Tätigkeiten, die beide Seiten des Gehirns und mehrere Sinne einbeziehen.

Ziehen Sie die Aufmerksamkeit des Lesers auf sich – und behalten Sie sie. Wir alle haben schon Erfahrungen dieser Art gemacht: »Ich will das wirklich lernen, aber ich kann einfach nicht über Seite 1 hinaus wach bleiben.« Ihr Gehirn passt auf, wenn Dinge ungewöhnlich, interessant, merkwürdig, auffällig, unerwartet sind. Ein neues, schwieriges, technisches Thema zu lernen, muss nicht langweilig sein. Wenn es das nicht ist, lernt Ihr Gehirn viel schneller.

Sprechen Sie Gefühle an. Wir wissen, dass Ihre Fähigkeit, sich an etwas zu erinnern, wesentlich von dessen emotionalem Gehalt abhängt. Sie erinnern sich an das, was Sie bewegt. Sie erinnern sich, wenn Sie etwas *fühlen*. Nein, wir erzählen keine herzzerreißenden Geschichten über einen Jungen und seinen Hund. Was wir erzählen, ruft Überraschungs-, Neugier-, Spaß- und Was-soll-das?-Emotionen hervor und dieses Hochgefühl, das Sie beim Lösen eines Puzzles empfinden oder wenn Sie etwas lernen, das alle anderen schwierig finden. Oder wenn Sie merken, dass Sie etwas können, was dieser »Ich-bin-ein-besserer-Techniker-als-du«-Typ aus der Technikabteilung *nicht kann*.

Metakognition: Nachdenken übers Denken

Wenn Sie wirklich lernen möchten, und zwar schneller und nachhaltiger, dann schenken Sie Ihrer Aufmerksamkeit Aufmerksamkeit. Denken Sie darüber nach, wie Sie denken. Lernen Sie, wie Sie lernen.

Wie könnte ich mein Gehirn wohl dazu kriegen, diesen Kram zu behalten ...

Die meisten von uns haben in ihrer Jugend keine Kurse in Metakognition oder Lerntheorie gehabt. Es wurde von uns *erwartet*, dass wir lernen, aber nur selten wurde uns auch *beigebracht*, wie man lernt.

Wir nehmen aber an, dass Sie wirklich lernen möchten, wie man in C programmiert, wenn Sie dieses Buch in den Händen halten. Und wahrscheinlich möchten Sie nicht viel Zeit aufwenden. Und Sie wollen sich an das *erinnern*, was Sie lesen, und es anwenden können. Und deshalb müssen Sie es *verstehen*. Wenn Sie so viel wie möglich von diesem Buch profitieren wollen oder von irgendeinem anderen Buch oder einer anderen Lernerfahrung, übernehmen Sie Verantwortung für Ihr Gehirn. Ihr Gehirn im Zusammenhang mit diesem Lernstoff.

Der Trick besteht darin, Ihr Gehirn dazu zu bringen, neuen Lernstoff als etwas wirklich Wichtiges anzusehen. Als entscheidend für Ihr Wohlbefinden. So wichtig wie ein Tiger. Andernfalls stecken Sie in einem dauernden Kampf, in dem Ihr Gehirn sein Bestes gibt, um die neuen Inhalte davon abzuhalten, hängen zu bleiben.

Wie bringen Sie also Ihr Gehirn dazu, die C-Programmierung für so wichtig zu halten wie einen Tiger?

Da gibt es den langsamen, ermüdenden Weg oder den schnelleren, effektiveren Weg. Der langsame Weg geht über bloße Wiederholung. Natürlich ist Ihnen klar, dass Sie lernen und sich sogar an die langweiligsten Themen erinnern *können*, wenn Sie sich die gleiche Sache immer wieder einhämmern. Wenn Sie nur oft genug wiederholen, sagt Ihr Gehirn: »Er hat zwar nicht das *Gefühl*, dass das wichtig ist, aber er sieht sich dieselbe Sache *immer und immer wieder* an – dann muss sie wohl wichtig sein.«

Der schnellere Weg besteht darin, **alles zu tun, was die Gehirnaktivität erhöht**, vor allem verschiedene Arten von Gehirnaktivität. Eine wichtige Rolle dabei spielen die auf der vorhergehenden Seite erwähnten Dinge – alles Dinge, die nachweislich helfen, dass Ihr Gehirn *für* Sie arbeitet. So hat sich z. B. in Untersuchungen gezeigt: Wenn Wörter *in* den Abbildungen stehen, die sie beschreiben (und nicht irgendwo anders auf der Seite, z. B. in einer Bildunterschrift oder im Text), versucht Ihr Gehirn, herauszufinden, wie die Wörter und das Bild zusammenhängen, und dadurch feuern mehr Neuronen. Und je mehr Neuronen feuern, umso größer ist die Chance, dass Ihr Gehirn mitbekommt: Bei dieser Sache lohnt es sich, aufzupassen, und vielleicht auch, sich daran zu erinnern.

