Steigerung der Motivation und des Lernerfolgs von ProgrammieranfängerInnen mithilfe von Gamification

Gamification in der integrierten Entwicklungsumgebung 5Code

Jennifer Rose
Fachbereich Medien
Hochschule Düsseldorf
jennifer.rose@
hs-duesseldorf.de

Markus Dahm
Fachbereich Medien
Hochschule Düsseldorf
markus.dahm@
hs-duesseldorf.de

Michael Czarnetzki
Fachbereich Medien
Hochschule Düsseldorf
michael.czarnetzki@
study.hs-duesseldorf.de

Marius Köhler
Fachbereich Medien
Hochschule Düsseldorf
marius.koehler@
study.hs-duesseldorf.de

ABSTRACT

Um ProgrammieranfängerInnen beim Erlernen von Konzepten und Fähigkeiten des Programmierens zu unterstützen, wurden sowohl ein didaktisches Konzept als auch die integrierte Entwicklungsumgebung (IDE) 5Code über mehrere Jahre hinweg entwickelt. In den letzten Iterationen wurde immer deutlicher, dass die intrinsische Motivation und Ausdauer vieler Studierender eher nachlassen. Daher wurden unter anderem einige neuartige Konzepte entwickelt, die mit aus Computerspielen bekannten Elementen Motivation und Ausdauer auf einem nachhaltig hohen Niveau halten sollen: Eine Level Map und ein Skill Tree visualisieren den Lernfortschritt sowohl in zeitlicher als auch in inhaltlicher Dimension. Erreichte Ziele werden nicht nur grafisch dargestellt, sondern führen auch zum Freischalten von speziellen Features. Im Workshop wollen wir diese und weitere Ansätze vorstellen und zur Diskussion stellen.

KEYWORDS

Gamification, Programmierung, Entwicklungsumgebung, AnfängerInnen, Motivation, Didaktik

1 Einleitung

1.1 Kontext

Die Grundlagen der Programmierung zu erlernen ist ein umfangreiches Vorhaben für AnfängerInnen. Sie sind einer hohen kognitiven Belastung durch mehrere gleichzeitig zu erfüllende Aufgaben ausgesetzt: Verständnis der Problemstellung, Entwicklung eines Lösungskonzepts und korrekte Umsetzung in der Programmiersprache sowie die korrekte Nutzung einer integrierten Entwicklungsumgebung (IDE). Diese Belastung führt nicht selten dazu, dass der Fokus der Lernenden hauptsächlich auf die reine Codierung gelegt wird, wodurch der Kontext zur ursprünglichen Aufgabenstellung sowie zur Aufgabe, zunächst eigene Lösungskonzepte zu erstellen, leicht verloren geht. Zudem besteht bei AnfängerInnen noch kein mentales Modell des „Programmierens“, was den Vorgang der Lösungsfindung zusätzlich erschwert [1][2]. Durch diese Schwierigkeiten kommt es häufig vor, dass sich Lernende die grundlegenden Konzepte nicht aneignen und damit den Programmierkurs nicht erfolgreich absolvieren können [3].

Um diesem Problem entgegenzuwirken, wurden ein didaktisches Konzept und die unterstützende integrierte Entwicklungsumgebung (IDE) 5Code für ProgrammieranfängerInnen entwickelt. Das zugrunde liegende didaktische Grundkonzept fasst alle Phasen des Entwicklungsprozesses in fünf Schritten zusammen, die auch AnfängerInnen einfach zugänglich sind: [4]

> Lesen → Verstehen → Überlegen → Aufschreiben → Codieren (kurz LVÜAC).

Natürlich gehört auch das Testen zur Software-Entwicklung, es steht aber vor allem in den ersten beiden Semestern nicht im Vordergrund. Sowohl das didaktische Konzept als auch die IDE werden im Praktikum des Moduls „Objektorientierte Programmierung 1“ im ersten Semester des Bachelors Medieninformatik an der Hochschule Düsseldorf eingesetzt. Das grundlegende Ziel besteht darin, AnfängerInnen die Vorgehensweise „*vom Problem zum Programm*“ näherzubringen. Um den Zugang dazu zu vereinfachen, werden in 5Code alle Phasen (LVÜAC) in einer Anwendung integriert. Somit ist immer der komplette Kontext verfügbar, ohne zwischen verschiedenen Anwendungen wechseln zu müssen. Zusätzlich vereinfachen Verknüpfungen von Aufgaben und Lösungen die Orientierung und damit den kompletten Weg vom Problem zum Programm. Sowohl professionelle Entwicklungsumgebungen als auch IDEs für AnfängerInnen wie Scratch oder BlueJ bieten demgegenüber ausschließlich den letzten Schritt „Codieren“ an [4].

Neben der Unterstützung von AnfängerInnen wird auch berücksichtigt, dass fortgeschrittenere ProgrammiererInnen im Praktikum mit einem erwartungskonformen Programm arbeiten können [5]. Die IDE wird fortlaufend weiterentwickelt und durch innovative, didaktisch begründete Konzepte erweitert.

