S. Diefenbach, N. Henze & M. Pielot (Hrsg.): Mensch und Computer 2015 Tagungsband, Stuttgart: Oldenbourg Wissenschaftsverlag, 2015, S. 319-322.

Integrierte Entwicklungsumgebung 5Code für Programmieranfänger

Markus Dahm, Frano Barnjak, Moritz Heilemann

Fachbereich Medien, Hochschule Düsseldorf

Zusammenfassung

Die integrierte Entwicklungsumgebung (IDE) 5Code unterstützt speziell Programmieranfänger. Zunächst wurde eine einfach verständliche Darstellung erarbeitet, wie man in fünf Schritten vom Problem zum Programm kommt: Lesen → Verstehen → Überlegen → Aufschreiben → Codieren. Um die kognitive Belastung der Lernenden dabei wirksam zu vermindern und so den Lernerfolg zu erhöhen wurde die IDE 5Code entwickelt, die über alle fünf Schritte den gesamten Kontext integriert – von der Aufgabenstellung über eigene Notizen bis zur Codierung. Zur Unterstützung des Verstehens können Aufgabenteile markiert und mit eigenen Überlegungen annotiert werden. Diese Notizen können in den Code übernommen und synchronisiert werden. 5Code wurde als Web-Applikation auf der Basis von Node.js implementiert. Die IDE wurde in einem Hochschul-Programmierpraktikum mit 90 Teilnehmern über ein Semester evaluiert und dabei in drei Runden iterativ verbessert.

1 Didaktische Ziele

Das Ziel des Konzepts von 5Code ist, dass Studierende erfolgreich den *vollständigen Weg vom Problem zum Programm* erlernen und anwenden können. Dabei sollen ihre kognitive Belastung möglichst klein gehalten werden. Dafür werden zunächst die Phasen des Software-Engineering in einer für Anfänger geeigneten Form dargestellt:

Lesen → Verstehen → Überlegen → Aufschreiben → Codieren

Gerade regelorientierte Anfänger in Einführungskursen zur Programmierung stützen sich stark auf die verwendeten Tools. Unglücklicherweise wird von den häufig in Praktika verwendeten Tools lediglich das Codieren und ggf. das Testen unterstützt. Das gilt für IDEs wie BlueJ oder Eclipse und für auch Microwelten wie Scratch oder Greenfoot. Die wichtigen Schritte Analyse und Design werden üblicherweise nicht angeboten und müssen von Studierenden selbständig und mit zusätzlichem kognitivem Aufwand durchgeführt werden.

2 5Code – die IDE mit komplettem Kontext

Um kognitive Kapazität optimal zu nutzen, bietet 5Code dem Nutzer den gesamten Kontext aller 5 Schritte in drei Teil-Fenstern innerhalb eines einzigen Browserfensters (Abbildung1).

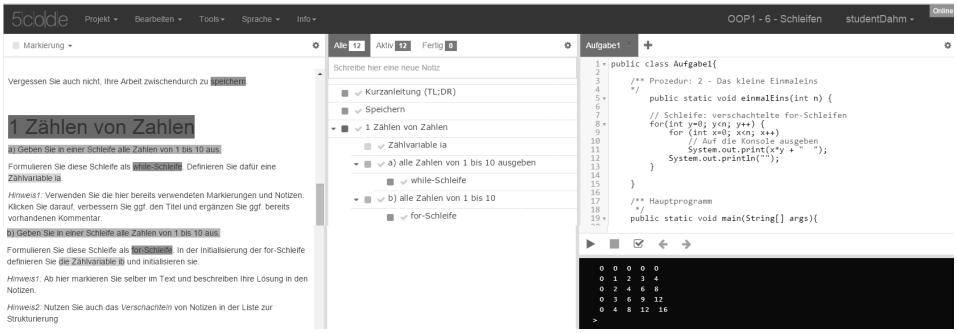

Abbildung1: Lesen → Verstehen → Überlegen → Aufschreiben → Codieren

Lernende brauchen so keine zusätzlichen Programmfenster, Papiere oder Dateien für die Aufgabenstellung und selbst geschriebenen Überlegungen, sondern haben immer alle Bestandteile gleichzeitig präsent. Der gesamte Kontext der Bearbeitung bleibt so immer erhalten. Bei der Lösung einer Aufgabe arbeitet man sich von links nach rechts vor. Daraus ergibt sich eine Aufteilung in drei vertikale Fenster, die so ein Maximum an Zeilen darstellen können. Auch auf den von Studierenden häufig verwendeten Laptops kann man gut damit in jedem Schritt arbeiten, zumal man die Breite der Fenster verstellen kann.

