THOMAS MARBELLA

Leseprobe Einsteiger

CODE TO CASHFLOW

Die 5 Level zu mehr Marktwert
& Freiheit im AI-Zeitalter

ForwardVerlag

Intro

„*Aller Anfang ist schwer*“, … gerade 2026, wo der AI Hype omnipräsent ist und es Berufseinsteigern mehr als schwer macht, Fuß zu fassen. Ging’ mir damals ähnlich, als 2008 die Finanzkrise dafür sorgte, dass kaum jemand mehr Einsteiger einstellen wollte. Kurz: I feel you. Been there, done that.

Was im gleichen Atemzug auch gesagt werden **muss**: Als Entwickler hast du eines der schönsten Handwerke, die es am Markt gibt. Das Gefühl, wenn du komplexe Architekturen mit dutzenden Klassen parallel umschreibst und der Compiler grünes Licht gibt. Wenn du quasi Gott spielen kannst und Dinge aus dem Nichts erschaffst. Und letztlich das befriedigende Gefühl, wenn dein Code das Leben von hunderten, tausenden oder gar Millionen Menschen ein Stückchen besser macht. **Das** ist ein Lebensgefühl!

Sich während seines Studiums oder seiner Ausbildung darüber den Kopf zu zerbrechen, lohnt allerdings kaum und ist nur verschwendete Energie. Sieh’ lieber zu, dass du deinen Abschluss gut hinkriegst und erste Schritte in eine Spezialisierung gehst. Nichts, was in Stein gemeißelt ist, aber du wirst sicher schnell merken, ob dir eher Dinge wie UI in React oder Backend in Java liegen, oder gar exotische Dinge wie Cobol, Haskell oder Scala. Alles zu seiner Zeit, die im Übrigen immer für dich spielt - ganz entspannt.

Die Lernkurve als Developer kann je nach Technologie und Paradigmen am Anfang ganz schön erschlagend sein. Egal ob Objektorientierung, Design- und Architektur-Pattern, Clean Code, CI/CD, funktionale Programmierung oder gar AI Assisted Coding: mit etwas Geduld kommen die Aha-Momente und du wirst sukzessive all diese Dinge, oft samt kryptischen Abkürzungen verstehen lernen.

Ob du dich für eine Ausbildung, ein Studium oder gar eine Alternative (Bootcamps/Quereinstieg) entscheidest, obliegt dir und deinen Präferenzen. Die Vor- und

Nachteile werden in diesem Kapitel natürlich im Detail erklärt. Für dich wichtig ist eher, dass du am Ende formal auf dem Papier einen Abschluss hast, da Deutschland immer noch sehr konservativ ist und auf sowas setzt. Was du letztlich während deiner Ausbildungszeit lernst und wie praxisnah das am Ende wirklich ist, ist ein anderes Thema. Viele Studiengänge (gerade an Unis) geben dir alle Theorie mit, während die effektive Praxis nicht selten auf der Strecke bleibt. Damit du deinen Karrierestart sauber hinkriegst, solltest du von Anfang an vor allem eins tun: oft und viel Coden. Am besten täglich und an Software, die dir einen praktischen Nutzen und Mehrwert gibt.

Während ich damals meine Ausbildung gemacht habe, hatte ich einige kleine Projekte im Team entwickelt. Den richtigen „Durchbruch“ hatte ich aber erst, als ich ein kleines Programm geschrieben habe, was mir persönlich dienen sollte: ein Warcraft3 Bot. :,)
So profan die Idee klingen mag, für mich war das ein riesengroßes Ding! Ein Bot, der mir damals dabei half, dass ich ein cooles, seltenes Avatar bekommen kann. Motivation? Check! Lernkurve? Check! Massig mittelmäßiger Code, aber Spaß dran? Check! Das Wichtigste: ich habe jeden Tag ein bisschen dazu gelernt und weiter gemacht. Damals sogar noch in C++ auf Windows XP!