1.2 Motivation für Gamification

Im Laufe mehrerer Jahre wurden etliche Konzepte und Hilfestellungen für ProgrammieranfängerInnen entwickelt, evaluiert und verbessert. Dennoch treten im Programmierpraktikum weiterhin häufig Probleme auf. Es fällt den Studierenden beispielsweise nicht selten schwer, das in der Vorlesung Gelernte auf ein konkretes Problem einer Praktikumsaufgabe zu übertragen. Durch diese gedanklichen Hürden verlängert sich die Arbeitszeit an den Aufgaben merklich. Zudem können vorhandene Hilfestellungen zum Teil nicht richtig gedeutet werden, wie z.B. Fehlermeldungen beim Ausführen von Code [6].

Um die vorhandenen Unterstützungen sowohl besser wahrzunehmen, als auch ihre Nutzung zu belohnen, sollen nun spielerische Elemente eingesetzt werden: Die Wahrnehmung und Akzeptanz von Feedback sowie die sich daraus ergebende Orientierung über den eigenen Lernstand soll durch Gamification-Ansätze verbessert werden. Gamification bezeichnet die Integration von Aspekten aus Spielen in einen spielfremden Kontext [36]. Häufig werden dabei Belohnungen verwendet, darüber hinaus umfasst das Konzept aber auch Spieletypische Elemente der Visualisierung und Interaktion. Dieses Konzept wird genutzt, um Verhalten und Motivation von Nutzern zu beeinflussen. Hier soll Gamification angewandt werden, um Lernende verstärkt intrinsisch zu motivieren. Intrinsische Motivation zeichnet sich dadurch aus, dass Handlungen zum Selbstzweck ausgeführt werden, und nicht aufgrund von externer Beeinflussung und möglichen Folgen wie eine Benotung (extrinsische Motivation) [7]. Gamification-Elemente erhöhen offensichtlich die extrinsische Motivation (z.B. durch Belohnungen), langfristig kann aber durch Fortschritte und Erfolgserlebnisse auch verstärkt intrinsische Motivation generiert werden [31]. Die Lernenden sollen dazu animiert werden, mehr lernen zu wollen, dabei (hoffentlich) noch Spaß zu empfinden, und die Aufgaben nicht nur aus dem Zwang zu bearbeiten, ihre Zulassung zur Prüfung zu erhalten.

2 Verwandte Arbeiten

Im Bereich Gamification gibt es bereits viele praktische Umsetzungen, die als Inspiration dienen können. Dabei gibt es einige zentrale Grundelemente, die häufig verwendet werden. Dazu gehören Punkte, Abzeichen und Ranglisten [8]. Diese können effektiv verschiedene Arten von Bedürfnissen ansprechen, sofern sie sinnvoll umgesetzt werden [32]. Diese Elemente werden z.B. bei Apps zur Verbesserung der Gesundheit oder Fitness eingesetzt. Sie stellen den Fortschritt der NutzerInnen dar und vergeben Abzeichen für erreichte Ziele und überwundene Herausforderungen, um die Motivation aufrecht zu erhalten. Weiterhin können Wettkämpfe mit anderen NutzerInnen ausgetragen werden, indem Leistungen in Ranglisten verglichen werden. Mit besonders guten Leistungen können auch zusätzliche Funktionen für die App freigeschaltet werden [9][10][19].

Auch im Bereich Lernen gibt es häufig Gamification-Umsetzungen, z.B. in der Sprachlernanwendung Duolingo [11]. Durch das Absolvieren von Lektionen können Erfahrungspunkte und virtuelle Währungen zum Austausch gegen Hilfsitems gesammelt werden. Richtige Antworten und längere Erfolgssträhnen werden visuell und akustisch belohnt, um die NutzerInnen zum Weitermachen zu motivieren. Das Freischalten von Abzeichen bei gesetzten Meilensteinen belohnt mit zusätzlichen Einheiten der virtuellen Währung. NutzerInnen können auch in wöchentlichen Ranglisten gegeneinander antreten, wobei diejenigen mit den meisten Erfahrungspunkten in höhere Ränge aufsteigen können.

Speziell für das Erlernen von Programmierung gibt es ebenfalls verschiedene Umsetzungen. JetBrains bietet die *JetBrains Academy* [20] als interaktive Lernplattform an. In verschiedenen Kursen können jeweils Kenntnisse zu einer Programmiersprache (z.B. Java, Kotlin oder Python) erworben und praktisch eingeübt werden. Zu jedem Kurs gibt es Achievements, eine Fortschrittsanzeige sowie eine Übersicht von allen erlernten Themen als vernetzte Karte (Map). Die Funktionalitäten können mithilfe eines Plugins auch direkt innerhalb der JetBrains-Entwicklungsumgebung IntelliJ IDEA genutzt werden [21]. Auch für die IDE Eclipse gibt es entsprechende Plugins, z.B. *CodeSkills* [33]; dieses fokussiert sich nur auf den Gamification-Aspekt und bietet 20 „Achievements" für Java-Programmierung an.

Neben solchen klassischen Lernansätzen gibt es auch verstärkt spielbasierte Umsetzungen, in denen Programmier-Puzzles gelöst oder Spiele live durch eigenen Code ergänzt werden können [22][23][24][34]. Diese sind meist an ProgrammieranfängerInnen gerichtet. Eine Applikation, die sich mit dem bereits vorgestellten Duolingo vergleichen lässt, ist Mimo [35]. Dort können kleine Aufgaben mit direktem Feedback gelöst werden, Erfolge werden in Ranglisten festgehalten und Erfolgssträhnen werden belohnt.

