

# Programming-Wiki: Online programmieren und kommentieren

Michael Hielscher  
Zentrum für Bildungsinformatik  
Pädagogische Hochschule Bern  
mail@michael-hielscher.de

Christian Wagenknecht  
Fachbereich Informatik  
Hochschule Zittau/Görlitz  
c.wagenknecht@hs-zigr.de

**Abstract:** In Zeiten von Web 2.0 finden immer mehr Aktivitäten orts- und zeitunabhängig über das Internet statt. Lerninhalte und Aufgabentexte in einem Wiki aufzubereiten, um sowohl Schülern und Schülerinnen als auch anderen Lehrpersonen die gemeinsame Nutzung zu ermöglichen, ist keine Seltenheit mehr. Hier wird beschrieben, wie Wikis durch Software-Erweiterungen auch als interaktive Lernumgebungen genutzt werden können. Dies wird am konkreten Beispiel „Einführung in die Programmierung“ gezeigt.

## 1 Einleitung

Wikis werden in erster Linie als Werkzeug für die computerunterstützte Mensch-Mensch-Kommunikation verwendet, um verschiedenste Materialien in einer Nutzergemeinde kollaborativ zu erarbeiten und zu verteilen. Unter dem Begriff Wiki verstehen wir hier die Verknüpfung aus einem technischen Wikisystem (Wiki-Engine) und den darin eingestellten Inhalten. Ein sehr populäres Wiki ist die Wikipedia. Sie basiert auf dem MediaWiki-System (vgl. [Med08]).

Neben dem stetig wachsenden Einsatz in der Industrie, sind Wikis auch in Hochschulen und Schulen gewinnbringend einsetzbar (vgl. [Döb07]). Lehrinhalte lassen sich in Wikis aufbereiten und je nach Wunsch des Autors gemeinsam mit anderen Lehrpersonen entwickeln. Ein gutes Beispiel ist das ZUM-Wiki (vgl. [ZUM08]) mit mehr als 1500 angemeldeten Benutzern, welches Informationen und Unterrichtsmaterialien zu praktisch allen Fächern der Sekundarstufe 1 und 2 zusammenträgt – eine Plattform von Lehrpersonen für Lehrpersonen.

Lerninhalte, die auf Wikis für den Unterricht bereitgestellt werden, lassen sich von Schülerinnen und Schülern nicht nur in der Schule, sondern auch vom heimischen Computer aus bearbeiten, kommentieren und ggf. modifizieren. Die Bedienanforderungen an Wiki-Autoren sind sehr gering, sodass auch Lernende Inhalte mit wenig Aufwand publizieren können. Das für traditionelle Webseiten typische Hochladen von Dateien auf den Webserver entfällt. Eine ausführliche Zusammenschau der Vorteile der Wiki-Technologie ist beispielsweise in [KH08] enthalten. Eine Aufgabenstellung wie „Verfasse eine Wikiseite zum Bubblesort-Algorithmus und präsentiere sie anschließend deinen Mitschülern.“ kann unter dem Aspekt *Lernen durch Lehren* vielversprechend sein.

Im Vergleich zum kooperativen Ansatz eines Wikis sind Learning Management Systems (LMS) eher zentralistisch organisiert. Sie beruhen auf der Vorstellung, dass Lernende vorbereitetes Lehrmaterial konsumieren, ohne es modifizieren (ergänzen, verändern, diskutieren) zu können. Ein LMS verfügt im Allg. über eine „eingebaute“ starre Didaktik, die nur mit größerer Mühe an die Wünsche guter Autoren angepasst werden kann. LMS sprechen daher eher Content-Provider statt Lehrende mit fach- und mediendidaktischer Qualifikation an. Um diesen didaktischen Defiziten entgegen zu wirken, werden Wikis in LMS integriert. Im Gegensatz dazu bevorzugen wir die Verwendung eines speziell präparierten Wikis als LMS-freies Basissystem mit unstrukturierter didaktischer Gestaltungsmöglichkeit für qualifizierte Lehrpersonen.