Lesen und Verstehen

Im linken Fenster wird die Aufgabe darstellt, wo sie jederzeit gelesen werden kann. Zur Unterstützung des Verstehens kann man, wie in jedem Texteditor üblich, beliebige Textteile selektieren. Solange das Textstück selektiert ist, kann man die Selektion auch mit einem Typ versehen und so daraus eine Markierung machen. Als Typen stehen zur Auswahl: (Text-) *Marker* sowie Implementierungs-orientierte Typen, die jeweils eine eigene Farbe haben: *Variable*, *Verzweigung*, *Schleife*, *Methode* und *Klasse*.

Überlegen und Aufschreiben

Eigene Überlegungen können wie üblich direkt im Code-Fenster als Kommentare geschrieben werden. Um dabei aber nicht zu schnell in die Phase des Codierens zu gelangen, gibt es die Möglichkeit, in einem eigenen Fenster Notizen anzulegen. Am schnellsten kann man eine Notiz direkt beim Markieren im Aufgabentext automatisch erzeugen lassen.

Notizen kann man auch durch einfaches drag&drop sowohl vertikal verschieben als auch hierarchisch anordnen und so logisch einander zuordnen (Abbildung2). Verschachtelte Notizen können zur besseren Übersicht ein- bzw. ausgeklappt werden (*folding*).

Seine Notizen kann man auch als ToDo-Liste nutzen. Wenn eine Aufgabe, die man durch eine Notiz definiert hat, fertig bearbeitet ist, kann man sie in der Liste abhaken; die Notiz erscheint dann ausgegraut. Die Notizliste kann man außerdem nach fertigen oder nach noch zu bearbeitenden Notizen filtern, so sieht man, was fertig ist und was noch getan werden soll.

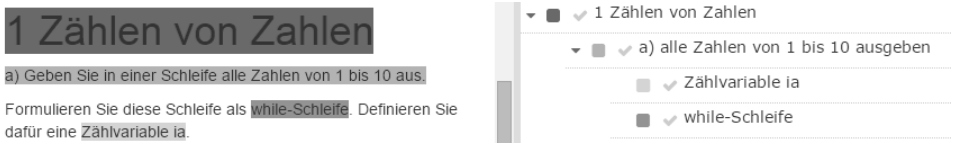

Abbildung2: Markierung im Aufgabentext und damit verbundene, bearbeitete und verschachtelte Notizen

Codieren – Tippen, Draggen, Testen und Roundtrip Engineering

So vorbereitet kann man nun (endlich) die Lösung codieren. Der Code-Teil von 5Code unterstützt aktuell Java, ist aber sprachunabhängig und wird zurzeit für HTML/CSS/ Javascript angepasst. Der Editor bietet Syntax-Highlighting, automatisches Einrücken, Folding, Zeilennummern und Tabs. Bewusst nicht geboten werden Autocompletion, Codegenerierung und Autocorrection, die für Experten wertvolle Hilfsmittel sind, da sie einfache Tätigkeiten automatisieren. Programmieranfänger müssen diese Routinetätigkeiten aber erst erlernen und verstehen; sie sollen ihnen daher nicht durch ein Tool abgenommen werden.

5Code kann immerhin das Tippen von Kommentaren abnehmen. Wer systematisch arbeitet, im Aufgabentext markiert und die erzeugten Notizen mit eigenen Beschreibungen ergänzt, wird dafür dadurch belohnt, dass er seine *Notizen direkt in den Code übernehmen* kann: Jede Notiz kann durch drag&drop in das Code-Fenster an die gewünschte Stelle gezogen werden.