Doch nicht nur in Applikationen die Fortschritte und Entwicklungen messen ist der Einsatz von Gamification sinnvoll. Dropbox, ein Filehosting-Dienst [12], bietet ein einfaches Onboarding in Form einer Liste von Aufgaben („Quests"), um die Funktionalitäten kennenzulernen. Durch das Erfüllen der Aufgaben kann zusätzlicher Speicherplatz erworben werden [13]. Diese und andere Umsetzungen dienen als Inspiration, das Erlernen von Programmierung mithilfe von Gamification interessanter und motivierender zu gestalten.

3 Voraussetzungen

3.1 Zielgruppe

Um sinnvolle Unterstützungen anbieten zu können muss zunächst ermittelt werden wie sich die Zielgruppe zusammensetzt. Auf diese Weise können gezielter Lösungen angeboten werden. Befragungen ergeben regelmäßig, dass die Zusammensetzung der Lernenden im Programmierpraktikum des hier betrachteten ersten Semesters des Bachelors Medieninformatik recht divers ist. Neben absoluten ProgrammieranfängerInnen (ca. 40%, Tendenz steigend) gibt es auch erfahrenere ProgrammiererInnen (ca. 15%). Ca. 35% (Tendenz leicht steigend) sind weiblich, ca. 65% sind männlich.

Darunter gibt es meistens sehr junge (18, 19 Jahre), aber auch ältere (23-30 Jahre) Studierende.
In Bezug auf Spiele werden Erfahrung und Kenntnisse der Zielgruppe von daher auch als sehr verschieden eingeschätzt [30]. Um dennoch eine möglichst hohe Akzeptanz der Gamification-Elemente zu erreichen, wird der geplante spielerische Anteil von beliebten und wiedererkennbaren Elementen der Popkultur inspiriert (z.B. Retro-Spiele, Referenzen, …) und individualisierbar gestaltet.
Aufgrund der diversen Zielgruppe ist es wichtig, die Umsetzung für alle nützlich, hilfreich und interessant zu gestalten, unabhängig von den einzelnen Vorkenntnissen und Präferenzen. Für gamifizierte Systeme lassen sich, inspiriert von Bartles Spielertypen [27], verschiedene Typen von NutzerInnen feststellen [26]. Diese sind mehr oder weniger bereit, sich auf spielerische Aspekte einzulassen, und verfolgen dabei unterschiedliche Ziele. Demnach ist es wichtig, auf verschiedene Ausprägungen einzugehen, wie z.B. einerseits NutzerInnen, die nicht „spielen" möchten und andererseits diejenigen, die das System für sich entdecken oder ihre Fähigkeiten verbessern wollen.

3.2 Ziele

Die übergeordneten Ziele des Gamification-Konzeptes sind die Erhöhung von Motivation und Aufmerksamkeit der Lernenden. Dazu wollen wir neue, innovative Hilfestellungen beim Lösen von Programmieraufgaben und beim Lernen von Programmierkonzepten bieten.
Die Entwicklungsumgebung 5Code wird dazu mithilfe von Gamification-Elementen erweitert, sie soll den bisherigen Funktionsumfang behalten. Der spielerische Anteil ergänzt und unterstützt, die IDE dient weiter in erster Linie dazu, die gestellten Programmieraufgaben zu lösen. Das bedeutet, dass die IDE selbst nicht als „Spiel" abstrahiert werden soll (was bereits, teils aufwändig, versucht wurde [14][15][16]). Daraus folgt ebenfalls, dass der Funktionsumfang auch ohne die Spiel-Elemente sinnvoll und nutzbar sein soll. Die Studierenden sollen nicht gezwungen sein zu „spielen", also den Spiel-Anteil in Anspruch zu nehmen, sondern in eigenverantwortlicher Nutzung davon profitieren können [13].
Auch wenn Gamification einen Unterhaltungswert haben kann, soll es nicht zu einer Ablenkung vom eigentlichen Ziel des Nutzers kommen. Das Ziel im Praktikum ist es, in der Vorlesung theoretisch kennengelernte Konzepte selbst praktisch einzusetzen, d.h. Aufgaben zu lösen. Formal wird damit auch die Zulassung zur Prüfung erworben. In diesem Sinne müssen alle zusätzlich integrierten Elemente in ihrem Nutzen begründet werden; Features und Belohnungen sollen den Studierenden dabei helfen, ihr Ziel zu erreichen, und somit einen direkten Bezug zum Programmieren haben.
Zuletzt sollen die ausgewählten Spiel-Aspekte nicht zusammenhangslos integriert werden. Sie sollen einem durchgängigen Konzept folgen und sich gegenseitig beeinflussen.

4 Konzepte

Zur Steigerung von Motivation und Aufmerksamkeit der Studierenden wurden bekannte Konzepte zur Gamification ausgewählt, aber auch neuartige Ansätze auf Basis bekannter Spiel-Elemente entwickelt. Diese werden in die Entwicklungsumgebung 5Code integriert und im laufenden Semester mit Studierenden evaluiert. Die einzelnen Aspekte werden im Folgenden mit den gesetzten Zielen, sowie den bisherigen Erfahrungen begründet und im Einzelnen vorgestellt.