Neben der Mensch-Mensch-Kommunikation lässt sich der Computer auch für die Mensch-Maschine-Kommunikation nutzen. Wir sprechen von Lernsoftware, bei der Schülerinnen und Schüler in der Interaktion mit dem Computer etwas lernen sollen. Bei dieser Nutzung ist der Grad der Interaktivität zwischen dem Lernenden und der Lernumgebung entscheidend. Höhere Interaktion ist aufwändig in der Produktion und oft nur für gut formalisierbare Themen umsetzbar (vgl. [RH04]). Klassische LMS bieten für diesen Zweck interaktive Elemente, die jedoch nur tiefe kognitive Interaktionsstufen (vgl. [Sch02]) adressieren (Multiple-Choice-Fragen, Lückentexte usw.).

Zur Anreicherung eines Wikis mit interaktiven Elementen haben wir entsprechende Software-Erweiterungen in eine konventionelle Wiki-Engine eingebaut. Im Folg. zeigen wir am Beispiel der Einführung ins Programmieren, wie derart *angereicherte Wikis* als interaktives Lernmedium eingesetzt werden können. Dieser Ansatz lässt sich auch auf andere Themengebiete, etwa aus der Mathematik, Physik, Chemie oder Wirtschaft, übertragen.

Ursprünglich primär als kollaborative Arbeitsplattform gedacht, können sich Wikisysteme mit diesen Erweiterungen noch mehr in Richtung *Autorenwerkzeug für interaktive Lernumgebungen* entwickeln.

## 2 Programming-Wiki

Zentrale Inhalte des Programmierunterrichts sind das Lesen und Schreiben von Code sowie das Testen entwickelter Programmen. Nicht in jedem Fall wird ein Programm auf einem leeren Blatt entworfen. Lehrpersonen können Programmgerüste oder geeignete Bausteine vorgeben, um den Entwurfsprozess anzuleiten. Die für den Lehrenden durchaus mühevoll verteilte solchen Materials, sei es auf Papier (Arbeitsblatt) oder in digitaler Form (Webseite, LMS, etc.), kann eine echte Unterstützung des Lernenden nicht garantieren: Oftmals ist der Zusammenhang zwischen Aufgabenstellung, Arbeitsanleitung, Lehrtext und eben diesen Bausteinen für den Lernenden nicht transparent oder geht im Bearbeitungsprozess unter. Außerdem muss Code in die verwendete Entwicklungsumgebung übertragen werden, was einen Medienbruch bedeutet.

Die Installation und Konfiguration einer Programmierumgebung auf dem eigenen Computer ist für Schülerinnen und Schüler aufwändig und erfordert häufig Fachwissen. Stimmen die lokalen Einstellungen nicht mit denen in der Schule überein, ergeben sich bei der Ausführung von Programmen, die im Unterricht erarbeitet wurden, schnell schwer zu

interpretierende Fehlermeldungen.

Im Trend von Web 2.0 entstehen immer mehr Online-Entwicklungsumgebungen (z.B.: [Her08]). Der Entwurf solcher Umgebungen steht aber erst am Anfang und sie sind bislang nicht für den Schuleinsatz konzipiert. Man darf nicht übersehen, dass komplexere Arbeitsumgebungen einen nicht zu unterschätzenden Einarbeitungsaufwand für die Nutzer erfordern. Gerade in der Anfangsphase des Programmierunterrichts stehen kurze Programme im Zentrum. Dafür würde eine einfache Entwicklungsumgebung ausreichen. Die Nutzung eines angereicherten Wikis als *Programming-Wiki* bietet sich an.

Mittels einer Software-Erweiterung (Extension) wird das bekannte MediaWiki-System zur Integration von interaktiven Elementen befähigt. Dadurch kann ein eingebetteter Programmtext im Wiki nicht nur gelesen, sondern aktiv verändert, gespeichert und ausgeführt werden. Wir bezeichnen diese Wikiseiten deshalb als *interaktive Arbeitsblätter*. Ein Interpreter der gewählten Programmiersprachen (derzeit Java, Pascal, Scheme, JavaScript, XSLT und XPath) arbeitet im Hintergrund, um Programmtexte zu evaluieren und die Ergebnisse der Abarbeitung wieder im Arbeitsblatt zu präsentieren. All dies geschieht unmittelbar und innerhalb des Webbrowsers mit Hilfe kleiner Java-Applets, die die nötigen Interpreter ohne Installation auf dem Clientrechner realisieren. Voraussetzung ist ausschließlich ein Java-fähiger Webbrowser. Derzeit werden Firefox und Internet Explorer unterstützt. Es braucht keine weitere Software installiert zu werden.