Dass du deinen persönlichen Weg zum Erfolg findest, ist am Anfang, genau wie später in deiner Karriere das A und O. Egal ob als Student und Angestellter, oder später als Freelancer oder gar Unternehmer mit Millionen Umsätzen. Lasse dich inspirieren, probiere Dinge aus, bleib’ diszipliniert dran (!) und finde den Weg, der zu dir passt.

Wirklich falsche Entscheidungen kannst du auf deinem Weg selten treffen, außer eben nichts zu tun und in deiner Komfortzone zu verweilen, ohne den Status Quo zu hinterfragen. Aber gut - du hältst dieses Buch in deinen Händen, das ist der erste Schritt in die richtige Richtung.

Je früher in deiner Karriere du verstehst, was wichtig ist, desto früher kannst du daran arbeiten. Gerade in Deutschland herrscht(e) die Mentalität, gute Abschlüsse zu machen und dann im Konzern Karriere zu machen - und das als erfolgreiche Karriere zu definieren. Die Realität in 2026 könnte kaum anders sein: Softskills, Netzwerk und unternehmerisches und finanzielles Verständnis werden kaum gelehrt, sind aber mittelfristig von großer Bedeutung. All diese Themen gehen wir nacheinander durch!

TL:DR - „*Welcome to the grind!*"

Soft Skills, die den Unterschied machen

Selbst mit (sehr) gutem Abschluss sind die meisten noch recht weit davon entfernt, alle wichtigen Skills verinnerlicht zu haben. Was es nicht einfacher macht: „*Du weißt nicht, was du nicht weißt*". Kurz: Viele haben ihre potenziellen Defizite nicht mal auf dem Schirm. Damit dir das nicht passiert und du so früh wie möglich deine Skills entwickeln kannst, hier ein kleiner Überblick:

Kommunikation: der offensichtlichste und wichtigste Skill von allen! Was aber wirklich gute Kommunikation ausmacht, wissen viele nicht. Gerade Nerds/Devs neigen dazu, in komplexen Sätzen mit vielen Details zu kommunizieren und kommen selten auf den Punkt. Egal ob verbal, E-Mail oder per Messenger: kommuniziere überlegt und auf den Punkt, oft ist „weniger mehr".

Lösungsorientiertes Handeln: Häufig als Buzzword missbraucht, steckt hier mehr dahinter, als viele denken. Viele Devs entwickeln Code und Ansätze, die oft unnötig komplex sind und verfehlen dadurch ihr Ziel. Dein Ansatz sollte sein: Was ist der nächste, kleinste und einfachste Schritt, der direkt in die Lösung einzahlt? Quasi so einfach wie möglich, so kompliziert wie nötig. Mehr nicht.

Empathie & Zuhören: So trivial, obwohl es trotzdem oft fehlt. Leider. In einem Gespräch wirklich aktiv (!) zuzuhören, ohne in Gedanken sich seine Antwort zu überlegen. Das macht einen sehr großen Unterschied! Nochmal: Zuhören, statt die eigene Antwort überlegen! Zudem solltest du dich immer in Empathie üben. Das wird dir langfristig viele gute Beziehungen erschaffen und dich sympathisch machen. Versuche, die Sichtweise deines Gegenübers aufrichtig zu verstehen. Stelle Fragen, höre gut zu. Die Tiefe deiner sozialen Interaktionen bekommt dadurch ein neues Level.

Proaktivität & Ownership: Dich nervt etwas, was bisher niemand angesprochen hat? Oder es gibt ein Thema, was wichtig ist, wofür sich aber keiner so richtig verantwortlich fühlt? There you go! Proaktiv Themen anzusprechen, auch wenn es manchmal Mut erfordert, ist ein Skill, der dich langfristig voranbringen wird. Auch wenn es manchmal unangenehm sein kann. Fast noch wichtiger ist Ownership: nimm die Verantwortung für ein Thema an dich! Die besten Ergebnisse entstehen durch Ownership, allein deswegen, weil eine Person mit ihrem Namen und ihrer Reputation für das Ergebnis gerade steht. Es erfordert ebenfalls Mut, proaktiv im richtigen Moment „Ich!" zu sagen, wird dich aber langfristig viel über Integrität lehren und dein Ansehen erheblich steigern.