4.1 Bewertungen

Bewertungen stellen eine Grundlage der meisten Spiele dar, sowohl mit quantitativen als auch qualitativen Ergebnissen. So können quantitative Fortschritte und Erfolgsraten, wie z.B. „Programm liefert erwartete Ergebnisse" oder auch „Programm wird fehlerfrei kompiliert" einfach ermittelt und verglichen werden. Diese können z.B. in Form von Punkten festgehalten werden, die bei besonderen Aktionen, oder auch durch das Spielen des Spiels an sich, vergeben werden. Punkte können als Indikator für den eigenen Lernfortschritt und -erfolg dienen oder als Währung zum Erwerb von Gegenständen und ähnlichem. Durch das Sammeln von Punkten können Ränge erreicht oder neue Inhalte freigeschaltet werden.
In Bezug auf das Lernen können Punktesysteme als motivierender Faktor genutzt werden. Durch eine höhere Leistung können mehr Punkte erreicht und damit Belohnungen, z.B. in Form von neuen Inhalten oder Funktionen, erlangt werden. Dabei ist wichtig, dass diese Belohnungen in erster Linie hilfreich sind und neue Inhalte dem Kontext entsprechen:

> *To gamify a system, start by rewarding users with the thing they use the application for.* [13]

Im Bereich Programmierung können dann z.B. neue Funktionalitäten, beispielsweise Autocomplete, Syntax-Highlighting oder ähnliches, für die IDE freigeschaltet werden. So kann ein reiner Code-Editor schrittweise zu einer Programmierumgebung entwickelt werden, in welcher auch fortgeschrittene ProgrammiererInnen bekannte Elemente wiederfinden. Auf diese Weise können auch ansteigende Schwierigkeitsstufen in den Aufgaben mit immer mehr Hilfsmitteln unterstützt werden. Einerseits werden erfahrene und fleißige Studierenden mit mehr Features belohnt, andererseits werden AnfängerInnen zu Beginn nicht mit zu vielen Funktionalitäten überfordert. Auch unabhängige, unterhaltsame Elemente wie Animationen, Themes oder individuelle Avatare können als Belohnung dienen. Jedoch sollen diese, wie in den Zielen festgelegt, nicht im Vordergrund stehen.
Zusätzlich zu den quantitativen Punkten können qualitative Bewertungen, z.B. in Form von textuellen Rückmeldungen, genutzt werden. Diese dienen zur eigenen Orientierung und Feststellung des Lernbedarfs. Auch Feedback von wissenschaftlichen MitarbeiterInnen und TutorInnen im Programmierpraktikum kann in diesem Sinne hilfreich sein. Zu diesem Zweck soll den Lernenden die Möglichkeit geboten

werden, pro (Unter-)Aufgabe eigene Notizen festzuhalten. Dazu werden an dieser Stelle auch erreichte Punkte für verschiedene Aspekte eingetragen, die bei Abgabe der Aufgabe vergeben werden. Dafür gibt es kein Rechte-System zur Verwaltung; alle Bewertungen werden *eigenverantwortlich* eingetragen und haben keinen Einfluss auf die Endbewertung im Praktikum. So soll die intrinsische Motivation zur Qualitätssteigerung der eigenen Lösungen erhöht werden. Insgesamt kommen diese selbst erstellten Bewertungen einer Art implizit geschriebenen Lerntagebuch nahe. Die Lernenden erstellen sich so eine Übersicht zum aktuellen Lernfortschritt, ohne sich selbstständig Formulierungen ausdenken zu müssen. Nach eigener Erfahrung haben Studierende oft das Problem, ihre fehlenden abstrakten Kenntnisse in Worte zu fassen und als konkrete Fragen zu formulieren. Diese Hürde wird hier umgangen, da Bewertungen und Rückmeldungen sich immer auf konkrete Aufgabenstellungen beziehen. Darüber hinaus können konkrete Fragestellungen bereitgestellt werden, für weitere Inspiration beim Ausfüllen des Lerntagebuchs. Besonders reflexive Fragestellungen können dabei hilfreich sein, da sie zu neuen Perspektiven der Problemlösung anregen können [28].

Abbildung 1: (Selbst-)Bewertung als eine Art Lerntagebuch [6]

Auf einen direkten Vergleich der Lernenden untereinander in Form von gruppenweit sichtbaren Ranglisten wird vorerst bewusst verzichtet, da diese bei schlechteren Ergebnissen auch leicht zu Demotivation führen können [8]. In den aktuellen Anwendungen sind die Gruppen auch zu divers bezüglich der Vorkenntnisse, als dass ein objektiver Vergleich möglich wäre. Es ist daher sinnvoller, die Studierenden zu eigenmotiviertem Lernen zu anzuregen, als sie basierend auf Punktzahlen zu vergleichen und als „besser" oder „schlechter" darzustellen.

4.2 Level Map

Die Einträge des Lerntagebuchs, sowie weitere Informationen, sind unter anderem über die sogenannte Level Map zugänglich. Diese stellt alle Aufgaben des Semesters als Spiellevel dar, die bis zum Ende gelöst werden müssen. Dabei werden bereits gelöste Aufgaben entsprechend gekennzeichnet. Auf diese Weise ist es möglich, den bisherigen Fortschritt des Semesters auf einen Blick einsehen zu können. Die Level-Übersicht bietet nicht nur eine quantitative, zeitlich orientierte Fortschrittsanzeige als Anzahl gelöster Aufgaben; mit der Anbindung an das Lerntagebuch ist gleichzeitig auch eine qualitative Bewertung des Fortschritts integriert.