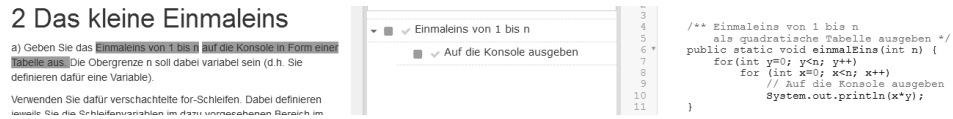

Abbildung3: Markierung → Notiz ← synchronisiert mit → Code-Kommentar

5Code hält außerdem Notizen und Code-Kommentare konsistent: Jede Änderung einer Notiz in der Liste oder im Dialog wird im daraus entstandenen Kommentar im Code-Editor sofort automatisch synchronisiert und umgekehrt (Abb3). Dieses *Roundtrip Engineering* in 5Code vermindert sowohl den kognitiven als auch den zeitlichen Aufwand der mehrfachen Eingabe.

Java-Programme können über Buttons compiliert, gestartet und gestoppt werden. Ein- und Ausgaben werden durch das Terminalfenster unterhalb des Codes gehandhabt. Fehlermeldungen des Compilers werden u.a. im Code durch Marker/Tooltipps angezeigt.

Übliche IDEs bieten Anfängern lediglich ein Fenster zum Codieren an. 5Code bietet darüber hinaus die Integration der vorangegangenen Phasen und deren Arbeitsergebnisse.

Vorbereitung und Bewertung durch Lehrende

Lehrende haben in 5Code einige Möglichkeiten zur Vorbereitung: Aufgaben erstellt und editiert man mit gängigen WYSIWYG-Funktionen zur Formatierung, Bilder oder Links können eingebunden werden. Bestehende Aufgaben können formatiert übernommen werden.

Programmieranfängern kann man in 5Code Hinweise zur Aufgabe und Lösung durch Markierungen im Aufgabentext und in den zugeordneten Notizen geben. Außerdem kann man Code vorgeben, wie ein Klassenrahmen für prozedurale Programminhalte. Analog können für OO-Aufgaben Hilfsklassen in eigenen Code-Tabs zur Verfügung gestellt werden. Lehrende können markierte Aufgaben, Notizen und Code der Lernenden online bewerten.

3 Evaluation und iterative Weiterentwicklung

5Code wurde in einem Anfängerpraktikum mit 90 Studierenden über ein Semester eingesetzt und ausführlich evaluiert. Dazu wurden im Abstand von fünf Wochen drei Umfragen mit den Tools AttrakDiff und SurveyMonkey durchgeführt sowie Rückmeldungen durch die Tutor-Innen aufgenommen. Während des laufenden Semesters konnten bereits einige wichtige Anregungen aus dem Feedback umgesetzt werden. Dadurch konnten iterativ Verbesserungen in der Gestaltung und der Funktionalität erreicht werden, was sowohl die pragmatische als auch die hedonische Qualität gesteigert hat, wie die jeweils folgende Evaluation gezeigt hat.

In den vorangegangenen Jahrgängen wurde mit Texteditor und Kommandozeile bzw. Eclipse entwickelt. Dem gegenüber zeigt die Evaluation dass mit 5Code deutlich mehr kommentiert wurde und die Unterstützung der Schritte Lesen, Verstehen, Überlegen und Aufschreiben vor allem von Anfängern überwiegend geschätzt und gerne genutzt wird. Das zeigt sich sowohl in den Umfrage-Antworten als auch in den tatsächlich erstellten Lösungen, die dadurch auch häufiger und schneller korrekt erstellt wurden und zudem auch besser verstanden wurden.

Literatur

Börstler, J. (2007). Objektorientiertes Programmieren – machen wir irgendwas falsch?, In S. Schubert (Hrsg.), *Didaktik der Informatik in Theorie und Praxis, INFOS 2007*, LNI112, Köllen Verlag, Bonn

Chandler, P., Sweller, J. (1996). Cognitive load while learning to use a computer program. *Applied Cognitive Psychology 10*, S. 151-170

Hubwieser, P. (2007). *Didaktik der Informatik*, Springer, Berlin

McCracken et al. (2001). *A multi-national, multi-institutional study of assessment of programming skills of first-year CS students*, ACM SIGCSE, Bulletin 33 (4), S. 125-140

Terwelp, S., Dahm, M. (2011). Entwicklungsumgebungen für Informatik-Anfänger. In Maximilian Eibl (Hrsg.) *Mensch und Computer 2011*, Oldenbourg Verlag

Kontakt:

www.medien.hs-duesseldorf.de/dahm, www.5code.de, markus.dahm@hs-duesseldorf.de