Einleitung

Ich freue mich sehr, mit Ihnen gemeinsam das Universum von Git zu erkunden. Ich erkläre dabei nicht nur die wichtigsten Befehle von Git, ich nehme Sie auch mit auf eine Reise in Open Source, DevOps und die Git-Workflows. Nach der Lektüre des Buches und vielen praktischen Übungen können Sie sicher mit Git umgehen und wissen, wie man im Team effektiv damit zusammenarbeitet. Es wird für Sie dann auch möglich sein, Beiträge zu Open-Source-Projekten zu leisten oder Ihr eigenes Projekt zu starten. Ich wünsche Ihnen viel Spaß beim Eintauchen in dieses spannende Thema.

Über dieses Buch

Ich gebe in diesem Buch eine sehr praxisorientierte Einführung in Git – mit vielen Beispielen zum Ausprobieren und Tipps aus der Praxis. Das Buch ist also kein Nachschlagewerk, das alle Optionen von Git erläutert. Es ist auch kein Theoriewerk über die Funktionsweise von Git. Es ist ein praktischer Einstieg, der Sie mit allem ausrüstet, um danach in bestimmte Themen tiefer einsteigen zu können.

Git ist ein sehr umfangreiches und mächtiges Werkzeug, welches man vielfältig in den unterschiedlichsten Szenarien einsetzen kann. Aber nur weil man etwas tun kann, heißt das noch nicht, dass man es auch tun soll. Ich fokussiere mich deshalb immer auf die Dinge, die in der Praxis gut funktionieren und Komplexität reduzieren, statt sie zu erhöhen.

Konventionen in diesem Buch

In diesem Buch finden Sie sehr viele Befehle, die über eine Kommandozeile einzugeben sind. Befehlen ist immer ein Dollarzeichen ($) vorangestellt. Vor einer Ausgabe, die von einem Programm auf der Konsole ausgegeben wird, steht immer ein Größerzeichen (›).


```
$ git status
› On branch main
› Your branch is up to date with 'origin/main'.
›
› nothing to commit, working tree clean
```

Manchmal finden Sie in diesen Teilen kursive Begriffe. In diesem Fall handelt es sich immer um Platzhalter, die Sie durch einen konkreten Wert ersetzen müssen, in nachfolgendem Beispiel durch Ihren Benutzernamen.


```
$ git clone https://github.com/benutzername/CodingDojo.git
```

Ansonsten sind kursive Begriffe immer *Fachbegriffe*, die das erste Mal in diesem Kontext verwendet werden.

Tauchen Befehle im laufenden Text auf, dann sind diese anders formatiert, wie zum Beispiel `git status`.

Menüeinträge sind immer in KAPITÄLCHEN gesetzt. Tasten und Tastenkombinationen werden wie folgt dargestellt: ↵ oder Strg+C.

Pfadangaben im Fließtext werden wie Befehle formatiert: `C:\Ordner\Ordner\Datei.txt` oder `/home/Ordner/Datei.txt`. Relative Pfade werden immer mit dem Schrägstrich geschrieben (`./Ordner/Datei.txt`), außer, sie beziehen sich explizit auf ein Betriebssystem. Wenn Sie also in der PowerShell die Beispiele ausprobieren, dann müssen Sie stattdessen die Windows Syntax (`.\Ordner\Datei.txt`) verwenden.

Webadressen sind in diesem Buch immer wie folgt formatiert: `https://writeabout.net`

Fachbegriffe

Ich verwende in diesem Buch fast immer die englischen Originalbegriffe, nachdem ich sie einmal am Anfang des Kapitels erklärt habe, weil auf der Kommandozeile auch alle Befehle in englischer Sprache eingegeben werden müssen. Außerdem ist es bei der Dokumentation oder der Fehlersuche auch hilfreich, wenn man das englische Original kennt. Das Verstehen der übersetzten Begriffe in anderen Anwendungen sollte weniger Probleme bereiten als das Erraten des englischen Begriffs, wenn man ihn nur auf Deutsch gelernt hat.

Warum Kommandozeile?