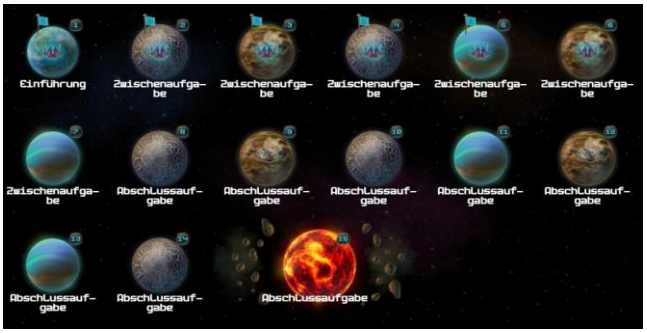

Abbildung 2: Level Map [6]

Die Level Map wird ähnlich wie in vielen Spielen dargestellt. Um eine möglichst hohe Akzeptanz bei der Zielgruppe zu erzielen können mehrere Erscheinungsformen ausgewählt werden [6], sodass die unterschiedlichen Präferenzen von möglichst vielen NutzerInnen abgebildet werden können. Im oben beschriebenen Sinne der reinen Ergänzung der IDE durch Gamification-Elemente ist die Ansicht in Form von Leveln optional: Wer den spielerischen Anteil nicht in Anspruch nehmen möchte, wird nicht dazu gezwungen, sondern kann eine gewöhnliche Listenansicht von Aufgaben zur Information und Navigation verwenden.

4.3 Skill Tree

Parallel zur Darstellung zeitlichen Verlauf des Lernens in der Level Map werden die erlernbaren und selbst erlernten Kenntnisse und Fähigkeiten des Semesters festgehalten. Alle erreichbaren und erreichten Skills werden in einem Skill Tree visualisiert. In diversen Spielen (z.B. Skyrim, Assassin's Creed, Watch Dogs [17]) wird ein solcher Baum verwendet, um die Skills und Entwicklungsmöglichkeiten eines Charakters darzustellen. Der Fokus liegt in Spielen meist darauf, sich auf ausgewählte Teilbäume zu konzentrieren. In diesem Fall ist jedoch das Ziel, zum Ende des Semesters alle Elemente des gesamten Skill Trees „freizuschalten".

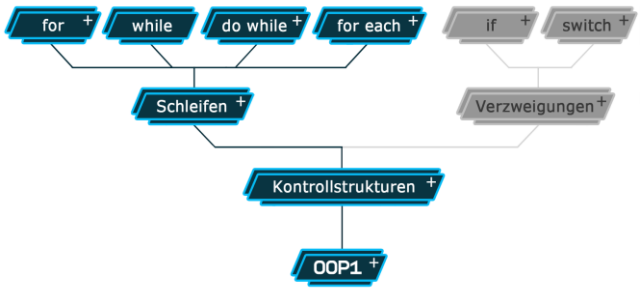

Abbildung 3: Skill Tree

Mithilfe des Skill Trees werden allgemeine Ziele („*Programmieren lernen*“) auf konkrete, erreichbare Teilziele aufgetrennt („*OOP – Kontrollstrukturen – Schleife – while – Variable als Bedingung*“). Auf diese Art wird das einzelne Lernziel als greifbarer, und hoffentlich auch weniger anspruchsvoll oder sogar erdrückend, von den Studierenden wahrgenommen.
Eine besondere Herausforderung ist dabei die eher große Menge von Skills, die hierarchisch geordnet dargestellt werden sollen. Sehr ähnlich wie in einer Mindmap [25] können einzelne Teilbäume ein- und ausgeklappt werden, außerdem können verwandte Skills, z.B. Schleifen-Varianten aggregiert dargestellt werden, um Platz auf dem Bildschirm zu sparen.
Der Skill Tree kann so den Studierenden dabei helfen, ihren Lernfortschritt inhaltlich überblicken und die einzelnen Teilschritte besser nachvollziehen zu können. Zusätzlich bieten die einzelnen Skills einen Link mit einer Erklärung; das gezielte Nachschlagen und Lernen werden so zusätzlich unterstützt.

4.4 Definition von Kompetenzen

Die in einer Aufgabe zu erlangenden Kompetenzen werden direkt im Text der einzelnen Aufgaben definiert. Die Darstellung wurde von den diversen Widgets inspiriert, wie sie in Software-Repositories verwendet werden, um Features darzustellen:

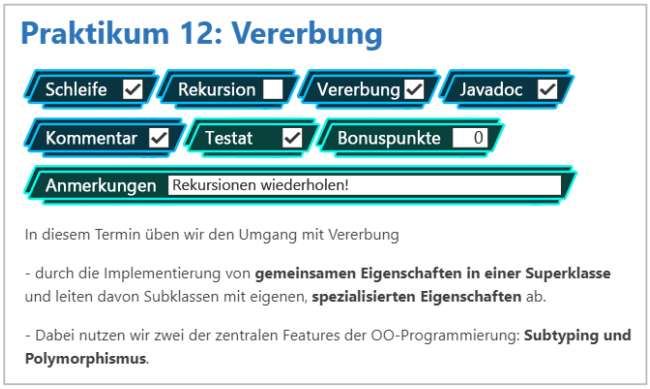

Abbildung 4: Kompetenz-Widgets im Aufgabendokument

Das Abhaken eines Widgets oder das Eintragen eines Punktewerts oder eigener Anmerkungen, wird mit der gelösten Aufgabe gespeichert. Damit können diese Fähigkeiten im Skill Tree entsprechend markiert werden, z.B. als „erreicht“, „ausbaufähig“ oder „muss noch geübt werden“.
Die Kompetenz-Widgets machen weiterhin den Studierenden explizit deutlich, welches Lernziel und welcher Zweck hinter jeder Aufgabenstellung stehen. Genau diese Lernziele sind vielen AnfängerInnen ohne jede Erfahrung sehr häufig unklar. Sie konzentrieren sich dann auf andere Features der Programmiersprache, die entweder schon als bekannt vorausgesetzt werden, oder die erst später wichtig werden. Werden die aktuellen Inhalte dagegen explizit gemacht – und das auch noch in einer attraktiven Art und Weise – werden falsche Denk-, Lern- und Arbeitsrichtungen von vorn herein vermieden.