### Aufgabe: Glücksspiel

[\[Bearbeiten\]](#)

Bei einem Glücksspiel geht es darum, mit 3 Würfeln möglichst viele gleiche Augen zu werfen. Entwickle ein Programm, welches dreimal würfelt und die Augenzahl ausgibt (zum Beispiel 6,3,4). Anschließend soll ausgegeben werden, ob der Spieler verloren oder gewonnen hat.



Gewinnbedingung: Mindestens 2 Würfel haben die gleiche Augenzahl.

```
x 1 int w1, w2, w3;
2 w1 = (int) (Math.random() * 6) + 1;
3 w2 = (int) (Math.random() * 6) + 1;
4 w3 = (int) (Math.random() * 6) + 1;
5 System.out.println("Du hast " + w1 + ', ' + w2 + ', ' + w3 + " gewürfelt.");
6 if (w1 == w2 )
7     System.out.println("Du hast gewonnen!"); else
8 if (w1 == w3 )
9     System.out.println("Du hast gewonnen!"); else
10 if (w2 == w3 )
11     System.out.println("Du hast gewonnen!"); else
12     System.out.println("Du hast verloren!");
13
```

#### speichern & ausführen

```
> Du hast 3, 6, 3 gewürfelt.
Du hast gewonnen!
```

Abbildung 1: Beispiel einer Programmieraufgabe (Schülerlösung)

Die Idee des Programming-Wiki ist im Kern nicht neu: Bereits 1984 beschrieb Donald Knuth seinen *Literate-programming*-Ansatz (vgl. [Knu84]), bei dem es darum geht, sowohl die Dokumentation als auch den Quelltext eines Programms in einer gemeinsamen Datei abzulegen. In unserem Fall geht es analog um die Verknüpfung von Lehrtexten und Quelltexten innerhalb einer Webseite. Seit mehreren Jahren wird dieser Ansatz an der Hochschule Zittau/Görlitz im Kurs Programmierparadigmen praktiziert. 2002 wurde zu diesem Zweck SchemeNet (vgl. [Wag05], [WB05]) entwickelt, welches es erlaubt, Programmtexte der Programmiersprache Scheme innerhalb einer Webseite einzubinden, zu editieren und direkt auszuführen. SchemeNet-Seiten werden vom Lehrenden in einer eigenen XML Sprache verfasst und in HTML transformiert. Für die Ausführung von Scheme-Programmen innerhalb einer solchen Webseite wird ein „Scheme-Server“ verwendet, der für jeden Nutzer separat die Rolle des Scheme-Interpreters übernahm. 2007 wurde der Scheme-Server durch einen Client-seitigen Scheme-Interpreter ersetzt, um Sicherheits- und Performanceprobleme zu lösen. Das System wurde über mehrere Jahre erfolgreich mit großem Interesse der Studierenden erprobt und eingesetzt. Die Herstellung von derartigen interaktiven Arbeitsblättern blieb einer kleinen Autorengemeinde mit der Bereitschaft, eine speziellen XML Sprache zu erlernen, vorbehalten. Die generierten HTML-Dateien mussten manuell auf einen Webserver hochgeladen werden, um sie den Lernenden zur Verfügung zu stellen. Das in diesem Beitrag vorgestellte Programming-Wiki stellt eine Weiterentwicklung des beschriebenen Ansatzes dar. Durch die konsequente Nutzung der Wiki-Technologie wird, neben vielen weiteren Vorteilen, die Handhabung für Autoren maßgeblich verbessert.

### 3 Programming-Wiki aus der Sicht des Lernenden

Zur Illustration der Lernenden-Perspektive auf ein Programming-Wiki verwenden wir in den folgenden Beispielen und Abbildungen jeweils nur Auszüge aus interaktiven Arbeitsblättern. Es wird speziell auf die interaktiven Elemente innerhalb umfangreicher Lehrtexte eingegangen. Dabei wurden bewusst klassische und kurze Beispiele aus der Mathematik und Geometrie gewählt, um den Blick auf die Realisierung im Programming-Wiki zu lenken. Im Anhang finden sich kreative Beispiele zur Ton- und Grafikerzeugung. Als Sprache wurde exemplarisch Java gewählt.

