Travail de candidature

Erfolgreiches Einführen der Programmierungsgrundlagen durch konsequent problemorientierte Unterrichtsmethodik

Das Fach Informatik im Sinne von PISA 2015

Danksagung

Ich bedanke mich bei Herrn Ingo Schandeler für seine Ideen, seine Beratung und seine Unterstützung bei der Realisierung dieser Arbeit.

Ich danke den Herren Robert Fisch, Laurent Haan, Serge Dondelinger, Serge Linckels und Alain Delosch für ihre Unterstützung bei der Durchführung der Schülerbefragungen in ihren Klassen, sowie den Austausch über den Kursverlauf.

Ich danke Herrn Marc Schmit für seine kontinuierliche Unterstützung während meiner gesamten bisherigen Laufbahn als Lehrer.

Last, but not least, bedanke ich mich bei meinen Eltern für ihre lebenslange Unterstützung und insbesondere bei meiner Mutter für die Korrektur dieser Arbeit.

Erklärung

Hiermit erkläre ich diese Arbeit selbst und aus eigenen Mitteln erstellt zu haben.

Gilles Everling

Gilles Everling

candidat professeur ingénieur

Erfolgreiches Einführen der Programmierungsgrundlagen durch konsequent problemorientierte Unterrichtsmethodik

Das Fach Informatik im Sinne von PISA 2015

Lycée technique des Arts et Métiers

2012

Zusammenfassung

In der vorliegenden Arbeit untersuche ich die Vor- und Nachteile einer konsequent problemorientierten Unterrichtsmethodik bei der Einführung der Programmierungsgrundlagen im Informatikunterricht.

Nach einer Einleitung zum Thema und dem Zusammenhang mit den PISA-Studien untersuche ich die theoretischen Grundlagen von Problem-Based Learning (PBL). PBL passt hervorragend in die konstruktivistische Sichtweise und harmoniert mit den aktuellen Erkenntnissen der Neurodidaktik. Ich untersuche die Voraussetzungen für erfolgreiches Problemlösen, um dann auf die Ursprünge und die Definition von PBL einzugehen. Ich untersuche die wichtigsten Aspekte der Problemerstellung und beschreibe den Lernablauf in drei verschiedenen Ausprägungen, die Grundcharakteristiken sowie die pädagogischen Ziele von PBL. Anschließend gehe ich auf verschiedene Möglichkeiten der PBL-gerechten Evaluation ein und fasse die wichtigsten Forschungsergebnisse aus Metaanalysen zur Wirksamkeit von PBL fächerübergreifend, sowie speziell im Informatikbereich, zusammen.

Ich illustriere die konkrete Umsetzung der im theoretischen Teil behandelten Grundlagen anhand von fünf Erfahrungsberichten zum Einsatz von PBL im Informatikunterricht auf Universitätsniveau.

Ich erläutere meine Erfahrungen mit einem von Grund auf neu entwickelten und auf PBL ausgerichteten Kurs zum Unterrichten der Programmierungsgrundlagen im technischen Sekundarunterricht und komme zum Schluss, dass PBL sich positiv auf Motivation und Kompetenzniveau auswirkt.

Im Anschluss untersuche ich, aufgrund von Schüler- und Lehrerfeedback sowie der Leistungsergebnisse, die Vor- und Nachteile von PBL gegenüber dem traditionellem Unterricht. Statistisch signifikante Rückschlüsse lassen sich nicht ziehen, tendenziell bestätigen jedoch vor allem die Leistungsergebnisse die Einschätzung, dass PBL nach dem Überwinden der Startschwierigkeiten zu einer starken Kompetenzentwicklung und hoher Motivation führt.

Um in Zukunft einen reibungsloseren Einstieg und noch bessere Ergebnisse mit PBL zu erzielen, gebe ich detaillierte Empfehlungen zu allen Aspekten des Unterrichts.

Rückblickend auf die gemachten Erfahrungen überwiegen die Vorteile von PBL und legen einen breiteren Einsatz, innerhalb und außerhalb der Informatik, nahe.

Inhaltsverzeichnis

1 Einleitung

1.1 *Begründung meiner Wahl*

„*Above all, information and knowledge are growing at a far more rapid rate than ever before in the history of humankind. As Nobel laureate Herbert Simon wisely stated, the meaning of 'knowing' has shifted from being able to remember and repeat information to being able to find and use it (Simon, 1996). More than ever, the sheer magnitude of human knowledge renders its coverage by education an impossibility; rather, the goal of education is better conceived as helping students develop the intellectual tools and learning strategies needed to acquire the knowledge that allows people to think productively about history, science and technology, social phenomena, mathematics, and the arts. Fundamental understanding about subjects, including how to frame and ask meaningful questions about various subject areas, contributes to individuals' more basic understanding of principles of learning that can assist them in becoming self-sustaining, lifelong learners.*“ (cf. [1] Bransford, Brown & Cocking S. 5)

Im Rahmen meines Mémoire professionnel, mit dem Titel „Klassenführung im Informatikunterricht“, ist mir bewusst geworden, dass die Motivierung und Aktivierung der Schüler eine entscheidende Rolle für die Qualität des Unterrichts und der Klassenführung spielt (cf. [2] Everling S. 9).

Dies ist auch ein wichtiger Punkt im Rahmen der Reform der Berufsausbildung: „*Lernsituationen sind herausfordernde Frage-, Aufgaben oder Problemstellungen, welche einen konkreten Bezug zu den zu erwerbenden Kompetenzen des Moduls haben.*“ (cf. [3] MEN S. 2).

Im bisherigen Informatikunterricht werden die Lerninhalte, nach meiner Erfahrung, vorrangig mit einer Vielzahl von relativ kleinen, unabhängigen Aufgaben geübt. Hierbei habe ich des Öfteren festgestellt, dass sich die Motivation der Schüler zur Lösung der Aufgaben in Grenzen hält. Bei interessanteren und anspruchsvolleren Aufgaben, etwa der programmatischen Steuerung eines Lego-Roboters, habe ich hingegen eine starke Steigerung der Motivation zur Auseinandersetzung mit neuen Entwicklungsumgebungen und sogar unbekannten Programmiersprachen seitens der Schüler beobachtet (cf. [4] Everling).

Dies deckt sich mit meiner langjährigen Berufserfahrung, wo ich erlebt habe, dass eine anspruchsvolle Problemstellung eine sehr motivierende Wirkung hat. Das Lernen findet sozusagen als Neben-

effekt der Auseinandersetzung mit dem Problem statt. Dadurch ist man gezwungen, sich aktiv auf die Suche nach Lösungen zu begeben, anstatt passiv darauf zu warten, dass sie einem vorgetragen werden. Die Rolle der Lehrkraft besteht dabei verstärkt in der Unterstützung des aktiven Lernprozesses der Schüler.

1.2 *PISA 2012*

Das OECD Programme for International Student Assessment (PISA) untersucht seit 1997 im Dreijahresrhythmus, mittels Erhebungen, die Kompetenzen von 15-jährigen Schülerinnen und Schülern in den Bereichen Lesen, Mathematik und Naturwissenschaften.

Für die PISA 2003 Studie wurde erstmals ein zusätzlicher Bewertungsrahmen speziell für die Problemlösekompetenz entwickelt. Die Ergebnisse (cf. [5] PISA 2012 S. 5) zeigten sehr große Unterschiede zwischen den einzelnen Ländern was die Fähigkeit der Schüler angeht, sowohl einfache wie auch komplexe Probleme zu lösen. Des Weiteren gab es innerhalb der Länder beträchtliche Differenzen zwischen den fachspezifischen Kompetenzen und der Fähigkeit, Probleme zu lösen.

Seit dieser ersten umfassenden Problemlösekompetenzerhebung wurde die Forschung in den Bereichen Lösen komplexer Problemstellungen, Transfer sowie rechnergestützte Prüfung der Problemlösekompetenz vorangetrieben. Für PISA 2012 wurde daher ein neuer Bewertungsrahmen entwickelt, der die neuesten Erkenntnisse berücksichtigt.

Die Forschung hat bestätigt, dass fachspezifische(s) Wissen und Strategien beim Problemlösen eine wichtige Rolle spielen (cf. 2.3). Die PISA 2012 Studie fokussierte auf die Messung der vom Fachwissen unabhängigen kognitiven Prozesse, die zur Problemlösung verwendet werden.

Die Forschung hat auch die Wichtigkeit von authentischen, relativ komplexen, Problemstellungen hervorgehoben. In dieser Studie war es dank der rechnergestützten Erhebung erstmals möglich, sogenannte interaktive Problemstellungen, bei denen der Problemlöser mittels Interaktion mit dem Problem wichtige Informationen herausfinden muss, einzubeziehen.

Im Kontext von PISA 2012 ist Problemlösekompetenz folgendermaßen definiert:

„Problem solving competency is an individual's capacity to engage in cognitive processing to understand and resolve problem situations where a method of solution is not immediately obvious. It includes the willingness to engage with such situations in order to achieve one's potential as a constructive and reflective citizen."

Die einzelnen zur Lösung eines Problems erforderlichen Prozesse lauten im Kontext der Studie (cf. [5] PISA 2012 S. 18):

- Erkunden und verstehen: Mit dem Problem interagieren, Informationen suchen und verstehen. Hier geht es darum, sich eine innere Darstellung der einzelnen Informationen zu erstellen.
- Darstellen und formulieren: Ein Modell des Problems wird erstellt, indem neue relevante Information ausgewählt, organisiert und mit bestehendem Wissen kombiniert wird.
- Planen und ausführen: Die Lösung wird geplant, gegebenenfalls indem mehrere Zwischenziele definiert werden. Anschließend wird der Plan ausgeführt.
- Überwachen und reflektieren: Das Erreichen der Zwischenziele und der Problemlösung überwachen und gegebenenfalls korrigierend eingreifen. Annahmen und Lösungen aus unterschiedlichen Blickwinkeln kritisch betrachten und nach Zusatzinformation Ausschau halten.