4.5 Belohnungen

Ein häufig in Gamification verwendeter Aspekt sind Belohnungen. Diese werden für bestimmte Aktionen und übergeordnete Errungenschaften vergeben, z.B. in Form von Abzeichen. Auch für das Programmierpraktikum lassen sich verschiedene Abzeichen integrieren: Anzahl erledigter Aufgaben, Erreichen der vollen Punktzahl, Anzahl fehlerfreier Compilerläufe, usw. Dies kann die Lernmotivation von Studierenden steigern; zum einen durch das positive Feedback zu erreichten Zielen, andererseits durch die Herausforderung („challenge“) und den Sammeltrieb, weitere Ziele freizuschalten. Zudem bieten Abzeichen einen Überblick darüber, was im Rahmen des Praktikums möglich ist. [8] Andererseits muss dabei auch beachtet werden, dass zu hohe, unerreichbar wirkende Ziele zu Demotivation führen können. Den NutzerInnen ist selbst überlassen, ob Sie alle Abzeichen sammeln möchten oder nicht. Besteht kein Interesse, entstehen daraus keine Nachteile.
Ein von den Studierenden im Praktikum explizit gewünschter Aspekt ist die Integration von mehr Hilfestellungen. Ein Konzept zur spielerischen Umsetzung sind *Hints* (Hinweise). Diese sind optionale, im Aufgabentext hinterlegte Informationen. Bei Schwierigkeiten beim Bearbeiten der Aufgabe können diese eingeblendet werden; dabei gibt es mehrere Stufen von verfügbaren Hinweisen (allgemein → konkret). So werden nicht von vornherein zu viele Informationen angegeben, bei Problemen stehen sie jedoch zur Verfügung. Dies soll die Frustration bei gedanklichen Hürden reduzieren. Hints sind jedoch nicht ohne Gegenleistung verfügbar. Zur Nutzung müssen zuvor einige der bereits für gute Bewertungen gesammelten Punkte gezahlt werden. Auf diese Weise sollen Studierende dazu motiviert werden, vor der Inanspruchnahme der Hinweise zuerst eigenständig andere Quellen auszuschöpfen (z.B. Online-Recherche). Durch dieses Vorgehen soll eine Förderung der Selbsthilfefähigkeit erzielt werden.

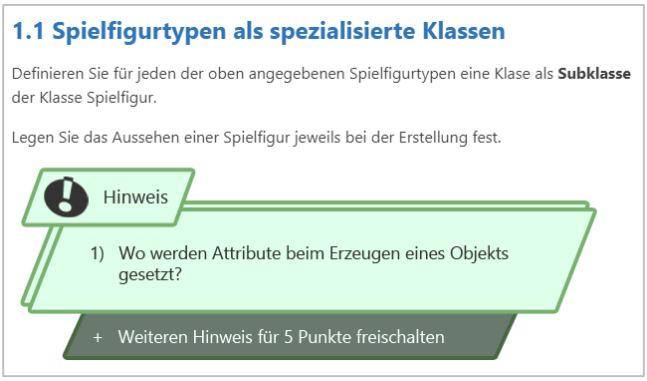

Abbildung 5: Hint

4.6 Fehlermeldungs-Quiz

Beim Ausführen und Kompilieren von Code werden Hinweise und Fehlermeldungen auf der Konsole ausgegeben. Jedoch wissen ProgrammieranfängerInnen oft nicht, wie sie diese interpretieren sollen und die Fehler-Ursache ermitteln können, da ihr mentales Modell dafür nicht ausreicht [29]. Aus diesem Grund sollen zusätzliche Hilfestellungen geboten werden, die zusätzliche Informationen zu den Fehlermeldungen darstellen.
Eine effektive Variante der spielerischen Wissensvermittlung ist das Quiz. Es ist eine einfache Art der Abfrage von Lerninhalten, die Lernende motivieren und direkte Rückmeldung zum eigenen Wissensstand geben kann [18]. Im Kontext der Konsole soll von daher ein Fehlermeldungs-Quiz integriert werden. Tritt ein Kompilier- oder Laufzeitfehler auf, wird dieser wie gewohnt ausgegeben. Zusätzlich wird ein Quiz zum aufgetretenen Fehler eingeblendet. Es wird eine Frage zum Verständnis des Fehlers gestellt, sowie mehrere Antwortmöglichkeiten. Bei Auswahl einer Lösung erfolgt ein direktes Feedback zur gegebenen Antwort. Für richtig beantwortete Fragen erhalten die Studierenden zusätzliche Punkte, die z.B. für Hints ausgegeben werden können. Die richtigen Lösungen sind, wenn man möchte, auch direkt als Hilfe zugänglich.

