Pair-Programming

Auf dieser Seite lernst du, wie man mit Pair-Programming zusammenarbeitet.

In den frühen Tagen der Computer war Programmieren eine Aufgabe, die man allein erledigte. Man plante das Programm im Kopf und setzte sich dann hin, um den Code zu schreiben.

Die Programme, die heutzutage geschrieben werden, sind dafür jedoch viel zu komplex. Grafische Benutzeroberflächen, die Interaktion mit Servern im Internet, Internationalisierung, Sicherheit gegen Malware-Angriffe und viele weitere Anforderungen gehören zu einer ernsthaften Softwareentwicklung, und ein Programm wird wahrscheinlich von Hunderten von Autoren erstellt.

Das bedeutet, dass Zusammenarbeit heute genauso wichtig ist wie Programmierkenntnisse. Auch wenn du nicht als Programmierer arbeitest, ist Zusammenarbeit Teil der meisten Berufe.

Es gibt verschiedene Möglichkeiten, eine Zusammenarbeit zu organisieren. Hier wird eine Technik namens Pair-Programming beschrieben – das Schreiben von Computerprogrammen in Zweierteams. Ein Partner, der **Fahrer**, steuert den Computer. Der andere, der **Navigator**, hilft dabei, Fehler zu erkennen und darauf zu achten, wie der geschriebene Code in das Gesamtprojekt passt. Der Navigator ist auch dafür zuständig, die Arbeitsanweisungen laut vorzulesen. Achtet darauf, dass ihr beide **gleich viel Zeit in jeder Rolle** verbringt und wechselt idealerweise alle fünf bis zehn Minuten.

> **So sollte man Pair-Programming nicht machen…**
>
> Der größte Fehler beim Pair-Programming ist es, wenn der Navigator abschaltet und den Fahrer die gesamte Arbeit machen lässt. Der Navigator muss aufmerksam bleiben; andernfalls wechselt ihr euch nur ab, die gesamte Arbeit alleine zu machen, anstatt die Vorteile der Zusammenarbeit zwischen zwei Personen zu nutzen.

Zusammenarbeit beim Pair-Programming

- **Redet miteinander**: Der Fahrer sollte ständig erklären, warum er oder sie jeden einzelnen Programmabschnitt verwendet. Der Navigator sollte mit Fragen unterbrechen: Wie wäre es, es anders zu machen? Wie vermeidet das Programm diesen oder jenen Fehler? Müssen wir nicht gleichzeitig diese und jene Anforderung erfüllen? Wir wiederholen hier ziemlich genau denselben Code wie dort; können wir diesen Code in einem Block verbergen und denselben Block an beiden Stellen verwenden? Und so weiter.
- **Zuhören**: Versucht, die Denkweise des anderen zu verstehen, bevor ihr widersprecht.
- **Als Team arbeiten**: Wenn du dich dabei ertappst, dass du bei Feedback defensiv wirst, erinnere dich daran, dass niemandes Code von Anfang an perfekt ist. Es ist die Aufgabe des Navigators, Probleme im Code des Fahrers zu finden – und wenn ihr die Rollen wechselt, revanchiert sich der neue Navigator. Ihr seid ein Team, kein Konkurrenzkampf.
- **Konsens anstreben**: Handelt nicht nach dem Prinzip „Wenn du das so machen darfst, darf ich das hier so machen“. Beide Partner sollten die Argumente des anderen anhören und gemeinsam nach einer Lösung suchen, die für beide akzeptabel ist.

- **Eure Meinung äußern und Unterschiede klären**: Das bedeutet nicht, dass ihr absichtlich Meinungsverschiedenheiten suchen solltet, aber falls ihr einmal unterschiedlicher Meinung seid, könnt ihr trotzdem gut miteinander auskommen. Löst den Konflikt, indem ihr zuhört und den Standpunkt des anderen versteht, anstatt einfach nachzugeben, um die Diskussion zu beenden.
- **Unabhängig von der Erfahrung die Rollen wechseln**: Lasst nicht zu, dass der erfahrenere Programmierer den anderen Partner dominiert. Stellt sicher, dass beide Partner sich in ihrer Rolle wohlfühlen und Beiträge leisten können. Ein Team ist in der Regel besser im Problemlösen als jeder Einzelne im Team.
- **Zeitmessung**: Stellt einen Wecker, um euch regelmäßig abzuwechseln!

From:
https://www.info-bw.de/ -

Permanent link:
https://www.info-bw.de/faecher:informatik:vermischtes:methode_pair_programming:start?rev=1731394805

Last update: **12.11.2024 07:00**