Das folgende Zitat, aus dem Bewertungsrahmen (cf. [5] PISA 2012 S. 7), stellt meines Erachtens den Zusammenhang zwischen der PISA Studie und der vorliegenden Arbeit sehr gut dar:

„*Problem solving competency can be developed by high quality education. Progressive teaching methods, like problem-based learning, inquiry-based learning, and individual and group project work, can be used to foster deep understanding and prepare students to apply their knowledge in novel situations. Good teaching promotes self-regulated learning and metacognition and develops the cognitive processes that underpin problem solving. It prepares students to reason effectively in unfamiliar situations, and to fill gaps in their knowledge by observation, exploration and interaction with unknown systems. The PISA 2012 computer-based assessment of problem solving aims to examine how students are prepared to meet unknown future challenges for which direct teaching of today's knowledge is not sufficient.*“

1.3 *PISA 2015*

Während PISA 2012 die individuelle Problemlösekompetenz untersuchte, soll bei der neuen Studie in zwei Jahren der Schwerpunkt auf der Problemlösung in einer Gruppe (collaborative problem solving (CPS)) liegen, wobei mehr Wert auf die Fähigkeit der Zusammenarbeit als auf die Problemlösekompetenz gelegt wird.

Als Hauptvorteile von CPS gegenüber der individuellen Problemlösung gelten Arbeitsteilung, das Einbinden von Informationen aus unterschiedlichen Quellen und Perspektiven, sowie erhöhte Kreativität und Lösungsqualität aufgrund der Ideen anderer Gruppenmitglieder (cf. [6] PISA 2015 S. 3).

Kritisches Denken, Problemlösen, Selbstmanagement, Informations- und Kommunikationstechnologiekompetenzen sowie Kommunikation und Zusammenarbeit gelten als Schlüsselelemente einer Erziehung, die den Herausforderungen des einundzwanzigsten Jahrhunderts gerecht wird.

In der heutigen Berufspraxis werden Probleme zunehmend in Gruppen, in einer global vernetzten Welt, gelöst. Daher ist ein wichtiger Aspekt der Vorbereitung der Schüler auf das Berufsleben sicherzustellen, dass sie effizient mit anderen Menschen zusammenarbeiten können, also über eine gut entwickelte Sozialkompetenz verfügen.

Das CPS Rahmenwerk baut auf dem PISA 2012 Rahmenwerk zur individuellen Problemlösung auf und erweitert es um die Gruppendimension mittels dreier gemeinschaftlicher Problemlösekompetenzen (cf. [6] PISA 2015 S. 10-11). Abbildung 1.1 zeigt die Interaktion zwischen den individuellen und den gemeinschaftlichen Problemlöseprozessen.

	(1) Establishing and maintaining shared understanding	**(2) Taking appropriate action to solve the problem**	**(3) Establishing and maintaining team organisation**
(A) Exploring and Understanding	(A1) Discovering perspectives and abilities of team members	(A2) Discovering the type of collaborative interaction to solve the problem, along with goals	(A3) Understanding roles to solve problem
(B) Representing and Formulating	(B1) Building a shared representation and negotiating the meaning of the problem (common ground)	(B2) Identifying and describing tasks to be completed	(B3) Describe roles and team organisation (communication protocol/rules of engagement)
(C) Planning and Executing	(C1) Communicating with team members about the actions to be/ being performed	(C2) Enacting plans	(C3) Following rules of engagement, (*e.g.*, prompting other team members to perform their tasks.)
(D) Monitoring and Reflecting	(D1) Monitoring and repairing the shared understanding	(D2) Monitoring results of actions and evaluating success in solving the problem	(D3) Monitoring, providing feedback and adapting the team organisation and roles

Abbildung 1.1: Matrix of Collaborative Problem Solving skills for PISA 2015 (cf. [6] PISA 2015 S. 11)

Das Rahmenwerk beruht auf der Annahme, dass die kognitiven und sozialen Fähigkeiten 15-jähri-

ger Schüler ausreichen, um die CPS-Aufgaben zu meistern. Diese Annahme wird vom aktuellen Forschungsstand validiert.

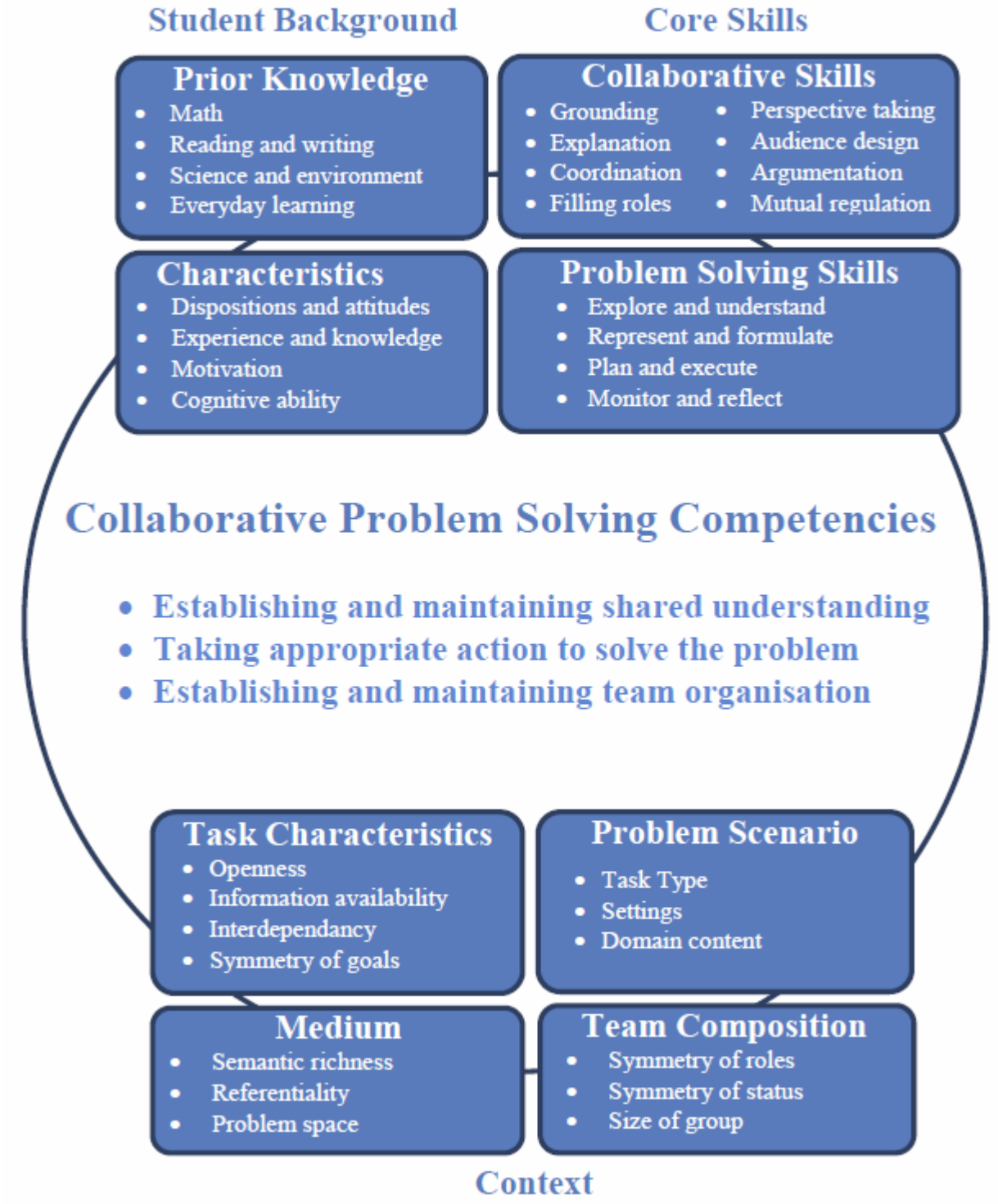


Abbildung 1.2: Overview of factors and processes for Collaborative Problem Solving in PISA 2015 (cf. [6] PISA 2015 S. 13)

Abbildung 1.2 gibt einen Überblick über die Faktoren, die die Kompetenz zur Problemlösung in der Gruppe beeinflussen, sowie die relevanten kognitiven und sozialen Prozesse. Die Gruppenzusammensetzung spielt eine wichtige Rolle im Hinblick auf die erfolgreiche gemeinschaftliche Problemlösung. Eine effektive Zusammenarbeit wird durch einen hohen Grad an Symmetrie in Bezug auf Wissensstand, Status und Ziele der einzelnen Gruppenmitglieder ermöglicht, wenngleich ihre Rollen und Aufgaben sehr unterschiedlich sein können (cf. [6] PISA 2015 S. 15).

Abbildung 1.3 zeigt die Kontextdimensionen und -zustände, die den Schwierigkeitsgrad einer gemeinschaftlichen Problemstellung beeinflussen.