In Abb. 1 ist eine Übungsaufgabe zur Erzeugung von Zufallszahlen und Textausgabe dargestellt. Die Schülerin hat bereits eine Lösung im Eingabebereich, der *Codebox*, erarbeitet und diese über die Schaltfläche abgespeichert und ausgeführt. Das Ergebnis wird im nachfolgenden Ausgabefeld präsentiert.

In vielen Fällen werden vom Lehrer bereits Programmteile vorgegeben, die vom Lernenden ergänzt werden sollen. Dies können etwa Methodenrumpfe als Schnittstellenbeschreibungen, Beispielaufrufe oder vollständige Programme zum Experimentieren sein.

Abb. 2 zeigt eine Aufgabe zur Entwicklung des Bubble-Sort-Algorithmus. Die von der Lehrperson gelieferte Vorgabe kann bereits vom Schüler ausgeführt und die Ausgabe betrachtet werden. Da die Methode `BubbleSort` keine Anweisung enthält, wird die vordefinierte Liste unverändert ausgegeben.

## Aufgabe: Bubble-Sort

[Bearbeiten]

In dieser Aufgabe sollst du den BubbleSort Algorithmus implementieren. Im nachfolgenden Eingabefeld ist bereits eine passende Methode für dich vorgegeben. Eine Liste aus Ganzzahlen wird als Parameter übergeben. Die Methode besitzt keinen Rückgabewert, da die Liste als Referenz übergeben und verändert wird.

```
x 1 void BubbleSort (int[] liste) {  
2     // Implementiere den BubbleSort Algorithmus  
3 }  
4
```

```
x 1 int[] testliste = new int[] {1,2,9,5,2,23,1};  
2 BubbleSort(testliste);  
3  
4 for (int i=0; i < testliste.length; i++)  
5     System.out.print(testliste[i]+" ");  
6
```

**speichern & ausführen**

```
> 1, 2, 9, 5, 2, 23, 1,
```

Abbildung 2: Vorgabe für eine Aufgabe zu Bubble-Sort

Je nach Aufgabenart können Arbeitsblätter auch *Feedback-Elemente* enthalten. Vom Lehrer definierte Testfälle werden automatisch überprüft. Abb. 3 illustriert dies am Beispiel des Bubble-Sort-Algorithmus'. Es geht nicht um eine echte Lösungskontrolle, sondern um eine einfache, qualitative Erfolgsbeurteilung für die Lernenden.

Die ggf. modifizierten Programmtexte in den Codeboxen und die Ergebnisse von Feedback-Elementen werden für jeden angemeldeten Benutzer separat in der Wiki-Datenbank gespeichert. Wechselt eine Schülerin ihren Arbeitsplatz im Computerpool, oder möchte von zu Hause ihre Lösung komplettieren, findet sie im Wiki stets ihren aktuellen Arbeitsstand vor. Ein Datentransfer von einem Rechner zum nächsten entfällt. Da keine lokalen Daten auf dem Arbeitsrechner abgelegt werden, können diese auch nicht verloren gehen, etwa durch Speichern auf einem falschem Datenträger oder unter einem falschem Dateinamen.

Die Navigation innerhalb von Lehrtexten und Aufgabenblättern wird durch das Wiki bereitgestellt. Eine Volltextsuche wird ebenso angeboten wie eine Versionsgeschichte und eine ausgereifte Benutzerverwaltung, über die sich jeder Schüler zu Kursbeginn anmelden kann.

## 4 Programming-Wiki aus der Sicht des Lehrenden

Das Konzept und die Handhabung zum Anlegen und Editieren von Wiki-Seiten wird unverändert aus dem MediaWiki-System übernommen und ist den meisten Lehrenden durch die Wikipedia vertraut. Der Autor von Wiki-Inhalten kann neben Text, Tabellen, Grafiken, Audiodateien (MP3), Videos, ZIP-Archiven oder PDF-Dokumenten, beliebig viele Codeboxen und Feedback-Elemente in eine Wikiseite einfügen. In einer frei editierbaren Code-

```
x|1 void BubbleSort (int[] liste) {
2   for(int i = liste.length-1; i > 0; i--)
3     for(int j = 0; j < i; j++)
4       if(liste[j] > liste[j+1] ) // tauschen
5         {
6           int temp = liste[j+1]; // zwischenspeichern
7           liste[j+1] = liste[j];
8           liste[j] = temp;
9         }
10  }
11
```

. . . (wie zuvor) . . .