Präsentation: Ohne jetzt mit großer Extroversion anzufangen, da viele Nerds/Devs eher introvertiert sind (ich war selbst mal ein Gaming-Kellerkind), aber brauchbare Präsentations-Skills sind mittelfristig unabdingbar. Egal ob du deine Ergebnisse im Code zeigst, oder gar auf Meetups oder Konferenzen Vorträge halten willst - dieser Skill lohnt sich! Ein „*dafür bin ich nicht der Typ*"-Mindset solltest du im Zweifelsfall sehr schnell ablegen. Am Ende ist es reine Kopfsache und Übung und keine Veranlagung. Um dich selbst gut verkaufen zu können, geht das super einher - und das wird für eine gute Karriere sehr förderlich werden. Lieber jetzt über deinen Schatten springen, als es unnötig zu prokrastinieren. Lohnt sich!

Wenn du diese 7 Skills verinnerlichst, regelmäßig anwendest und übst, verschaffst du dir einen riesigen Vorteil am Markt, der dich langfristig fast zwangsläufig zum Erfolg bringt. Gute Entwickler gibt es da draußen viele. Sehr viele. Gute Entwickler mit den richtigen Soft Skills und dem „gewissen etwas" allerdings nur in überschaubarer Anzahl. Wenn du dran bleibst und deine Soft- als auch Hard-Skills kultivierst, wirst du recht schnell in den oberen 1% mitspielen. Logischerweise zur Folge auch finanziell.

Coding in 2026: The AI Hype is real

Reden wir über den Elefanten im Raum: „*Habe ich als Dev in Zeiten von AI, Agents und AGI überhaupt noch langfristig Potenzial? Braucht mich der Arbeitsmarkt überhaupt noch?*“ Kurz: Ja, definitiv! Don’t worry.

Lange Antwort:
Um das Thema rund um AI besser zu verstehen, reicht es oft schon mal genauer hinzuschauen, kritisch zu hinterfragen und selbst damit Dinge auszuprobieren. Die Möglichkeiten und zugleich Grenzen zu verstehen ist die Grundlage für eine kompetente Meinungsbildung, jenseits jeglichen Hypes und gefährlichen Halbwissens.

Disruptiert (schönes Wort!) AI gerade den ganzen Arbeitsmarkt? Ja.
Sind davon u. A. gut bezahlte, „teure“ Jobs, wie bei Devs betroffen? Definitiv.
Löst es alle Arbeitsaufwände, Probleme und Jobs auf dieser Welt? … sicher nicht.

Die Realität sieht dann am Ende doch etwas ernüchternder aus:
Repetitive, stumpfe Arbeiten löst AI super, keine Frage. Genauso schaffen es Automatisierungs-Tools wie Zapier, Make & Co allerdings auch seit Jahren. Gerade im deutschen Mittelstand gab es in White-Collar Jobs selten große Ambitionen zum Automatisieren - zumindest meiner Erfahrung nach. Gerade in Zeiten, in denen jede Firma sparen muss und Krisen durchlebt, werden Paradigmen hinterfragt und FOMO am Markt gestreut. Allen voran für AI als Wundermittel zum Kostensparen und als Productivity Boost.

Natürlich würde ich mir AI in meinem Alltag nicht mehr wegdenken! Weder als Dev, noch als Unternehmer, oder als wissbegieriger und neugieriger Mensch.

Der Gap an Hype und Realität liegt bei AI dennoch weit auseinander:

Von meinem Entwicklerteam bei OneCode, sowie meinem Netzwerk von CTOs oder Devs höre ich subjektiv immer dasselbe:
„AI nutze ich täglich, für 20-50% mehr Produktivität und nervigen Kleinkram. Um die Ergebnisse sinnvoll zu verwenden, braucht man trotzdem jahrelange Dev Erfahrung."