Ein lockerer Sprachstil hilft, denn Menschen tendieren zu höherer Aufmerksamkeit, wenn ihnen bewusst ist, dass sie ein Gespräch führen – man erwartet dann ja von ihnen, dass sie dem Gespräch folgen und sich beteiligen. Das Erstaunliche daran ist: Es ist Ihrem Gehirn ziemlich egal, dass die »Unterhaltung« zwischen Ihnen und einem Buch stattfindet! Wenn der Schreibstil dagegen formal und trocken ist, hat Ihr Gehirn den gleichen Eindruck wie bei einem Vortrag, bei dem in einem Raum passive Zuhörer sitzen. Nicht nötig, wach zu bleiben.

Aber Abbildungen und ein lockerer Sprachstil sind erst der Anfang.

Das haben WIR getan:

Wir haben ***Bilder*** verwendet, weil Ihr Gehirn auf visuelle Eindrücke eingestellt ist, nicht auf Text. Soweit es Ihr Gehirn betrifft, sagt ein Bild *wirklich* mehr als 1.024 Worte. Und dort, wo Text und Abbildungen zusammenwirken, haben wir den Text *in* die Bilder eingebettet, denn Ihr Gehirn arbeitet besser, wenn der Text *innerhalb* der Sache steht, auf die er sich bezieht, und nicht in einer Bildunterschrift oder irgendwo vergraben im Text.

Wir haben ***Redundanz*** eingesetzt, d. h. dasselbe auf *unterschiedliche* Art und mit verschiedenen Medientypen ausgedrückt, damit Sie es über *mehrere Sinne* aufnehmen. Das erhöht die Chance, dass die Inhalte an mehr als nur einer Stelle in Ihrem Gehirn verankert werden.

Wir haben Konzepte und Bilder in ***unerwarteter*** Weise eingesetzt, weil Ihr Gehirn auf Neuigkeiten programmiert ist. Und wir haben Bilder und Ideen mit zumindest *etwas* ***emotionalem*** *Charakter* verwendet, weil Ihr Gehirn darauf eingestellt ist, auf die Biochemie von Gefühlen zu achten. An alles, was ein *Gefühl* in Ihnen auslöst, können Sie sich mit höherer Wahrscheinlichkeit erinnern, selbst wenn dieses Gefühl nicht mehr ist als ein bisschen ***Belustigung***, ***Überraschung*** *oder* ***Interesse.***

Wir haben einen ***umgangssprachlichen Stil*** mit direkter Anrede benutzt, denn Ihr Gehirn ist von Natur aus aufmerksamer, wenn es Sie in einer Unterhaltung wähnt, als wenn es davon ausgeht, dass Sie passiv einer Präsentation zuhören – sogar dann, wenn Sie *lesen*.

Wir haben mehr als 80 ***Aktivitäten*** für Sie vorgesehen, denn Ihr Gehirn lernt und behält von Natur aus besser, wenn Sie Dinge ***tun,*** als wenn Sie nur darüber *lesen*. Und wir haben die Übungen zwar anspruchsvoll, aber doch lösbar gemacht, denn so ist es den meisten Lesern am liebsten.

Wir haben ***mehrere unterschiedliche Lernstile*** eingesetzt, denn vielleicht bevorzugen *Sie* ein Schritt-für-Schritt-Vorgehen, während jemand anders erst einmal den groben Zusammenhang verstehen und ein Dritter einfach nur ein Codebeispiel sehen möchte. Aber ganz abgesehen von den jeweiligen Lernvorlieben profitiert *jeder* davon, wenn er die gleichen Inhalte in unterschiedlicher Form präsentiert bekommt.

Wir liefern Inhalte für ***beide Seiten Ihres Gehirns,*** denn je mehr Sie von Ihrem Gehirn einsetzen, umso wahrscheinlicher werden Sie lernen und behalten und umso länger bleiben Sie konzentriert. Wenn Sie mit einer Seite des Gehirns arbeiten, bedeutet das häufig, dass sich die andere Seite des Gehirns ausruhen kann; so können Sie über einen längeren Zeitraum produktiver lernen.

Und wir haben ***Geschichten*** und Übungen aufgenommen, die ***mehr als einen Blickwinkel repräsentieren,*** denn Ihr Gehirn lernt von Natur aus intensiver, wenn es gezwungen ist, selbst zu analysieren und zu beurteilen.

Wir haben ***Herausforderungen*** eingefügt: in Form von Übungen und indem wir ***Fragen*** stellen, auf die es nicht immer eine eindeutige Antwort gibt, denn Ihr Gehirn ist darauf eingestellt, zu lernen und sich zu erinnern, wenn es an etwas *arbeiten* muss. Überlegen Sie: Ihren *Körper* bekommen Sie ja auch nicht in Form, wenn Sie nur die Leute auf dem Sportplatz *beobachten*. Aber wir haben unser Bestes getan, um dafür zu sorgen, dass Sie – wenn Sie schon hart arbeiten – an den *richtigen* Dingen arbeiten. Dass Sie ***nicht einen einzigen Dendriten darauf verschwenden,*** ein schwer verständliches Beispiel zu verarbeiten oder einen schwierigen, mit Fachbegriffen gespickten oder übermäßig komprimierten Text zu analysieren.

Wir haben ***Menschen*** eingesetzt. In Geschichten, Beispielen, Bildern usw. – denn *Sie sind* ein Mensch. Und Ihr Gehirn schenkt *Menschen* mehr Aufmerksamkeit als *Dingen*.