> 1,1,2,2,5,9,23,


 **Prima, deine Lösung scheint zu stimmen.**  
jetzt prüfen & speichern

Abbildung 3: Schülerlösung der Übungsaufgabe zu Bubble-Sort


box können fertige Quelltexte zum Experimentieren, Teillösungen zur Vervollständigung oder ausschließlich Hinweise zur Entwicklung einer eigenen Lösung als Vorgabe hinterlegt werden. Für ein Feedback-Element wird hingegen ein Testausdruck angegeben, welcher auf die Lösung des Schülers bei Betätigen der Prüfschaltfläche angewendet wird.

Um ein interaktives Element einzufügen, wird in den Quelltext des Arbeitsblatts ein XML-Tag wie `<eval>...</eval>` eingefügt. Das Wiki erstellt für den Betrachter der Seite eine geeignete Darstellung etwa in Form eines Editors inklusive Syntax Highlighting (vgl. Abb. 4).

Das Programming-Wiki unterstützt neben der reinen Textausgabe auch graphische und akustische Ausgaben. In Abb. 4 wird ein Turtle für das Zeichnen eines Kreises eingesetzt. Als Ausgabe wird ein *Canvas-Element* verwendet, welches vom Autor mit dem Tag `<canvas>` unterhalb eines `<eval>` Elements platziert wurde. In diesem Beispiel wird eine Lösung bereits vorgegeben, mit der die Schüler und Schülerinnen experimentieren können.

Ein Feedback-Element kann über das Tag `<check>` eingebunden werden. In Abb. 5 wird dies am Beispiel der Fibonacci-Funktion gezeigt. Die Methode `fibonacci` wird hierfür mit mehreren Eingabewerten aufgerufen und mit Referenzwerten verglichen.

Da alle Schülerlösungen in der Wiki-Datenbank abgelegt werden, können diese vom Lehrenden ausgewertet werden. Meldet sich eine Lehrperson am System an, werden alle Benutzernamen aufgelistet, die bereits Lösungen für das aktuelle Arbeitsblatt erarbeitet haben. Verwendet der Schüler die enthaltenen Feedback-Elementen zur Beurteilung seiner Lösungen, so wird angezeigt, ob er erfolgreich war oder nicht (siehe Abb. 6). Durch einen Klick auf einen Namen werden die Lösungen dieses Nutzers geladen und können vom Lehrenden begutachtet und ggf. auch kommentiert oder korrigiert werden. Betrachtet der



Programming: **Java** | Eval | Code | Hidden | Check | Canvas

Experimentieren Sie mit nachfolgendem Beispiel für Turtle Grafiken mit Java:


```

<eval>
canvas.clear(); // Zeichenfläche löschen
turtle.home();

boolean isUp = false;

for(int i = 0; i < 18; i++){
  turtle.penColor(i*10+50,0,250-10*i);
  turtle.penWidth(8);
  turtle.forward(20);
  turtle.right(20);
  isUp = !isUp;
  if(isUp) turtle.penUp(); else turtle.p
}
</eval>

<canvas></canvas>
  
```



Experimentieren Sie mit nachfolgendem Beispiel für Turtle Grafiken mit Java:

```

1 canvas.clear(); // Zeichenfläche löschen
2 turtle.home();
3
4 boolean isUp = false;
5
6 for(int i = 0; i < 18; i++){
7   turtle.penColor(i*10+50,0,250-10*i);
8   turtle.penWidth(8);
9   turtle.forward(20);
10  turtle.right(20);
11  isUp = !isUp;
12  if(isUp) turtle.penUp(); else turtle.penDown();
13 }
14
  
```

speichern & ausführen

>





Abbildung 4: Eine Codebox in ein Arbeitsblatt einbinden



Java | Eval | Code | Check

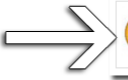
==Aufgabe 4==

Entwickeln Sie eine Funktion zur Bestimmung der n. Fibonaccizahl.

```

<code>
int fibonacci (int n){
  // berechne hier die n. Fibonacci-Zahl
  return 0;
}
</code>

<check>
return fibonacci(11)==89 &&
  fibonacci(12)==144 &&
  fibonacci(13)==233 &&
  fibonacci(14)==377;
</check>
  
```



**Aufgabe 4** [Bearbeiten]

Entwickeln Sie eine Funktion zur Bestimmung der n. Fibonaccizahl.

```

1 int fibonacci (int n){
2   // berechne hier die n. Fibonacci-Zahl
3   return 0;
4 }
5
  
```



**Quelltext überprüfen:**
jetzt prüfen

Abbildung 5: Hinzufügen eines Feedback-Elements

Schüler anschließend erneut sein Arbeitsblatt, kann er die Anmerkungen des Lehrenden einsehen und entsprechend darauf reagieren.