Je nachdem, welcher Studie man glauben mag, ist die gemessene Produktivität oft sogar nur on par oder je nach Branche schlechter, wenngleich subjektiv besser. Zwischenfazit für dich: gutes, wichtiges Tool, aber kein Allheilmittel. Zudem: Stumpf Kopieren ging damals schon mit Stackoverflow. Wem dann die Expertise fehlt, das Kopierte zu verstehen und anzupassen, hatte am Ende selten brauchbaren Code oder ein gutes Produkt.

Auch die Art und Weise wie man codet hat sich über die Dekaden geändert, die Prinzipien des Verstehens und kritischen Hinterfragens aber nicht:

- 2000er Jahre: Assembler, C/C++ und dicke Dokus in Büchern
- 2010er Jahre: Googeln, Stackoverflow, Communities
- 2020er Jahre: AIs befragen (Claude, Co-Pilot & Co)

Dadurch, dass Software immer komplexer wird, ist es nur angemessen, dass AI als Tool mehr Produktivität gibt. Wo man 2000 noch „krass" war, ein 3D Game herauszubringen, kriegt das 2020+ jeder Teeny mit Unity und etwas Zeit hin. Mehr Komplexität als Standard, bessere Tools um dem Herr zu werden. Damit ist AI kein Wundermittel, sondern eher das nächste Tool, um da mitzuhalten.

Gerade seitens großer Konzerne, die Innovationen pushen müssen, wird AI deswegen immer mehr gehyped. Bei VCs und Non-Techies, wie klassischen Managern, kommt das entsprechend an. Jeder halbwegs versierte CTO wird von obiger Aussage der Devs wenig abweichen, wohingegen viele CEOs ihrer FOMO erliegen und „*jetzt sofort und unbedingt*" AI machen müssen.

Als Dev langfristig keine AI zu nutzen wäre ein Wettbewerbsnachteil, aber im eigenen Interesse deiner persönlichen Produktivität wirst du es sicher nutzen. Disruption ist gut, viele am Markt (gerade in Deutschland!) haben sich zu viele Jahre auf ihren Lorbeeren ausgeruht - und haben jetzt FOMO den Anschluss zu verlieren.

Wichtig für dich als Anfänger: Lerne die Basics! Nachhaltig und gründlich! Und nutze dann ergänzend AI, aber bitte nicht andersherum. Sonst ist dein Fundament als Entwickler schon am wackeln, ehe deine Karriere überhaupt anfängt.

Kurz zu Vibe Coding: für PoCs oder Wegwerf-MVPs taugt es gut, um Thesen am User schnell zu vertesten und Feedback zu sammeln. Darüber hinaus wird es aber oft eng. Aktuellen Erfahrungswerten nach, hält ein Vibe coded PoC/MVP ca. 3-4 Wochen, ehe die Codequalität merklich nachlässt. Das wird sich in Zukunft sicher (oder hoffentlich?) bessern, deine Priorität sollte aber immer sein, kritisch zu hinterfragen und nachhaltigen Code zu schreiben.

TL:DR -> Nutze AI für dich als Dev, aber sorg' dich nicht zu sehr um den Hype.

Wo Anfangen?

Ausbildung, Studium, AI - Check. Gretchenfrage: Wie und wo starte ich jetzt als Dev?
Natürlich kannst du dich ins nächstbeste Tutorial stürzen und die 1000te Notes-App nachbauen, in einer fancy modernen Programmiersprache.
Die Wahl der Sprache, des Stacks und der Technologie an sich ist dabei erstmal alles andere als wichtig - du wirst und solltest (!) am Anfang 10+ Sprachen ausprobieren und die Basics lernen, ehe Spezialisierung relevant wird.