Context	Dimension	States
Problem Scenario	Task type	*E.g.* Jigsaw, consensus building, negotiation
	Settings	Private vs. public Technology vs. non technology School (formal) vs. non-school (informal)
	Domain content	*E.g.* Math, science, reading, environment, community, politics
Team Composition	Size of group	2 or more (including the student)
	Symmetry of status of team members	Symmetrical vs. Asymmetrical
	Symmetry of roles: Range of actions available to each team member	Symmetrical vs. Asymmetrical
Task characteristics	Openness (c.f. PISA PS 2012)	Well-defined vs. Ill-defined
	Information availability: Does the student receive all necessary information at once? (c.f. PISA PS 2012)	Static vs. Dynamic
	Interdependency: Student A cannot solve problem without student B's acts)	Low to High
	Symmetry of goals	Group vs. individual
	Distance to solution (From beginning state to goal state)	Small, medium or large
Medium	Semantic richness	Low to High
	Referentiality to the outside world	Low to High
	Communication medium cost of grounding Interdependency: Student A cannot solve problem without student B's acts)	Low to High
	Problem space: does the student get information about other team members' actions?	Explicit vs. implicit

Abbildung 1.3: CPS context dimensions (cf. [6] PISA 2015 S. 16)

2 Theoretische Grundlagen von PBL

2.1 *Die Paradigmen des Lehrens und Lernens*

Ich teile die Meinung von Dubs (cf. [7] Dubs S. 24) wenn er schreibt: „*Lehren und Lernen bleibt ein dermaßen anspruchsvolles Unterfangen, das mit einseitigen Paradigmata und Ansätzen weder erklärbar noch wirksam umsetzbar ist.*“ Daher sehe ich PBL nicht als ein pädagogisches Allheilmittel.

In der Folge werde ich die in den vergangenen Jahrzehnten diskutierten Paradigmen des Lehrens und Lernens besprechen und den Zusammenhang zu PBL erkunden. Die Palette reicht vom Objektivismus an einem Ende zum Konstruktivismus am anderen Ende.

2.1.1 Objektivismus

Laut diesem Ansatz (cf. [7] Dubs S. 25) gibt es ein objektives Wissen, das die Vorgänge in der Welt weitestgehend erklärt und den Lernenden vermittelt werden kann. Die Lernenden übernehmen dieses Wissen und erhalten somit ebenfalls ein objektives Verständnis der Realität. Das Wissen über Denkprozesse wird gleichermaßen vermittelt und übernommen. Ziel der Lehrkraft ist es daher, „*den Lernenden zu helfen, die Inhalte des objektiven Wissens über die Welt in die Strukturen ihres Denkens zu übernehmen*“.

Behaviorismus und Kognitivismus stellen die zwei wichtigsten objektivistischen Lerntheorien dar.

2.1.1.1 *Behaviorismus*

Dieser Ansatz betrachtet den Menschen als ein fast ausschließlich von Umweltreizen gesteuertes Wesen das auf Belohnung und Bestrafung reagiert. Hier geht es primär darum, festgelegte und beobachtbare Verhaltensweisen zu unterrichten. Dazu teilt die Lehrkraft komplexere Lernvorgänge in einfachere Teile auf und steuert den gesamten Lernprozess. Korrektes Verhalten wird durch Lob verstärkt, inkorrektes Verhalten sofort korrigiert.

Die Wirksamkeit dieses Ansatzes beim Erlernen einfacher Vorgänge ist nachgewiesen. Zum Erreichen anspruchsvollerer Lernziele ist die Aneinanderreihung von kleinen Lernschritten jedoch nicht geeignet, da sie kein Gesamtbild vermitteln kann. Des Weiteren bleibt für die Förderung der Metakognition aufgrund der starken Steuerung des Lernprozesses durch die Lehrkraft kein Spielraum.

2.1.1.2 *Kognitiver Behaviorismus*

Dieser Ansatz versucht die Prinzipien des Behaviorismus „*durch Erkenntnisse aus der Kognitionspsychologie zu ergänzen*“ (cf. [7] Dubs S. 26) indem er versucht, die zum Erlernen und Verstehen neuen Wissens notwendigen Prozesse (Identifizieren, Analysieren und Systematisieren, Erinnern, Lösen von Problemen, Entwickeln neuer Ideen, ...) zu berücksichtigen. Anstatt das Gehirn wie beim Behaviorismus als „Black Box“ zu betrachten, wird versucht die Denkprozesse der Lernenden anzuregen und sie von ihnen durcharbeiten zu lassen. Des Weiteren wird versucht, die Metakognition der Schüler zu entwickeln, indem diese ihre einzelnen Verhaltensschritte selbst beschreiben und sich derer somit bewusster werden. Die starke Lehrerzentrierung mittels regelmäßigem Feedback bleibt jedoch auch bei diesem Ansatz erhalten.

Für das Erlernen von Grundlagen liefert dieser Ansatz insbesondere bei schwächeren Schülern gute Lernergebnisse. Da weiterhin mit kleinen Lernschritten gearbeitet wird, ist es jedoch fragwürdig, ob die Gesamtzusammenhänge hier verständlich werden. Außerdem scheint es für den Schüler einfacher, sich mit dem Lernen und Anwenden der vermittelten Kleinschritte zu begnügen, wodurch die Metakognition nicht wirklich gefördert wird.

2.1.1.3 *Traditioneller Kognitivismus*

Dieser Ansatz basiert auf den Erkenntnissen der Kognitionspsychologie und fokussiert auf die Verarbeitung von Wissen. Im Gegensatz zum Behaviorismus geht der kognitivistische Ansatz davon aus, dass die internen Verarbeitungsprozesse des individuellen Menschen, d.h. seine Wahrnehmung, Informationsverarbeitung und -bewertung, bei der Wissensverarbeitung eine wichtige Rolle spielen.

Das Ziel besteht darin, die Denkprozesse der Schüler anzuregen und zu erreichen, dass diese Prozesse von den Schülern selbst verstanden und gesteuert werden (cf. [7] Dubs S. 27 - 28). Aufgabe der Lehrperson ist es, eine Umgebung zu schaffen in der „*die Lernenden die reale Welt (objektives Wissen) verstehen. Deshalb sind kognitive Lernziele zu erreichen, bei denen aber nicht nur das Lernergebnis (Produkt), sondern auch der Lern- und Denkprozess bedeutsam sind (Kognition und Metakognition). Dies ist umso eher der Fall, je mehr die Lernenden die Gelegenheit zum aktiven Handeln und Denken erhalten und je stärker durch strukturierende Hilfen der beschränkten Aufnahmekapazität von Wissen Rechnung getragen wird.*

Denkprozesse werden nicht durch kleine, linear verabreichte Problemstellungen in Gang gesetzt, sondern durch die Vorgabe anspruchsvoller Aufgaben- und Problemstellungen, die bei den Lernen-

den eine Fragehaltung und einen Suchprozess auslösen, der zu Können und Einsichten führt, die auf andere Situationen übertragbar sein sollen (Transfer).

Besonders bedeutsam ist das Lernen in Gruppen, weil sie nicht nur zur gegenseitigen Anregung und als Korrektiv in den Lernprozessen dienen, sondern auch die sozialen Fähigkeiten stärken. Ganz wesentlich ist schließlich die Balance zwischen dem, was die Lehrkraft vermittelt und dem, was die Lernenden selbst erarbeiten. Wahrscheinlich wird es nie gelingen, das richtige Verhältnis auch nur annähernd zu bestimmen, weil die Lernvoraussetzungen bei den Schülerinnen und Schülern und die Fähigkeit der Lehrkräfte in lernwirksamer Weise zu intervenieren sehr verschieden sind.“

Der Kognitivismus geht von objektiv reproduzierbaren kognitiven Strukturen aus, die es aufzubauen gilt (cf. [8] Koubek S. 5).

2.1.2 Konstruktivismus

Die konstruktivistische Didaktik hat ihre theoretischen Ursprünge in den Ansätzen von John Dewey, Jean Piaget und Levy S. Wygotski (cf. [9] Reich S. 71-72).

Dewey betrachtet Lernen als einen aktiven Vorgang, „*der keineswegs äußere Wirklichkeiten abbildet, sondern in den Handlungsprozessen selbst erst herstellt.*“ Durch seine Erfahrungen im Umgang mit seiner Umwelt entwickelt der Mensch „*Verhaltensweisen, die dem Wissen einen Kontext, einen interpretativen Rahmen von Verwendung und Bedeutung geben, der für das Lernen unerlässlich ist.*“

Die konstruktive Psychologie von Piaget basiert auf den Untersuchungen der geistigen Entwicklungsstufen des Menschen vom Säuglingsalter bis zur Adoleszenz. Laut seiner Forschung „*bilden die psychologischen Strukturen die Grundlagen der geistigen Aktivität. Sie sind das Ergebnis eines vielschichtigen Zusammenwirkens biologischer Faktoren und solcher der Erfahrung.*“ (cf. [10] Ginsburg, Opper S. 32). Ein wichtiger Faktor sind die allgemeinen Funktionsprinzipien der Organisation und Anpassung. Mit ersterem meint Piaget die Tendenz aller Lebewesen, ihre Prozesse zu organisieren. Die Anpassung unterteilt er in Assimilation und Akkommodation. Die Akkommodation stellt die Beeinflussung der geistigen Strukturen aufgrund von Einwirkungen aus der Umwelt dar. Die Konfrontation mit einem Problem, das wir mit unserer bisherigen Erfahrung nicht lösen können, stellt also den Impuls und Organisator für unseren Lernprozess dar. Die Assimilation bezeichnet die Verwendung vorhandener Strukturen um mit der Umwelt umzugehen. Somit entwickeln sich die psychologischen Strukturen im Laufe des Lebens. Ein weiterer wichtiger Aspekt von Piaget's

Theorie ist das Gleichgewichtsbestreben, d.h. wir versuchen eine Balance zwischen unseren psychologischen Strukturen und der Welt herzustellen.