The screenshot shows the 'Übungsaufgabe 1 mit Code' page in the Programming Wiki. At the top, there are navigation links: Admin, Eigene Diskussion, Einstellungen, Beobachtungsliste, Eigene Beiträge, and Abmelden. The main title is 'PROGRAMMING WIKI' with a pencil icon below it. The page has tabs for 'Seite', 'Diskussion', 'Bearbeiten', 'Versionen/Autoren', 'Löschen', 'Verschieben', and 'Schützen'. The 'Benutzer:' section shows 'Seminargruppe 1' with user avatars for MaxMustermann and EvaMustermann. Below this, it says 'Lösungen von: MaxMustermann'. There is an 'Inhaltsverzeichnis [Verbergen]' section with a list of tasks: 1 Aufgabe 1, 2 Aufgabe 2, 3 Aufgabe 3, 4 Aufgabe 4, and 5 Aufgabe 5. The 'Aufgabe 1' section is highlighted, with a '[Bearbeiten]' link. To the right of the task list is a 'Java' label and a globe with a sign that says 'Hallo'. On the left side, there are navigation and search boxes, and a 'Werkzeuge' section with links like 'Links auf diese Seite', 'Änderungen an verlinkten Seiten', 'Hochladen', 'Spezialseiten', 'Druckversion', and 'Permanentlink'.

Abbildung 6: Benutzerauswahl für den Administrator

## 5 Zusammenfassung und Ausblick

Am konkreten Beispiel der Vermittlung von Programmiergrundlagen haben wir gezeigt, wie interaktive Elemente in ein Wiki-System integriert werden können. Damit lassen sich die Vorteile eines Wikis mit denen von spezialisierten Lernumgebungen verknüpfen und Synergieeffekte nutzen. Der Erfolg von Wikis ist nicht zuletzt ihrer einfachen und leicht zu erlernenden Bedienung zuzuschreiben. Wikis, wie das verwendete MediaWiki, haben eine breite Akzeptanz und sind kostenfrei erhältlich.

Das vorgestellte Konzept lässt sich auch auf andere Fachbereiche übertragen. In weiteren Projekten sollen interaktive Elemente zur Mathematik, Chemie aber auch zum Fremdsprachenunterricht entwickelt werden. Unser Interesse gilt hier vor allem Inhalten, die gut formalisierbar sind.

Unter der Adresse: <http://www.michael-hielscher.de/wiki> steht das hier beschriebene Programming-Wiki allen Interessenten zum Experimentieren offen. An der Implementierung der Programming-Wiki-Erweiterungen waren die Studierenden Anna Prenzel, Philipp Herzig, Filip Martinovský dankenswerterweise beteiligt.

## Anhang: Weitere Beispiele aus Programming-Wiki

Im Folg. illustrieren wir einige ausgewählte Programming-Wiki-Beispiele, die im Programmierunterricht von Schülerinnen und Schülern vollständig oder teilweise bearbeitet wurden. In den entsprechenden Codeboxen (s. Screenshots) sind jeweils zugehörige Lösungen angegeben.

Abb. 7 zeigt ein Beispiel zu John Conway's Game of Life, bei dem Arrays für das Spielfeld zum Einsatz kommen. Die Abarbeitung des von Conway angegebene Algorithmus' führt zu einer graphischen Darstellung, die im Programming-Wiki nahtlos aufgenommen wird.

```
26 if(x>0 && y<h-1 && Spielfeld[x-1][y+1] == 1) r++;
27 if(x<w-1 && y>0 && Spielfeld[x+1][y-1] == 1) r++;
28 if(x<w-1 && y<h-1 && Spielfeld[x+1][y-1] == 1) r++;
29 return r;
30 }
31 void flipCells(){
32 int[][] SpielfeldNeu = new int[w][h];
33 for(int x=0; x<w; x++){
34 for(int y=0; y<h; y++){
35 int neighbors = countNeighbors(x,y);
36 SpielfeldNeu[x][y] = Spielfeld[x][y];
37 if(Spielfeld[x][y] == 1){
38 if (neighbors < 2 || neighbors > 3) SpielfeldNeu[x][y] = 0;
39 }else{
40 if (neighbors == 3) SpielfeldNeu[x][y] = 1;
41 }
42 }
43 Spielfeld = SpielfeldNeu;
44 }
45
46
47 canvas.clear();
48 while(true) { drawCells(); flipCells(); Thread.sleep(100); }
49
```

Ausführung abbrechen

ausführen

> User break.