Git ist ein Kommandozeilen-Programm. Deshalb lernt man es am besten auch auf der Kommandozeile. Natürlich gibt es zahlreiche Tools, welche eine Benutzeroberfläche für Git anbieten. Aber wenn man ein Problem hat und nach einer Lösung sucht, dann findet man meistens nur Lösungen für die Kommandozeile. Der Vorteil der Kommandozeile ist auch, dass das Interface sehr konstant bleibt und sich kaum ändert. In Kapitel 5 stelle ich einige grafische Anwendungen und ihre Git-Integration vor, ansonsten verwende ich immer die Kommandozeile.

Der Name für den Haupt-Branch

Der Name für die Verzweigung (*Branch*), der allen anderen Branches zugrunde liegt, war bei Git seit jeher `master`. In der #blacklivesmatter-Bewegung im Jahr 2020 gab es zu Recht Kritik an diesem Namen, da er an die Sklaverei erinnert. Da es sich bei diesem Namen nur um eine Konvention handelt, spielt es keine Rolle, wie er lautet. Der Branch kann genauso gut `trunk` (Stamm) oder `main` (Haupt) genannt werden.

Aus Rücksicht auf Menschen, die durch diesen Namen an schlimme Zeiten unserer Geschichte erinnert werden, werde ich den Haupt-Branch in diesem Buch `main` nennen. Wenn

Sie also in einem anderen Tutorial oder in einer anderen Dokumentation die Bezeichnung `master` für einen Branch finden, dann wissen Sie, dass dieser Branch meinem `main` entspricht. Technisch hat der Name keine Auswirkungen, er ist nur eine Konvention.

Was Sie nicht lesen müssen

Die Teile und Kapitel dieses Buchs bauen thematisch aufeinander auf: von den Grundlagen über die Zusammenarbeit im Git-Universum bis zu den Vertiefungsthemen. Trotzdem ist das Buch so modular aufgebaut, dass Sie jedes Kapitel auch für sich lesen können. Wenn Sie schon Erfahrung mit Git haben, dann können Sie zum Beispiel einige Kapitel der Grundlagen überspringen. Dennoch empfehle ich Ihnen, gerade diese Kapitel zumindest querzulesen. Vielleicht findet sich ja doch noch der eine oder andere Tipp, den Sie noch nicht kennen.

Wenn Sie den Techniker sehen – den jungen Mann mit Brille –, dann erkläre ich Ihnen einen technischen Hintergrund. Manche Leser finden gerade das spannend, andere wollen sich bei der ersten Lektüre nicht damit befassen oder kennen den Hintergrund bereits. Sie können diese Passage bei Bedarf getrost überspringen. Sie ist eher für den Blick über den Tellerrand gedacht und für den weiteren Verlauf nicht unbedingt nötig.

Der erhobene Zeigefinger zeigt, dass ich einen Inhalt wiederhole, der an einer anderen Stelle schon behandelt wurde. Sie müssen den Text also nicht lesen, wenn Sie das andere Kapitel gelesen haben.

Törichte Annahmen über die Leser

Dieses Buch ist für Menschen gedacht, welche gemeinsam im Team Software erstellen und betreiben, also nicht nur für Entwickler oder gar Berufseinsteiger, die sich neu mit dem Thema befassen. Es ist genauso geeignet für Entwickler mit vielen Jahren Erfahrung in anderen Versionsverwaltungen, die jetzt einen Umstieg zu Git machen.

In Büchern der »... für Dummies«-Reihe werden anspruchsvolle Themen verständlich präsentiert, ohne dass vom Leser übertrieben viel Vorwissen erwartet wird. Diesen bewährten Weg schlage ich auch in diesem Buch ein.

Trotzdem gehe ich davon aus, dass Sie sich mit dem von Ihnen verwendeten Betriebssystem auskennen – egal, ob Windows, Linux oder macOS.

An der einen oder anderen Stelle des Buches taucht auch etwas Quellcode auf. Das ist bei einer Quellcode-Verwaltung nicht ganz zu vermeiden. Die dazugehörigen Beispiele sind dann möglichst simpel gewählt: entweder in JavaScript oder C#. Da die meisten Leser Entwickler sein werden, sollte dies kein Problem darstellen. Aber selbst ohne Programmierkenntnisse dürften diese Abschnitte kein Problem für das Verständnis des Buches sein.