Wygotski betont die Bedeutung der sozialen Interaktion für den Lernprozess. Diese Interaktion steigert den Lerneffekt, da „s*oziale Prozesse und Werkzeuge des Handelns in psychische Forderungen umgesetzt werden, die Lerner antreiben, ein neues Niveau des Wissens und Verhaltens zu erreichen*“ (cf. [9] Reich S. 72). Laut ihm gestalten wir unseren Lernprozess aktiv und berücksichtigen dabei das kulturelle Umfeld sowie unsere Handlungsperspektive. Die berufliche Praxis bestätigt die Bedeutung der sozialen Interaktion. Dies gilt insbesondere in der Informatik, wo die Mehrheit der Projekte zahlreiche „Stakeholders“ betreffen, deren oft sehr unterschiedlichen Ziele nur durch eine gut entwickelte Sozialkompetenz zu Tage befördert, strukturiert und für alle verständlich formuliert werden können. Soziale Interaktion ist jedoch nicht nur während dieser Phase der Problemanalyse von hoher Bedeutung, sondern auch während des restlichen Projektlebenslaufes, vom Design über Implementierung bis hin zu Test und Abnahme sowie der folgenden Iterationen durch diese einzelnen Projektphasen. Dies liegt nicht zuletzt auch daran, dass nichttriviale Projekte nicht von Einzelpersonen sondern von Teams realisiert werden und somit eine gute Lösung nur durch effektive soziale Interaktion entwickelt werden kann. Darüber hinaus gibt es bei realen Projekten meist nicht eine einzelne Person die alles weiß und kann und mit allen Aspekten des Projekts Erfahrung hat. Somit benötigen wir die Rückmeldung anderer um alternative Sichten zu entdecken, was unseren Lernprozess stimuliert.

Laut dem konstruktivistischen Ansatz gibt es kein objektives, sondern nur subjektives Wissen, das jede Person aufgrund ihrer eigenen Erfahrungen und Interpretationen selbst konstruiert. Jeder Mensch entwickelt also seine eigene Perspektive der Realität. Hier wird die zweite wichtige Rolle der sozialen Interaktion erkennbar: wenn es kein objektives Wissen gibt, dann entstehen die „Tatsachen“ dadurch, dass alle Beteiligten sich auf diese geeinigt haben (cf. [11] Savery & Duffy S. 4).

Die pädagogische Herausforderung besteht nun darin, „*den Lernenden Erlebnisse zu verschaffen und Probleme vorzulegen, damit sie ihr Wissen und Können selbst aktiv aufbauen können, denn nur dann verstehen sie es. Wissen kann deshalb nicht passiv übernommen werden, weil es einerseits nicht beschreibbar ist, und andererseits bei einer passiven Übernahme nur angelerntes, aber nicht verstandenes und jederzeit anwendbares Wissen ist*“ (cf. [7] Dubs S. 25).

2.2 *Erkenntnisse der Neurodidaktik*

Der Neurowissenschaft ist es in den letzten Jahren, insbesondere mit Hilfe von bildgebenden Verfahren, gelungen, unser Verständnis der Lernprozesse im menschlichen Gehirn erheblich zu erweitern. So konnte man die Nervenzellen im Hippocampus, einem Teil des menschlichen Gehirns das beim Lernen eine wichtige Rolle spielt, beim Aufbauen der Repräsentationen von neuen Vokabeln beobachten.

Unser Gehirn speichert Information mittels der Verbindungs- oder Synapsenstärken zwischen einzelnen Neuronen. Mit anderen Worten, Wissen ist nicht explizit gespeichert, sondern vielmehr führt ein gewisses Inputmuster zu einem definierten Output. So können wir z.B. gehen ohne diesbezügliches Wissen, d.h. die physiologischen Regeln, gespeichert zu haben. Daher Manfred Spitzer's Feststellung: „*Fast alles, was wir gelernt haben, wissen wir nicht. Aber wir können es.*“ (cf. [12] Spitzer S. 59)

Spitzer (cf. [12] S. 73) beschreibt den Lernvorgang wie folgt: „*Man konnte nun zeigen, dass sich neuronale Netzwerke bei entsprechendem Training mit Beispielen ebenso verhalten wie Kinder: Sie lernen zuerst die Ausnahmen, dann die Regel (und machen Fehler, indem sie überregularisieren) und können schließlich die Regel und die Ausnahmen. Allein dadurch also, dass Synapsenstärken im Netzwerk langsam in Abhängigkeit von den Lernerfahrungen verändert werden, kommt es dazu, dass das Netzwerk eine Regel kann. Es 'weiß' um diese Regel ebenso wenig wie die Kinder. Dieses Wissen ist jedoch für das Können völlig unerheblich.*“

Des Weiteren ergibt sich (cf. [12] Spitzer S. 76): „*Das Lernen von einzelnen Fakten oder Ereignissen ist daher meist nicht nur nicht notwendig, sondern auch ungünstig. Unser Gehirn ist – abgesehen vom Hippocampus, der auf Einzelheiten spezialisiert ist – auf das Lernen von Allgemeinem aus. Dieses Allgemeine wird aber nicht dadurch gelernt, dass wir allgemeine Regeln lernen. - Nein! Es wird dadurch gelernt, dass wir Beispiele verarbeiten und aus diesen Beispielen die Regeln selbst produzieren.*“

Für den Unterricht ergibt sich, dass viele und gute Beispiele von entscheidender Bedeutung sind, auch für das Lernen von Regeln. Denn, „*nur dann, wenn die Regel immer wieder angewendet wird, geht sie vom expliziten und sehr flüchtigen Wissen im Arbeitsgedächtnis in Können über, das jederzeit wieder aktualisiert werden kann.*“ (cf. [12] Spitzer S. 78)

Eine Grundvoraussetzung für den Lernprozess ist die Aktivierung der Synapsen. Diese findet jedoch

nur statt, wenn wir aufmerksam sind. Um etwas Bestimmtes zu lernen, muss unsere selektive Aufmerksamkeit darauf gerichtet sein, um so diejenigen Hirnareale zu aktivieren, die für das Lernen dieser Lerninhalte zuständig sind.

Einen nicht zu unterschätzenden Einfluss auf den Lernprozess haben Emotionen. Studien haben die alte „*Erkenntnis, dass emotionale Beteiligung das Lernen erheblich verbessert*“ (cf. [12] Spitzer S. 159) bestätigt. Gleiches gilt für soziale Interaktion, sprich Gruppenarbeit. Die besten Ergebnisse werden mit positiven Emotionen erzielt. Spitzer (cf. [12] S. 181) fasst dies folgendermaßen zusammen: „*Gelernt wird immer dann, wenn positive Erfahrungen gemacht werden, wobei klar sein muss, dass für den Menschen die positive Erfahrung schlechthin in positiven Sozialkontakten besteht. Menschliches Lernen vollzieht sich immer schon in der Gemeinschaft, und gemeinschaftliche Aktivitäten bzw. gemeinschaftliches Handeln ist wahrscheinlich der bedeutsamste 'Verstärker'.*“

Die Motivation der Lernenden wird nicht zuletzt durch die Motivation des Lehrers beeinflusst. Ein Lehrer, der Begeisterung ausstrahlt und interessante Geschichten zu erzählen vermag, wird eine größere Aufmerksamkeit erlangen.

Die Neurodidaktik bestätigt auf neurologischer Ebene das konstruktivistische Paradigma. Es ist nur im Kontext seiner eigenen Erfahrungen wo der Schüler lernen kann. „*Hunger produziert sich jeder selbst, und Lernen produziert sich auch jeder selbst. Jeder auf seine Weise; und jeder lernt auch auf seine Weise und eben genau dasjenige, was in das Gefüge seiner Synapsengewichte am besten passt.*“ (cf. [12] Spitzer S. 417)

2.3 *Problemlösekompetenz*

In seinem Artikel „*Cognitive, metacognitive, and motivational aspects of problem solving*“ geht Mayer (cf. [13]) der Frage auf den Grund, was die Voraussetzungen für erfolgreiches Problemlösen sind. Er kommt zum Schluss, dass es deren drei gibt:

1. *Skill*: Fachspezifisches Wissen ist unabdingbar. Um ein Programmierproblem lösen zu können, muss der Schüler über ausreichendes Wissen bezüglich der zu verwendenden Programmiersprache verfügen oder sich dieses erarbeiten. So kann man Probleme aus der Sicht von z.B. der Bloom Taxonomy analysieren und in eine Reihe von Lernzielen aufteilen, die dann von den Schülern erreicht werden sollen. Allerdings reicht das Erreichen dieser Lernziele an sich nicht aus, um noch nicht gesehene Problemstellungen effektiv zu lösen. Dies ist das klassische Phänomen, dass Schüler ähnliche Problemstellungen mit Leichtigkeit lösen, von

abweichenden Problemstellungen aber völlig überfordert sind, da sie ihr Wissen nicht transferieren können.

2. *Metaskill*: Problemlösen erfordert neben Fachwissen auch zu wissen, wann welches Wissen wie eingesetzt werden soll. Dies kann nur in einem realistischen Kontext anhand von konkreten Problemstellungen erlernt werden. Eine sehr erfolgreiche Technik um die Metaskills der Schüler zu entwickeln ist die Lösung eines konkreten Problems durch die Lehrkraft, wobei sie den Lernenden ihre Denkprozesse erklärt. Das Problem hierbei ist jedoch, dass, angenommen die Lehrkraft ist Expertin auf dem Gebiet, es ihr nicht leicht fallen wird, ihre Vorgehensweise zu beschreiben, da sie gewohnt ist, das Problem intuitiv, aufgrund ihrer Erfahrung, zu lösen und sich somit ihrer eigenen Denkprozesse dabei wahrscheinlich gar nicht bewusst ist.
3. *Will*: Wie die Neurodidaktik (cf. 2.2) inzwischen bewiesen hat, ist Motivation eine entscheidende Voraussetzung für jeglichen Lernprozess. Daher ist es sehr wichtig, Problemstellungen zu wählen, die ein starkes Interesse seitens der Schüler hervorrufen.

2.4 *Ursprünge und Definition von PBL*

Wie Reich (cf. [14] S. 1) beschreibt, lassen sich die Ursprünge von problemorientiertem Lernen mindestens bis auf Sokrates zurückführen. Aus pädagogischer Sicht leistete Dewey in seiner Laborschule um 1900 Grundlagenarbeit (cf. [15] Wikipedia).