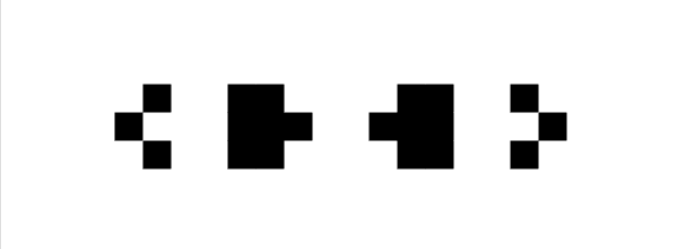


Abbildung 7: Beispiel zu John Conway's Game of Life

Abb. 8 enthält ein Einstiegsbeispiel zur imperativen Programmierung. Die Schülerinnen und Schüler experimentieren mit vordefinierten Prozeduren (Befehlen), um eine Spielfigur durch ein Labyrinth zu navigieren. Befehlsfolgen können zusammengefasst und unter einem neuen Namen gespeichert werden. Auf diese Weise konstruieren die Lernenden neue Prozeduren, wie etwa in folgendem Beispiel:

```
void laufe3nachoben() {  
    nachoben();  
    nachoben();  
    nachoben();  
}
```

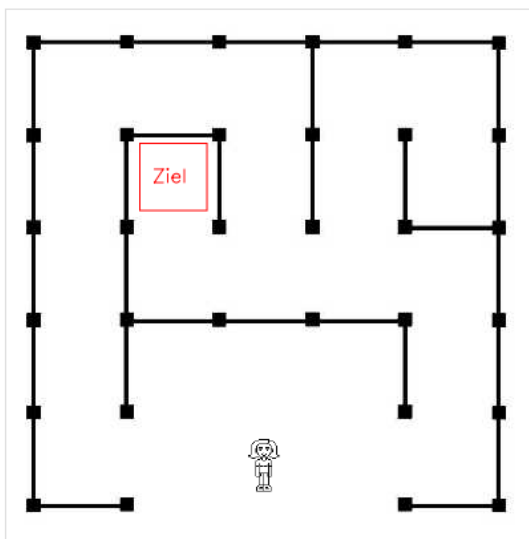
## Der Weg aus dem Labyrinth

[Bearbeiten]

Anna ist in einem Labyrinth gefangen und du musst ihr helfen. Anna kann über 4 **Methoden** gesteuert werden. Wende diese *nacheinander* an, um Anna bis zum rot markierten Ziel zu lenken.

Die Methode **start()** stellt Anna wieder an den Anfang des Labyrinths. Wir rufen diese Methode deshalb zu Beginn von jedem Programmstart auf.

Anna versteht die folgenden 4 Anweisungen: **nachlinks(); nachoben(); nachrechts(); nachunten();**



```
1 start();  
2 nachrechts(); nachrechts();  
3 nachoben(); nachoben();  
4 nachlinks(); nachlinks(); nachlinks();  
5 nachoben();  
6
```

ausführen

>

Abbildung 8: Beispiel zu Grundlagen imperativer Programmierung

Abb. 9 zeigt eine Übungsaufgabe, bei der mit graphischen Hilfsmitteln ein Codestreifen für einfache Musikstücke erzeugt werden soll. Um diese Aufgabe lösen zu können, haben die Schülerinnen und Schüler in vorangegangenen Übungen Methoden zur Ton- und Grafikerzeugung kennen gelernt und angewandt. Ziel dieser Übung ist es, die Schleife als Kontrollstruktur zu thematisieren.

### Musik veranschaulichen

[Bearbeiten] Java

Du kennst die Tonleiter und die Notenschrift aus dem Musikunterricht. Für Drehorgeln und Spieluhren wird hingegen eine Art Code in Form von Stifen verwendet, die auf einer Walze aufgebracht sind. Im Bild rechts sieht man das Innenleben einer Spieluhr.