Wie dieses Buch aufgebaut ist

Dieses Buch ist in vier Teile gegliedert:

Teil I: Grundlagen

In Teil I lernen Sie die Grundlagen kennen. Ich erkläre, was Git ist, wie es funktioniert und wie man es installiert. Darüber hinaus erkläre ich Ihnen alles, was Sie für einen Einstieg auf Windows, Mac oder Linux benötigen und welche Dinge Sie installieren und einrichten müssen. Sie machen Ihre ersten praktischen Übungen und lernen die Grundlagen und wichtigsten Befehle kennen. Außerdem stelle ich in diesem Teil grafische Anwendungen vor, die Ihnen das Leben einfacher machen werden.

Teil II: Zusammenarbeit

In Teil II geht es um die effektive Zusammenarbeit mit anderen Entwicklern. Ich gebe Ihnen in diesem Teil eine Einführung in die vier wichtigsten Git-Systeme für die Zusammenarbeit. Darüber hinaus lernen Sie, was Sie bei der Arbeit im Team beachten müssen und welche Richtlinien Sie einhalten sollten. Insbesondere Git-Workflows – also die genauen Arbeitsabläufe von Git in einem Team – werde ich detailliert erklären. Sie werden lernen, wie Sie mit Pull-Requests zusammenarbeiten und wie Sie Beiträge zu Open-Source-Projekten auf GitHub leisten können.

Teil III: Vertiefung

In Teil III werden ich bestimmte Themen vertiefen. Ich werde Ihnen genau zeigen, wie Git unter der Haube funktioniert. Sie werden die unterschiedlichen Merge-Strategien lernen, den Umgang mit Tags, Git-Hooks und signierten Commits.

Teil: IV: Der Top-Ten-Teil

Teil IV ist der Tipps-&-Tricks-Teil. Ich zeige Ihnen in Form von nützlichen Listen, wie Sie sich das Leben mit Git leichter machen können.

Symbole, die in diesem Buch verwendet werden

Im Buch finden Sie die folgenden Symbole. Einige kennen Sie ja schon, wenn Sie diese Einleitung gelesen haben.

Die Glühbirne kennzeichnet einen Tipp für die Praxis. Wann immer sie auftaucht, sollten Sie besonders aufmerksam lesen!

Achtung heißt es hier. Unter diesem Symbol erfahren Sie, was Sie lieber lassen sollten oder wann Sie sehr vorsichtig vorgehen sollten.

Mit der Lupe erkläre ich Ihnen einen Fachbegriff, der das erste Mal auftaucht.

Manchmal erzähle ich eine Anekdote aus meinem Berufsleben, die Sie vielleicht interessant finden. Sie ist aber nicht für den weiteren Verlauf nötig. Wenn Sie die Sprechblase sehen, dann können Sie den Absatz überspringen, falls die Anekdote Sie nicht interessiert. Aber wer wäre nicht an einer guten Geschichte interessiert?

Die Weltkugel weist auf weitere Informationen im Internet hin.

Manchmal nehme ich Dinge vorweg, da sie im aktuellen Kontext wichtig sind. Das Fernglas kennzeichnet immer einen Ausblick auf ein Thema, das in einem der nachfolgenden Kapitel genauer erklärt wird.

Wie oben bereits erwähnt wurde, erkläre ich Ihnen bei diesem Symbol den technischen Hintergrund einer Sache. Wenn Sie der Inhalt interessiert, dann lesen Sie den Text. Ansonsten machen Sie einfach weiter. Der Abschnitt ist nicht zwingend nötig, um im Buch weiterzukommen.

Auch dieses Symbol haben Sie schon kennengelernt. Wenn Sie den erhobenen Finger sehen, dann wiederhole ich einen Inhalt, der an einer anderen Stelle schon behandelt wurde.

Wie es weitergeht

Und jetzt wünsche ich Ihnen viel Spaß bei dem Einstieg in die spannende Welt der Git-Versionsverwaltung, und zwar nicht nur beim Lesen, sondern auch beim Ausprobieren!