1960 dokumentiert Shoemaker (cf. [16]) seine Erfahrungen mit der „*functional context method of instruction*“ in der Ausbildung von Radioreparaturfachleuten. Das Hauptmerkmal dieses Ansatzes ist die Betonung der funktionalen Bedeutung eines Themas, d.h. ihrer praktischen Relevanz. Diese liefert den Kontext innerhalb dessen der Lerner sich ein immer detaillierteres Verständnis des Themas erarbeitet. Shoemaker illustriert seine induktive Methode anhand der Ausbildung zur Radioreparatur. Hier werden die Grundlagen der Elektronik im Rahmen der allgemeinen Funktion eines Radiogeräts unterrichtet. Erst wenn der Lerner die Funktion des Radios auf höchster Ebene verstanden hat, werden die Bestandteile näher betrachtet.

Die traditionelle Sichtweise ist genau umgekehrt: erst nachdem der Lerner die allgemeinen Funktionsprinzipien beherrscht, kann er Problemlösungen verstehen, die die Anwendung dieser Prinzipien beinhalten. Hierbei handelt es sich um eine deduktive Unterrichtsmethode. Diese bietet Anlass zur Kritik aus folgenden Gründen:

1. Die Schüler sollen abstrakte Prinzipien sozusagen im luftleeren Raum, ohne einen direkten Zusammenhang mit einer praktischen Anwendung, erlernen. Analogien schaffen hier nur bedingt Abhilfe, da sie oft nicht wirklich relevant für das Lernziel sind. Ein Beispiel in der Informatik wäre die Einführung von Schleifen. Um den Sinn davon zu illustrieren, kann man einfache Beispiele verwenden wie einen Countdown oder die Berechnung des größten gemeinsamen Teilers. Diese stellen jedoch nur einen sehr kleinen Anwendungskontext dar.
2. Die Motivation, etwas zu lernen, rührt in der Praxis meist daher, dass man ein konkretes Problem lösen möchte. Ohne ein solches Problem fehlt der Lernstimulus und das Lernen beschränkt sich auf Wissensakkumulation, die für den Lerner, außerhalb des Bestehens der nächsten Prüfung, keine praktische Relevanz hat. Dies ist ein den meisten Lehrern nur allzu bekanntes Problem und wird von der Neurowissenschaft bestätigt. So schreibt Spitzer (cf. [12] S. 35): *„Einzelheiten machen nur im Zusammenhang Sinn, und es ist dieser Zusammenhang und dieser Sinn, der die Einzelheiten interessant macht. Und nur dann, wenn die Fakten in diesem Sinne interessant sind, werden wir sie auch behalten.“*

In den 70er Jahren hat sich PBL insbesondere an der kanadischen McMaster Universität für die medizinische Ausbildung entwickelt. Angesichts des sich rapide entwickelnden theoretischen Wissens, der zunehmenden Spezialisierung, des verstärkten Einsatzes neuer Technologien sowie der sich verändernden Praxisansprüche zeigte sich, dass die traditionellen Vorlesungen nicht geeignet waren, den Studenten den Kontext sowie die klinische Anwendung des unterrichteten Wissens zu vermitteln. PBL wurde eingeführt um eine lernerzentrierte Ausbildung, die lebenslanges Lernen fördert, zu ermöglichen (cf. [11] Savery S. 7 - 8).

Die Begeisterung für PBL hat sich insbesondere seit den 90er Jahren auch außerhalb des medizinischen Bereiches gesteigert. In zahlreichen Ländern wurde der Ansatz in sehr unterschiedlichen Bereichen, von Naturwissenschaften über Sprachen, Soziologie, Politik, Musik bis hin zum Sport (cf. [14] Reich S. 6 – 7) eingesetzt.

Den Ausgangspunkt stellt ein zu lösendes Problem dar, das der Lerner lösen möchte. Die Motivation, das Problem zu lösen, führt den Lerner dazu, sich das zur Lösung notwendige Wissen und die erforderlichen Kompetenzen anzueignen. „*Ziel von PBL ist die Entwicklung der Kompetenzen zum konkreten Handeln. Die Problemlöse-Kompetenz bildet hierfür über das Auflösen von möglichst authentisch konstruierten Problemen den Schwerpunkt, wobei die fachliche, soziale, personale und methodische Kompetenz explizit mit einbezogen wird.*“ (cf. [14] Reich S. 1) Damit rückt die Wahl

und Spezifikation der Problemstellung in den Vordergrund, da sie fest legt, „*was gelernt werden muss, um die Situation zu entschlüsseln und die grundlegenden Fakten und Zusammenhänge zu verstehen*“.

2.5 *PBL als konstruktivistischer Unterrichtsansatz*

Der problembasierte Unterrichtsansatz passt theoretisch sehr gut zum konstruktivistischen Paradigma des Lehrens und Lernens. Im Folgenden betrachten wir PBL anhand der Hauptmerkmale des Konstruktivismus (cf. [7] Dubs S. 30):

1. „*Es gibt kein objektives Wissen. Wissen als Prozess und Produkt wird individuell konstruiert.*“ In der Berufspraxis geht es darum, reale Probleme zu lösen. Diese Probleme treiben den Prozess der Entdeckung und Entwicklung von Wissen und konkreten Lösungen. Somit konstruiert jeder Mensch sich sein eigenes Wissen aufgrund der Problemstellungen, mit denen er sich auseinandergesetzt hat.

2. „*Inhaltlich muss sich der Unterricht an komplexen, lebens- und berufsnahen, ganzheitlich zu betrachtenden Erlebnis- und Problembereichen (authentische Probleme) orientieren. Nicht vereinfachte (reduktionistische) Modelle, sondern die Realität (unstrukturierte Probleme) sind zu betrachten, denn verstehen lässt sich etwas nur, wenn es im komplexen Gesamtzusammenhang als Problem erfasst ist, dann Einzelheiten im größeren Zusammenhang betrachtet und vertieft und schließlich wieder in den Gesamtzusammenhang gebracht werden.*“ Dies entspricht der Grundidee von PBL. Es geht dabei natürlich nicht darum, beispielsweise ein Atomkraftwerk in einem Schulsemester zu planen und zu realisieren. Stattdessen kommt es vielmehr darauf an, die Schüler mit den gleichen Typen kognitiver Herausforderungen zu konfrontieren wie sie in der Berufspraxis auftreten. Innerhalb dieser komplexen Problemstellung lassen sich Lernaktivitäten aller Art unterbringen, solange der Lernende die Relevanz zum Gesamtproblem klar erkennt (cf. [11] Savery & Duffy S. 3).

3. „*Lernen kann nur in einem aktiven Prozess geschehen, weil allein aus eigenen neuen Erfahrungen und Erkenntnissen das individuell vorhandene Wissen und Können als Ganzes (in seiner Struktur) verändert und personalisiert wird, d.h. auf das eigene Interpretieren und Verstehen ausgerichtet wird.*“ Daher ist es wichtig, dass die Schüler sich nicht nur mit der Problemstellung identifizieren, sondern auch Kontrolle über ihren eigenen Lernprozess haben. Somit sollte dieser nicht von der Lehrkraft diktiert werden. Auch sollten die Lerninhalte

nicht vorgegeben werden sonst riskiert die Problemstellung zum reinen Anwendungsbeispiel zu verkommen (cf. [11] Savery & Dubs S. 5). Diese theoretische Erkenntnis in der Praxis vollständig umzusetzen hat sich jedoch als schwierig erwiesen (siehe 4).

4. „*Wesentlich ist das Lernen in Gruppen (soziales oder kollektives Lernen), denn erst die Diskussion der individuellen Interpretation und des persönlichen Verstehens, der entworfenen Hypothesen und möglicher Lösungen trägt dazu bei, die eigene Interpretation zu überdenken oder die gewonnen Erkenntnisse anders (besser) zu strukturieren. In diesem Sinn regulieren die Schülerinnen und Schüler ihr Lernen selbst und halten es auch dauernd in Gang.*“ In diesem Zusammenhang spielt die Entwicklung der Sozialkompetenz der Schüler eine wichtige Rolle. Sie sollen lernen mit konstruktiver Kritik umzugehen und die Sichtweisen anderer zu verstehen und gegebenenfalls ganz oder teilweise in ihr eigenes Verständnis zu integrieren.

5. „*Bei diesem selbstgesteuerten sozialen Lernen sind Fehler sehr bedeutsam. Diskussionen in Lerngruppen sind nur sinnvoll, wenn Fehler geschehen und diese diskutiert und korrigiert werden. Die Auseinandersetzung mit Fehlüberlegungen wirkt verständnisfördernd und trägt zur besseren Konstruktion des Wissens bei.*“ Die Lehrkraft spielt hierbei eine wichtige Rolle, indem sie reflexives Denken vorführt und die Schüler anregt, sowohl über das Gelernte wie auch über ihren Lernprozess nachzudenken und gegebenenfalls Korrekturen vorzunehmen.

6. „*Die komplexen Lernbereiche sind auf die Interessen der Schülerinnen und Schüler auszurichten, weil am leichtesten aus Erfahrungen gelernt werden kann, die als interessant oder herausfordernd empfunden werden.*“ Die Lernenden müssen sich mit dem Problem identifizieren, es als ihr eigenes betrachten, denn es sind die Ziele des Schülers, die seinen Lernprozess größtenteils bestimmen, nicht die der Lehrkraft (cf. [11] Savery & Duffy S. 4). Somit ist eine Grundvoraussetzung dafür zu sorgen, dass die Ziele des Lernenden mit denen des Lehrenden konsistent sind. Um dies zu erreichen, kann man entweder die Schüler die Problemstellungen bestimmen lassen oder aber Problemstellungen vorgeben, bei denen man annehmen kann, dass sie auf hohe Akzeptanz stoßen werden.