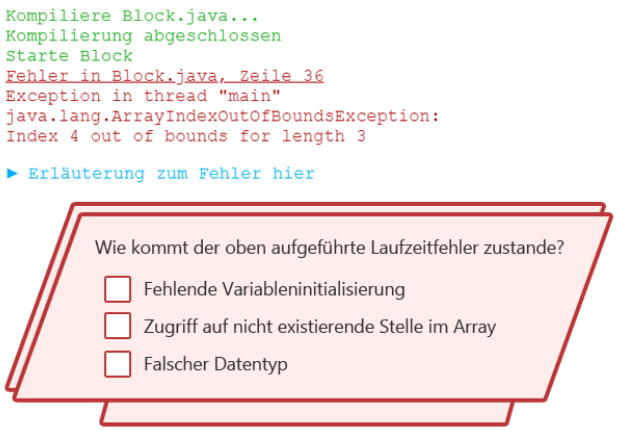

Abbildung 6: Fehlermeldungs-Quiz in Konsole

Mithilfe des Quiz soll die Fehlerlösungskompetenz der Studierenden gesteigert werden. Im Praktikum werden Fehlermeldungen oft übersehen oder nicht verstanden; das Quiz soll die NutzerInnen dazu trainieren, sich genauer mit den Meldungen zu beschäftigen. Andererseits unterbrechen die Quiz-Einheiten nicht den Fluss der Nutzung der IDE und können, wenn erwünscht, übersprungen werden.

4.7 Weitere Ideen

Zur Vereinfachung des Onboardings in die IDE für Erstbenutzer soll ein Kontext-Tutorial integriert werden, welches neue NutzerInnen über die wichtigsten Features informiert. Dieses kann bei Bedarf auch deaktiviert bzw. übersprungen werden, wenn keine Erläuterungen erwünscht sind. Auch die Ersteinrichtung und Individualisierung der Spielaspekte (z.B. Auswahl der Level Map) sollte beim ersten Start des Programms möglich sein. Damit die AnfängerInnen nicht zu Beginn zu viele Erläuterungen auf einmal verarbeiten zu müssen, werden neu freigeschaltete Features erst bei der Einführung erklärt.
Zusätzlich zu den Abzeichen können zu bestimmten Meilensteinen auch „Mikro-Zertifikate" vergeben werden. Diese haben keine Auswirkung auf die Bewertung der Aufgaben, dienen aber als zusätzliche Belohnung. Solche Zertifikate können beispielsweise für die Vervollständigung von Teilbäumen im Skill Tree ausgestellt werden.
Gibt es andererseits noch fehlende Einträge im Skill Tree, sollen diese auch durch Warnhinweise hervorgehoben werden. So wird deutlich, wenn eine Kompetenz nach einiger Zeit noch nicht beherrscht wird und ein Nachholbedarf besteht.
Neben vorgegebenen Zielen (Abzeichen, Skills, usw.) kann man Studierenden eventuell auch die Möglichkeit bieten, sich eigene Ziele zu setzen. Diese könnten unter Vorgaben (z.B. bezogen auf erreichte Punkte) selbstständig generiert und mit einer Auswahl an Belohnungen versehen werden. Auf diese Weise kann zusätzlich intrinsische Motivation generiert werden und Studierende erhalten die Möglichkeit, sich eigene Meilensteine zu setzen.

5 Fazit und Ausblick

Der sinnvolle Einsatz von Gamification kann die Motivation und Aufmerksamkeit von Nutzern erhöhen und erhalten. Die hier vorgestellten Elemente umfassen nicht nur Belohnungen, sondern Spiele-typische Features wie Level-Map, Skill Tree und Quiz. Die Motivation durch die hier vorgestellten Elemente ist intrinsisch: Die Spiel-Elemente ergänzen die Möglichkeiten der Darstellung und der Funktionalität, sie sind aber zum erfolgreichen Arbeiten nicht notwendig. Außerdem werden alle Erfolgsmeldungen und Resultate von Tests eigenverantwortlich von den Studierenden eingetragen werden und nicht zur Bewertung durch die Lehrenden verwendet werden.
Durch den Einsatz von Gamification wird von uns angestrebt, dass sich Studierende, geleitet von ihrer intrinsischen Motivation, verstärkt mit den Lerninhalten auseinandersetzen. Damit soll die Eigenreflektion der Studierenden unterstützt werden: Der eigene Fortschritt und der Lernbedarf sollen so besser erfasst und überblickt werden können. Dafür werden verschiedene Arten von Features, Funktionen und Visualisierungen angeboten, die je nach Präferenzen freiwillig in Anspruch genommen werden können, oder auch nicht.
Die Wirksamkeit der vorgestellten Konzepte soll im Rahmen eines Programmierpraktikums für AnfängerInnen getestet werden. Dabei wird die so erweiterte IDE 5Code im Rahmen des Wintersemesters 2020/21 eingesetzt, die Nutzung und die Lernerfolge werden evaluiert. Dabei soll auch erfasst werden, inwiefern die Spiel-Aspekte genutzt werden und als hilfreich empfunden werden, und welche weiteren Verbesserungs-möglichkeiten existieren.

LITERATUR

[1] Jürgen Börstler. 2007. Objektorientiertes Programmieren – machen wir irgendwas falsch? In: Sigrid Schubert (Hrsg.), Didaktik der Informatik in Theorie und Praxis, INFOS 2007, (LNI), P-112, Köllen Verlag, Bonn.