Das beste Beispiel: ich hatte damals in der Ausbildung in 3 Jahren 14 Sprachen kennengelernt. Angefangen von Assembler und C++, über Cobol und VBA, bis hin zu Java,

Javascript (himmelweiter Unterschied!) und PHP. Am Ende mochte ich Java am meisten, weil mir deren UIs in AWT und Swing so gut gefiel und mir MFC samt C++ doch etwas zu oldschool war.

Mein damaliges „praktisches Lernprojekt“ mit dem Bot hatte ich ja eingangs schon erwähnt. Der Mehrwert war für mich so greifbar, dass die Begeisterung fürs Coden geboren war!

Einen praktischen Nutzen in seiner Arbeit zu sehen, wird dich noch dein ganzes Leben begleiten. Motivation und Drive für deine Arbeit zu haben außerdem. Warum also nicht mit etwas Praktischem anfangen? Dein eigenes Problem zu lösen, jenseits jeglicher geführter Tutorials gibt dir genau *die* IRL Erfahrung, die du brauchst.

Du wirst Fehler machen. Viele. Sehr, sehr viele. Versprochen.
Aber genau da fängt Lernen an. Nicht in Hörsälen. Nicht in YouTube Videos. Nicht in Büchern. Praktische Anwendung ist das A und O, um ein guter Coder zu werden. Du kannst und wirst zwangsläufig erfolgreich sein, wenn du am Anfang jeden Tag dran bleibst und Code schreibst. Quantität und Übung stehen an erster Stelle.
Gleich danach das Wichtigste: besser werden! Du willst nicht 10 Jahre lang dasselbe Level haben, denselben Code schreiben, dieselben Ansätze fahren. Gut, außer du bist schon Top1% Coder, aber bis dahin ist es ein langer Weg.

Was ungemein beim Lernen hilft, sind diese 6 Strategien:

1. Lerne die Basics & Architektur deiner Sprache in- und auswendig!
2. Nochmal #1, weil’s so wichtig ist!
3. Lass deinen Code von erfahrenen Devs (kritisch!) reviewen
4. Lies fremden Code und versuch ihn aufrichtig zu verstehen
5. Paire mit anderen Devs und lasse dir Denkweisen erklären
6. Nutze bitte nicht gleich bei jeden Kleinkram AI, sondern lerne nachhaltig

Am Anfang wirst du von aller Komplexität beim Coden gar überwältigt sein. Das gehört dazu - been there, done that. Die ersten 6 Monate Ausbildung waren damals für mich die Hölle und ich habe mich schon als IT-Kaufmann gesehen, nicht als Dev.