7. „*Konstruktivismus beschränkt sich nicht bloß auf die kognitiven Aspekte des Lehrens und Lernens. Gefühle sowie persönliche Identifikation sind außerordentlich bedeutsam, denn kooperatives Lernen, Umgang mit Fehlern in komplexen Lernsituationen, Selbststeuerung und Eigenerfahrung verlangen mehr als nur Rationalität.*“ Dies ist ein Grund für meine Wahl

von spielerischen Problemstellungen (siehe 4) um die emotionale Motivation zu steigern. Dies insbesondere bei Jugendlichen, die bekanntlich bereits sehr viel Erfahrung als Konsumenten von Computerspielen haben. Diese haben für sie eine nicht unerhebliche Bedeutung.

8. „*Weil eine eigene Wissenskonstruktion und nicht die passive Wissensaufnahme und -reproduktion angestrebt wird, darf die Evaluation des Lernerfolgs nicht auf Lernprodukte (mit ausschließlich richtigen und falschen Lösungen) ausgerichtet werden, sondern zu überprüfen sind die Fortschritte bei den Lernprozessen, und dies wiederum in komplexen Lernsituationen.*“ In ihrer Rolle als Tutor verbringt die Lehrkraft mehr Zeit mit den einzelnen Schülern als bei herkömmlichem Unterricht und hat somit die Möglichkeit, deren Entwicklung anhand der Bearbeitung der Problemstellungen kontinuierlich mitzuverfolgen.

Dubs beschreibt, dass sich im Laufe der Zeit viele Ausprägungen des Konstruktivismus herausgebildet haben, die sich hauptsächlich darin unterscheiden, wie viel Wissen die Lernenden sich selbst erarbeiten und wie viel Wissen ihnen die Lehrkraft bereit stellen soll, sowie wie stark die Lehrkraft die Schüler bei Wissensbeschaffung und Problemlösung unterstützt. Meine Erfahrungen (siehe 6) zeigen, dass dies einen sehr wichtigen Faktor bei der Umsetzung von PBL darstellt.

Die Bandbreite reicht vom radikalen bis zum gemäßigten Konstruktivismus. Bei ersterem werden nur die „*Lernvoraussetzungen (starke Lernumgebung mit komplexen Problemstellungen, welche zu anregenden Lernsituationen führen)*“ geschaffen, „*damit die Schülerinnen und Schüler im Wechselspiel von neuen Erfahrungen sowie bisherigem Wissen und Können in Lerngruppen ohne wesentliche Hilfe durch die Lehrperson neues Wissen konstruieren und ihr Verstehen selbständig ausweiten. Da die Lernarbeit weitgehend selbständig in Gruppen erfolgt, wird auch von Sozialkonstruktivismus gesprochen.*“ (cf. [7] Dubs S. 31)

Beim gemäßigten Konstruktivismus steht auch die „*Arbeit an komplexen Problemen im Unterricht im Vordergrund. Die Lehrkräfte bieten aber mehr anleitende (unterstützende) Hilfen an. Sie verzichten aber auf die Vermittlung von Strukturen und Strategien sowie auf das Modelllernen, sondern Hilfen werden nur soweit angeboten, als sie von den Lernenden zur Fortführung der eigenen Denkprozesse notwendig sind.*“ Die Lernenden sollen anhand der Auseinandersetzung mit den Problemstellungen ihre eigenen Lernprozesse entwickeln und ihr Wissen und Können in einer ihnen verständlichen (nicht von außen vorgegebenen) Form internalisieren (Dubs beruft sich hier auf L. S. Vygotsky's Werk „Thought and Language“ von 1962).

Die Kritik am Konstruktivismus lässt sich laut Dubs (cf. [7] S. 32 – 33) folgendermaßen zusammen-

fassen:

1. Nicht alles Wissen muss unbedingt konstruiert sondern kann auch vermittelt werden.
2. Die Zeitverhältnisse in der Schule verbieten einen puren konstruktivistischen Ansatz.
3. Ein rein konstruktivistischer Ansatz kann auf die Dauer auch uninteressant werden.
4. Die „*Überlegenheit des Konstruktivismus bezüglich Lernwirksamkeit [ist] (noch?) nicht belegt*“.
5. Es „*muss noch viel systematischer untersucht werden, ob das Erarbeiten und das Einüben von grundlegenden Fertigkeiten in komplexen Themenbereichen in genügendem Ausmaß erfolgen kann oder ob es im alltäglichen Unterricht bei der ausschließlichen Lernarbeit an Problembereichen nicht eher zu einer Vernachlässigung der Fertigkeiten in den einzelnen Fachbereichen kommt*“.
6. Macht es Sinn alles selbst zu erarbeiten? Es gibt Lerninhalte, die sich effizienter vermitteln als konstruieren lassen. Dies gilt z.B. für die Syntax einer Programmiersprache, die man die Schüler nicht selbst konstruieren lässt, sondern die man entweder selbst erklärt oder man verweist auf die zahlreichen Bücher und Informationsquellen im Internet.
7. In heterogenen (bzgl. Vorwissen und Kompetenzen) Klassen kann es zu organisatorischen Problemen kommen, da eventuell nicht alle die gleiche Problemstellung bearbeiten können oder dafür unterschiedlich viel Zeit benötigen. Vorschläge wie man hiermit umgehen kann mache ich in 6.1.
8. Im realen Leben spielt neben der Gruppenarbeit auch die Einzelarbeit eine wichtige Rolle.

2.6 *Problementwicklung*

Problemstellungen bilden das Fundament von PBL. Probleme unterscheiden sich bezüglich Inhalt, Form und Prozess (cf. [17] Jonassen S. 3). Die Fähigkeit, ein Problem zu lösen, basiert auf einem Schema für diesen Problemtyp. Verfügt der Lernende über ein vollständiges Schema für den gegebenen Problemtyp, so braucht er dieses nur anzuwenden. Ein solches Schema ist das Ergebnis vorheriger Auseinandersetzungen mit derartigen Problemstellungen und ist letztlich das, was den Experten zum Experten macht: er erkennt das passende Schema um ein spezifisches Problem zu lösen und wendet es an. Der PBL-Neuling verfügt über keinerlei derartige Schemata und muss daher versuchen, mit allgemeinen Strategien das Problem zu lösen, was in der Regel zu schwächeren Ergeb-

nissen führt.

Die Problemlösekompetenz ist eine Funktion der Problemart, der Art und Weise, wie es dargestellt wird, sowie einer Reihe von individuellen Unterschieden, die den Lösungsprozess beeinflussen (cf. Abbildung 2.1).

Problem Variations →	**Representation** →	**Individual Differences** =	**Problem Solving Skill**
Ill-structuredness	Context	Domain knowledge	
Complexity	social	familiarity	
Abstractness/	historical	perplexity	
situatedness	cultural	experience	
(domain specificity)	Cues/Clues	Structural knowledge	
	Modality	Procedural knowledge	
		Systemic/conceptual knowledge	
		Domain-specific reasoning	
		Cognitive styles	
		General problem-solving strategies	
		Self-confidence	
		Motivation/perseverence	

Abbildung 2.1: Problem-solving skills (cf. [17] Jonassen S. 4)

Eine gute Problemstellung muss eine Reihe von Bedingungen erfüllen. Weiss (cf. [18] S. 1) schlägt zwei Etappen vor. Zuerst muss bestimmt werden, was der pädagogische Zweck der Problemstellung ist, mit anderen Worten, was wollen wir damit erreichen? Im nächsten Schritt wird dann ein Problem entworfen, das es erlaubt dieses Ziel zu erreichen.

Duffy und Cunningham (cf. [18] Weiss S. 2) geben fünf sinnvolle Zweckbestimmungen für Problemstellungen an:

1. Um den Schüler an spezifische Inhalte oder Konzepte heranzuführen.
2. Als Test, um zu überprüfen, ob der Schüler die Lerninhalte anwenden kann.
3. Um die Schüler sich Prinzipien, Konzepte oder Prozeduren anhand der Problemstellung selbst erarbeiten zu lassen. Die Problemstellung ersetzt hier die Vorlesung.
4. Um die Denkprozesse der Schüler zu fördern.
5. Um die Schüler auf möglichst hoher Ebene zu aktivieren. Ein Problem, das diesen Zweck erfüllt, wird durch seinen Mangel an Struktur die Schüler dazu bringen, einen Lösungsansatz zu strukturieren.

Die Erstellung eines guten und realistischen Problemfalls erfordert die didaktische Auseinandersetzung mit dem Zielpublikum sowie der im Lehrplan vorgegebenen Kompetenzen und der verfügbaren Zeit. Hierbei ist es sehr wichtig, den Anschluss an das vorhandene Wissen und Können der Schüler zu finden, was in der Praxis aufgrund der Heterogenität einer Klasse nicht einfach ist.

Einerseits soll das Problem einen kognitiven Konflikt herbeiführen, der für den Schüler die Herausforderung darstellt, seine derzeitigen Kompetenzen, die für die Lösung des Problems nicht ausreichen, derart weiter zu entwickeln, dass er das Problem lösen kann. Andererseits darf es jedoch nicht so anspruchsvoll sein, dass der Schüler mit seinem aktuellen Entwicklungsstand gar keine Chance sieht, das Problem zu lösen und verzweifelt aufgibt.