In dieser Aufgabe sollst du versuchen, aus einem Lied als Zeichenkette wie:

```
String Lied = "E-4 E-4 E-2 E-4 E-4 E-2 E-4 G-4 C-4 "+
              "D-4 E-1 F-4 F-4 F-4 F-4 F-4 E-4 E-4 "+
              "E-8 E-8 E-4 D-4 D-4 E-4 D-2 G-2";
```

eine Darstellung für eine zugehörige Walze einer Spieluhr zu erzeugen. Gleichzeitig kannst du das Lied auch mit Hilfe der Methoden, die wir letzte Woche kennengelernt haben, vorspielen lassen.



Spieluhr und Walze mit Stifen



```
1 canvas.clear();
2 String Lied = "E-4 E-4 E-2 E-4 E-4 E-2 E-4 G-4 C-4 "+
3              "D-4 E-1 F-4 F-4 F-4 F-4 F-4 E-4 E-4 "+
4              "E-8 E-8 E-4 D-4 D-4 E-4 D-2 G-2";
5
6 Orgel = new Sound(19); // 19. MIDI-Gerät = Orgel
7
8 String[] Noten = Lied.split(" ");
9
10 int posx = 0;
11 int posy = 0;
12 for(int i=0; i < Noten.length; i++){
13     int key = 0;
14     if (Noten[i].charAt(0) == 'C') key = 60;
15     if (Noten[i].charAt(0) == 'D') key = 62;
16     if (Noten[i].charAt(0) == 'E') key = 64;
17     if (Noten[i].charAt(0) == 'F') key = 65;
18     if (Noten[i].charAt(0) == 'G') key = 67;
19     if (Noten[i].charAt(0) == 'A') key = 69;
20     if (Noten[i].charAt(0) == 'H') key = 71;
21     int dauer = 0;
22     if (Noten[i].charAt(2) == '1') dauer = 1000/1;
23     if (Noten[i].charAt(2) == '2') dauer = 1000/2;
24     if (Noten[i].charAt(2) == '4') dauer = 1000/4;
25     if (Noten[i].charAt(2) == '8') dauer = 1000/8;
26
27     orgel.play(key, dauer);
28     canvas.fillRect(posx, 80-(key-60)*8, dauer/20, 10);
29     posx = posx + dauer/20+5;
30 }
31
```

speichern & ausführen

Abbildung 9: Beispiel zur Grafik- und Tonerzeugung

## Literatur

- [Döb07] Beat Döbeli. Wiki und die starken Potenziale - Unterrichten mit Wikis als virtuellen Wandtafeln. *Zeitschrift Computer und Unterricht, Web 2.0 und Schule* Nr 66:39–41, 2007.
- [Her08] Heroku, Oktober 2008.  
<http://heroku.com/>.
- [KH08] Christian Kohls und Simone Haug. Gemeinsam sind wir stark! - Kooperativer Wissenserwerb mit Wikis. *LOG IN*, 152:22–28, 2008.
- [Knu84] Donald E. Knuth. Literate Programming. *The Computer Journal*, vol.27:97–111, 1984.
- [Med08] MediaWiki, Oktober 2008.  
<http://www.mediawiki.org>.
- [RH04] Raimond Reichert und Werner Hartmann. On the Learning in E-Learning. *Proceedings of EDMEDIA 2004, World Conference on Education Multimedia, Hypermedia and Telecommunications* June 23-26, Lugano, Swizerland:1590–1595, 2004.
- [Sch02] Rolf Schulmeister. Taxonomie der Interaktivität von Multimedia - Ein Beitrag zur aktuellen Metadaten-Diskussion. *it+ti*, Ausgabe 4:193–199, 2002.
- [Wag05] Christian Wagenknecht. Mediendidaktische Begleitung im Informatikunterricht mit SchRepo, SchemeNet und SchemeGrader. *Unterrichtskonzepte für informatische Bildung*, INFOS'05 - Praxisband:21–24, 2005.
- [WB05] Christian Wagenknecht und Veit Berger. Programmierparadigmen mit Scheme. *Unterrichtskonzepte für informatische Bildung*, INFOS'05:219–229, 2005.
- [ZUM08] ZUM Wiki, Oktober 2008.  
<http://wiki.zum.de/>.