[2] James Stephen Williams. 2014. A Computer Learning Environment for Novice Programmers That Supports Cognitive Load Reducing Adaptations and Dynamic Visualisations of Computer Memory, Dissertation, Paper 574. Univ. of Wisconsin, Milwaukee.

[3] Kohl Bromwich, Masood Masoodian, Bill Rogers. 2012. Crossing the Game Threshold: A System for Teaching Basic Programming Constructs. CHINZ '12: Proceedings of the 13th International Conference of the NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction. Dunedin, New Zealand: ACM.

[4] Markus Dahm, Frano Barnjak, Moritz Heilemann. 2015. 5Code - Eine integrierte Entwicklungsumgebung für ProgrammieranfängerInnen. [Hrsg.] Hans Pongratz und Reinhard Keil. Lecture Notes in Informatics - DeLFI 2015 - Die 13. E-Learning Fachtagung Informatik. Bonn: Köllen Verlag.

[5] Jennifer Rose. 2019. Optimierung des Workflows der Programmierumgebung 5Code für ProgrammieranfängerInnen – Konzeption, Implementierung und Evaluation. Hochschule Düsseldorf: FB Medien.

[6] Michael Czarnetzki. 2020. Gamification in 5Code – Konzipierung und Implementierung von Game-Elementen in einer integrierten Programmierumgebung. Hochschule Düsseldorf: FB Medien.

[7] Edward L Deci. 1975. Intrinsic motivation. New York: Plenum Press.

[8] Kevin Werbach, Dan Hunter. 2012. For the Win - How Game Thinking Can Revolutionize Your Business. Philadelphia: Wharton Digital Press.

[9] Nike: Nike Run Club App für iPhone und Android. https://www.nike.com/de/de_de/c/nike-plus/running-app-gps

[10] Fitbit: Fitbit-App und Dashboard. https://www.fitbit.com/de/app

[11] Duolingo: Duolingo - Die weltbeste Methode, eine Sprache zu lernen. https://de.duolingo.com/

[12] Dropbox. https://www.dropbox.com/

[13] Dan Damsa. 2018. uxstudio: Gamification in UX – 3 Examples Of Using Gamification In UX. https://uxstudioteam.com/ux-blog/gamification-ux/

[14] Caitlin Kelleher. 2007. Storytelling Alice. http://www.alice.org/get-alice/storytelling-alice/

[15] RWTH Aachen. Codescape. https://codescape.medien.rwth-aachen.de/

[16] Cameron Chapman. Skillcrush: 15 Free Coding Games to Improve and Level Up Your Coding Skills. https://skillcrush.com/blog/free-coding-games/

[17] Interface in Game – Video games UI: Screenshots. 2020. https://interfaceingame.com/screenshots/?elements=skill-tree

[18] Bill Cushard. Mindflash: Three Benefits of Quizzes in e-Learning. https://mindflash.com/blog/three-benefits-of-quizzes-in-e-learning

[19] Headspace Meditation App. https://www.headspace.com/de/headspace-meditation-app

[20] JetBrains: JetBrains Academy. https://www.jetbrains.com/de-de/academy/

[21] JetBrains: EduTools. https://plugins.jetbrains.com/plugin/10081-edutools

[22] CodeCombat. https://codecombat.com/

[23] CodinGame. https://www.codingame.com/start

[24] Microsoft Store: Code Hunt Game. https://www.microsoft.com/de-de/p/code-hunt-game/9nblggh6d0gs

[25] Westermann Gruppe: Lernmethode Mindmap. https://c.wgr.de/f/verlage/westermanngruppe-at/dimensionen-mathematik/_dim7/materialien/00_Lernmethoden/08_mindmap.pdf

[26] Andrzej Marczewski. 2015. Even Ninja Monkeys Like to Play. CreateSpace Independent Publishing Platform.

[27] Richard A. Bartle. 1999. Players Who Suit MUDs http://mud.co.uk/richard/hcds.htm

[28] Institut für angewandte Menschenkunde: Systemisch orientierte Psychotherapie und Beratung. http://www.iam.or.at/material/systemischetherapie.pdf

[29] Andrew J. Ko, Brad A. Myers. 2003. Development and Evaluation of a Model of Programming Errors. Human-Computer Interaction Institute. Paper 184. http://repository.cmu.edu/hcii/184

[30] Electronic Arts GmbH, Jung von Matt AG, GEE Magazin (Redaktionswerft GmbH): Spielplatz Deutschland. 2006 http://www.geemedia.de/downloads/Spielplatz_Deutschland.pdf

[31] Growth Engineering: Does Gamification Inspire Intrinsic Motivation?. https://www.growthengineering.co.uk/intrinsic-motivation-gamification/

[32] Michael Sailer, Jan Ulrich Hense, Sarah Katharina Mayr, Heinz Mandl. 2016. How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. In: Computers in Human Behavior. Volume 69, April 2017, Pages 371-380. https://doi.org/10.1016/j.chb.2016.12.033

[33] GitHub: CodeSkills. http://stefansurkamp.github.io/cs/

[34] Apple: Swift Playgrounds. https://www.apple.com/de/swift/playgrounds/

[35] Mimo: Learn to Code. https://getmimo.com/

[36] Kai Huotari, Juho Hamari. 2012. Defining Gamification - A Service Marketing Perspective. ACM J. 10.1145/2393132.2393137.