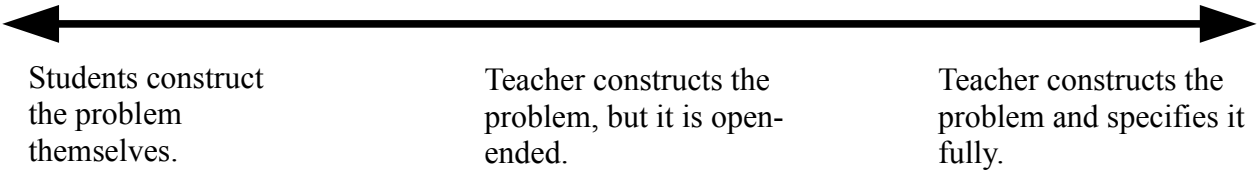


Abbildung 2.2: Spektrum der Problemdefinition (cf. [19] Ellis et al. S. 42)

Um durchwegs den richtigen Einstiegspunkt im PBL-Unterricht zu finden, ist eine differenzierte Vorgehensweise, von Ellis et al. als Spektrum der Problemdefinition (siehe Abbildung 2.2) bezeichnet, erforderlich. Die Differenzierung bezieht sich auf das zu behandelnde Lernthema sowie den Entwicklungsstand der Schüler. Bei Schülern, die keine PBL-Erfahrung haben, empfiehlt es sich, zunächst strukturierte und vom Lehrer spezifizierte Problemstellungen zu verwenden, dies insbesondere bei Themen, wo klar definierte kompetenzorientierte Ziele vorliegen. Dies verringert das Risiko, dass die Lerner aufgrund ihrer noch wenig entwickelten Problemlösekompetenz und Metakognition die Lernziele verfehlen. In diesem Fall sind auch starke Unterstützungsmaßnahmen seitens der Lehrkraft erforderlich, sowohl was den Prozess wie auch die benötigten Ressourcen zur Lösung des Problems betrifft. Wenn jedoch der Lernprozess, die Abstraktion oder die Festigung von bereits Gelerntem im Vordergrund steht und die Schüler bereits PBL-Erfahrung haben, dann sind offenere Problemstellungen und im Idealfall von den Schülern selbst entwickelte Problemfälle anzustreben, da dadurch die Motivation sowie die Kompetenzentwicklung am stärksten gefördert werden (cf. [19] Ellis et al. S. 43).

Jonassen (cf. [20] S. 8) hat eine Typologie von Problemtypen anhand ihres Strukturgrades erstellt:

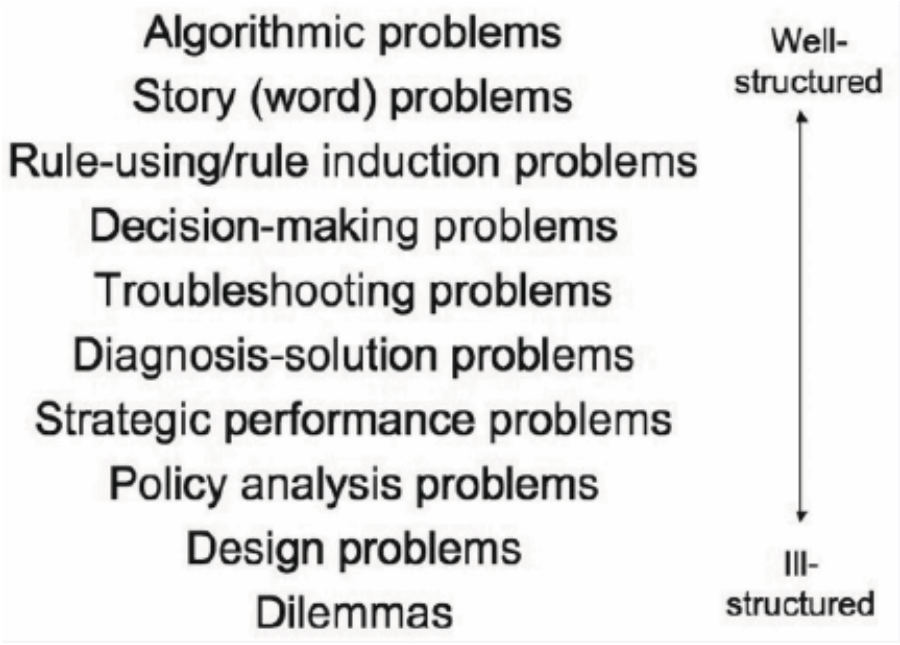


Abbildung 2.3: Typologie der Problemtypen (cf. [20] Jonassen S. 8)

In der Informatik sind realistische Problemstellungen zumeist vom Typ Design, d.h. sie erfordern die Entwicklung einer akzeptablen Lösung anhand einer offenen Problemstellung ohne klaren Lösungsweg.

Der Schwierigkeitsgrad einer Problemstellung hängt nicht nur vom Grad der Strukturiertheit, sondern auch von der Komplexität ab, welche an das Niveau des Schülers angepasst werden muss.

Neben der Differenzierung entlang der Problemdefinitionsachse und der Komplexität kann man zudem die schnellen Schüler bitten, ihre langsameren Kollegen zu unterstützen. Dies kann für alle Beteiligten bis zu einem gewissen Grad produktiv sein, da der Empfänger der Unterstützung im Optimalfall schneller voran kommt und der Sender durch seine Erklärungen sein eigenes Verständnis auf eine höhere Ebene bringt und sich seiner eigenen Denkprozesse bewusster wird, d.h. seine Metakognition wird gefördert, was ihm wiederum erlauben wird, sich weiter zu entwickeln und noch anspruchsvollere Problemstellungen zu bewältigen. Außerdem wird auch die soziale Kompetenz der beteiligten Schüler gefördert.

Jedoch ist hier unbedingt darauf zu achten, dass die Unterstützung hauptsächlich auf metakognitiver Ebene abläuft d.h., dass der Helfer nicht einfach das Problem für den schwächeren Schüler löst, sondern ihm beim Lösungsprozess hilft, indem er ihn mittels Fragen auf der richtigen Spur hält, die

zur Erarbeitung der Lerninhalte führt.

Andererseits kann man auch festlegen, dass alle Schüler bestimmte Problemstellungen meistern müssen, welche insgesamt das komplette Lehrprogramm abdecken, während es für die fortgeschrittenen optionale Problemstellungen gibt, die sie angemessen fördern und über die zum Bestehen des Kurses erforderlichen Aspekte hinausgehen.

Gut strukturierte Problemstellungen sind hauptsächlich für den ersten und dritten der oben erwähnten Zwecke geeignet, d.h. um Schüler an spezifische Inhalte und Konzepte heranzuführen oder um sie Prinzipien, Konzepte oder Prozeduren anhand der Problemstellung erarbeiten zu lassen. Solche Problemstellungen sind auch von mit PBL unerfahrenen Schülern vergleichsweise leicht zu bewältigen, sind jedoch eher akademischer Natur und in der Berufswelt selten anzutreffen.

Realistische Problemstellungen sind oft wenig bis gar nicht strukturiert und es ist bisweilen nicht ersichtlich, ob es überhaupt eine oder mehr als eine akzeptable Lösung gibt. Die Lösung einer solchen Problemstellung stellt deutlich höhere kognitive Anforderungen. Die Bewertung muss gegebenenfalls auch anders erfolgen, da nicht einfach simple Kategorien abgehakt werden können.

2.7 *Lernablauf*

Für die Struktur von PBL-Unterricht haben sich verschiedene Ausprägungen entwickelt (cf. [21] Hämäläinen S. 2). Ellis et al. unterscheiden deren drei:

1. „*Problem-based approach*“: die Lerninhalte werden in traditionellem Frontalunterricht vermittelt. Problemstellungen dienen dazu die Schüler zu motivieren und die Theorie zu veranschaulichen.
2. „*Guided problem-based learning*“: Probleme werden in Gruppen gelöst, jedoch werden die Grundkonzepte und schwierigsten Aspekte in klassischen Vorlesungen erläutert.
3. „*Full problem-based learning*“: Die Problemstellungen steuern die gesamte Lernerfahrung ohne Wissensunterstützung der Lehrkraft.

Es gibt verschiedene Vorstellungen von PBL und bis jetzt hat sich noch kein Konsens entwickelt. Laut O'Grady (cf. [22] S. 2) wird sich eine rigorose Definition von PBL vielleicht auch nie ergeben, da die Methodologie dem jeweiligen Lernbereich angepasst werden muss.

Nichtsdestotrotz lassen sich folgende Grundcharakteristiken von PBL festhalten (cf. [14] Reich S. 29, [21] Hämäläinen S. 2):

1. Der Lerner steht im Mittelpunkt und übernimmt soweit wie möglich die Verantwortung für sein eigenes Lernen.
2. Gelernt wird zumindest teilweise in kleinen Gruppen.
3. Die Rolle des Lehrers variiert je nach Ausprägung, tendiert jedoch eindeutig weg von der reinen Wissensvermittlungsrolle hin zum Lernbegleiter. D.h. die Lehrkraft versucht verstärkt, den einzelnen Schüler im Rahmen des Möglichen individuell bei der Entwicklung seines Lernprozesses zu unterstützen. Es geht ihr weniger darum, einfache Wissensinhalte zu vermitteln, als vielmehr die Entwicklung der zur Lösung der gegebenen Problemstellungen erforderlichen Kompetenzen zu fördern. Die Prozesse, die zur Wissensakquisition führen, stehen im Mittelpunkt, nicht deren Produkte.
4. Die Problemstellungen sind das motivierende Instrument, das die Entwicklung der Problemlösekompetenzen ermöglicht.
5. Kommunikation und soziale Kompetenzen im Allgemeinen spielen eine verstärkte Rolle.
6. Die Schüler sollen ihre Metakognition entwickeln, indem sie lernen, ihre eigenen Prozesse und Ergebnisse sowie die anderer Schüler objektiv zu evaluieren und zu kommentieren.