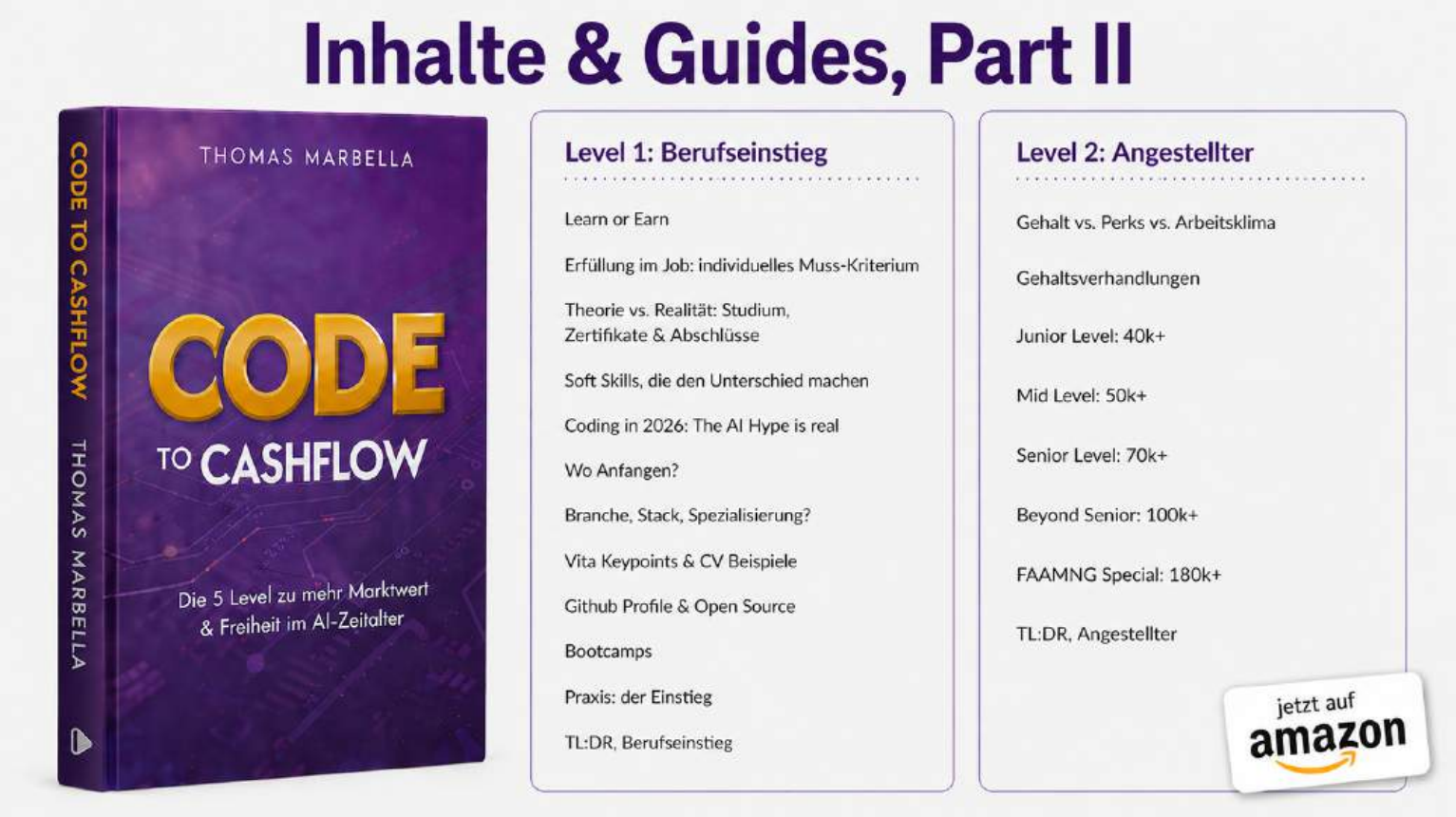
Nachdem du die Kurve zu Beginn geschafft hast, ist die größte Hürde genommen. Suche dir Kontakte (LinkedIn, Slacks, Discords), die dir ein paar Schritte voraus sind und frage proaktiv nach Hilfe, sei es beim Coden oder später bei der Jobsuche. Lohnt sich!

Alles, was danach folgt, ist meiner Meinung nach eines der schönsten „Handwerke", die es gibt: nahezu künstlerische Möglichkeiten den Code zu gestalten und Probleme zu lösen, gepaart mit Karriere- und Gehaltsmöglichkeiten, die einfacher und vielseitiger kaum woanders auffindbar sind. Und das Beste: mit diesem Buch hast du den ersten Schritt gemacht und die Entscheidung getroffen, mehr daraus zu machen. Top!

Motivation ist temporär, aber Resilienz, Persistenz und der Ehrgeiz „mehr" zu erreichen, nicht. Die drei werden deine treibenden Kräfte, damit du zum erfolgreichen Dev wirst.

Wer dran bleibt, wird belohnt - auch wenn der Weg oft steinig und lang ist.

Weitere Einsteiger Themen im Buch:



Die Leseprobe hat dir gefallen?
https://www.amazon.de/Code-Cashflow-Marktwert-Freiheit-AI-Zeitalter/dp/3987552042/

Bonus für PreOrder:

- Exklusiver Zugang zu unserer Community und -Jobmarkt
- Kostenlose 1:1 Beratungs Session mit uns (auf 50 limitiert)
- Dein Karriere und Netzwerk Boost - du musst den Weg nicht alleine gehen!