Die pädagogischen Ziele beinhalten die Entwicklung folgender Kompetenzen (cf. [14] Reich S. 29 - 31):

1. „*Sach-Kompetenz*“: Der Erwerb der im Lehrplan vorgegebenen Kompetenzen. Ein Großteil der Herausforderung von PBL für die Lehrkraft besteht in der Entwicklung von angemessenen Problemstellungen, die einen hohen Motivationsfaktor mit einer vollständigen Abdeckung der erforderlichen Kompetenzen kombinieren. Je nach Ausprägung des gewählten PBL-Ansatzes (siehe oben), werden Sachwissen und/oder Metakognition von der Lehrkraft ganz oder teilweise vermittelt oder von den Lernern ganz oder teilweise selbst erarbeitet. Im letzteren Fall müssen die erforderlichen Ressourcen (Bücher, Internetseiten) zur Verfügung stehen und die Schüler sind auf ihre „*Methoden- bzw. Medien-Kompetenz*“ angewiesen.
2. „*Methoden- bzw. Medien-Kompetenz*“: Die Fertigkeit, zu erkennen, welches Wissen unvollständig oder gar nicht vorhanden ist, sowie das gezielte und effiziente Finden der gesuchten Information, sind unabdingbare Voraussetzungen für die Entwicklung einer Lösung.
3. „*Sozial-Kompetenz*“: Bei komplexeren Problemstellungen werden die Vorteile des Ideen-, Prozess- und Wissensaustausches in der Gruppe offensichtlich und die Förderung der Sozial-

kompetenz ergibt sich fast von selbst.

4. „*Personal-Kompetenz*“: Die hohe Eigenaktivität, sowie der Austausch in der Gruppe, bietet dem Schüler zahlreiche Möglichkeiten, seine Persönlichkeit zu entwickeln. Er wird seinen eigenen Stil entdecken, um Probleme anzugehen und seine Ideen mitzuteilen. In der Informatik lässt sich dies unter anderem auch an der Problemlösung erkennen, z.B. anhand der gewählten Struktur, der Häufigkeit und dem Inhalt der Kommentare, Namensgebung, etc.
5. „*Problemlöse-/Handlungs-Kompetenz*“: Unter Einsatz der vorherigen vier Kompetenzen geht es hier darum, sich mit neuen Problemstellungen erfolgreich auseinandersetzen zu können, indem vorhandenes Wissen und Prozeduren kreativ und in einem neuen Kontext eingesetzt und konkrete Lösungen entwickelt werden.

Informatik ist eine Disziplin, die sich für PBL gut eignet, da sie selbst sehr stark problemgetrieben ist (cf. [19] Ellis et al. S. 42) und die meiste Arbeit im Team realisiert wird. Darüber hinaus wird die Informatik zunehmend über Fachbereichsgrenzen hinweg eingesetzt und die schnelle Entwicklung der Industrie bedingt lebenslanges Lernen. Das Ziel von PBL ist es, den Schülern zu helfen, die Kompetenzen für das Lösen von realen Problemstellungen sowie das lebenslange Lernen zu entwickeln.

PBL hat einige Gemeinsamkeiten mit projektorientiertem Lernen. Projekte fördern die Aktivierung der Lernenden und ermöglichen es, eine Verbindung zur beruflichen Realität herzustellen. Aus diesem Grund werden sie insbesondere auch in der Informatik oft verwendet. Problemstellungen unterscheiden sich von Projekten, da sie viel weniger, wenn nicht sogar keine, Struktur und Vorgaben bezüglich des Endergebnisses, und der Art und Weise, wie dieses zu erreichen ist, aufweisen. (cf. [24] Fee und Holland-Minkley S. 2) Somit fördern sie die Fähigkeiten zur Bestimmung sowie der Entwicklung möglicher Lösungen in verstärktem Masse und entsprechen damit eher der Realität des beruflichen Alltags in der Informatik (cf. [25] Savery S. 9).

Der idealisierte Lernablauf besteht aus folgenden 4 Teilen:

1) **Wahrnehmung und Analyse des Problems**
   - Wahrnehmung
   - Analyse
2) **Diskussion mit Mitgliedern der Lerngruppe**
   - Hypothesenbildung (Ideen / Annahmen)
   - Lernzielformulierung
3) **Selbststudium**
   - Informationsakquise / Erweiterung von Wissen und Fertigkeiten
4) **Ergebnisdiskussion der Lerngruppenmitglieder in Bezug auf das Problem mit offenem Ende**
   - Überprüfung und Modifikation der Hypothesen
   - Lösungsvorschlag als vorläufige Synthese

Abbildung 2.4: PBL-Lernablauf nach Barrows 2005 (cf. [14] Reich S. 2)

Hier kann man noch „*die differenzierte, möglichst standardisierte Evaluation, die für den vollen Erfolg von PBL notwendig ist*“ (cf. [14] Reich S. 2) als Teil 5 hinzufügen. Die Teile eins, zwei und vier finden in der Gruppe statt. Oft findet man den Lernablauf auch in sieben Schritte aufgeteilt, der sogenannte Siebensprung (cf. [26] Zumbach, Weber & Olsowski S. 24). Am eigentlichen Ablauf ändert dies jedoch nichts.

Schematisch lässt sich der Lernablauf als Prozessdiagramm folgendermaßen darstellen:

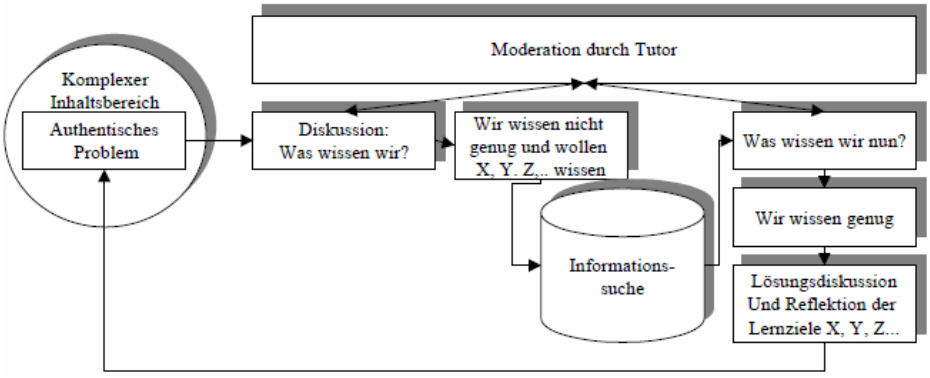


Abbildung 2.5: Der PBL-Prozess (cf. [27] Zumbach & Reimann S. 13)

Kompetenzen resultieren aus dem Einsatz individueller Kenntnisse (deklaratives Wissen), Fertigkeiten (prozedurales Wissen) und Haltungen in konkreten Situationen (cf. [26] Zumbach, Weber & Olsowski S. 22). Die konkrete Situation ist die Problemstellung, deren zentrale Rolle aus dem Diagramm erneut ersichtlich wird. Problemlösen ist ein fundamentaler Bestandteil der Informatik. Der Fachbereich hat sich jedoch bis heute mit dem Einsatz von PBL schwer getan, was dazu geführt hat,

dass es für Lehrer, die PBL erstmalig einsetzen wollen, nicht viele gute Problemstellungen gibt, die ihnen die praktische Umsetzung von PBL erleichtern würden (cf. [28] O'Kelly & Gibson S. 1). Stattdessen gibt es eine Vielzahl von Programmierproblemen, die sich auf das Erlernen eines Programmierkonzepts beschränken und somit die Problemlösekompetenz nicht wirklich fördern.

Selbstverständlich können die Kompetenzen im Verbund mit anderen Methoden vervollständigt und geübt werden (cf. [26] Zumbach, Weber & Olsowski S. 22).

2.8 *Evaluation*

Die Evaluation im PBL-Unterricht sollte den Zielen von PBL entsprechen, das heißt anstatt sich auf das Abfragen von Wissen zu beschränken, sollte die Problemlösekompetenz und damit implizit die Sach-, Methoden-, Sozial- und Personalkompetenz geprüft werden (cf. [29] Beaumont, Sackville & Cheng S. 10 – 11, [14] Reich S. 46).

Die formative Evaluation ist von fundamentaler Bedeutung, um dem Schüler und der Lehrkraft regelmäßig Rückmeldung über die erzielten Lernfortschritte zu geben. Die Evaluation durch andere Schüler kann eine konstruktive Ergänzung darstellen, die eine andere Perspektive bietet und nebenbei die Sozial-Kompetenz und Metakognition fördert.

Offene und ausreichend komplexe Problemstellungen in Kombination mit einem Bewertungsgespräch eignen sich für die summative Evaluation hervorragend, da sie Einblick in die Gedankengänge des Schülers geben und gleichzeitig Plagiate leicht erkennen lassen.

PBL ermöglicht jedoch auch die kontinuierliche Evaluation über den gesamten Modulverlauf, da die Lehrkraft in ihrer Tätigkeit als Tutor regelmäßig Kontakt mit den einzelnen Schülern hat und anhand der bearbeiteten Problemstellungen und den informellen Gesprächen den Entwicklungsstand eines Schülers beurteilen kann.

2.9 *Forschungsergebnisse zur Wirksamkeit von PBL*

PBL wird auf internationaler Ebene oft für die Berufsausbildung insbesondere im medizinischen Bereich als angemessene Ausbildungsstrategie angepriesen (cf. [30] Newman S. 5). Obwohl zahlreiche Praxisberichte und Datenanalysen zu PBL vorliegen, erlauben diese es, aufgrund ihres begrenzten Umfangs und/oder ihrer Qualität, aus Newman's Sicht nicht, eindeutige Schlussfolgerungen bezüglich der Wirksamkeit von PBL im Vergleich zu anderen Ansätzen zu ziehen. Hier besteht eindeutig Bedarf an einer umfassenden systematischen Untersuchung (cf. [30] Newman S. 7). Aller-

dings untersucht Newman in seiner Metaanalyse nur Studien, die sich auf die Anwendung von PBL in der Erwachsenenausbildung im Gesundheitsbereich beziehen. Aufgrund der sehr strengen Kriterien wurden von 91 Studien nur 12 zurückbehalten.

Laut Weber (cf. [26] Zumbach, Weber & Olsowski, S. 29) ergeben sich aus der Unterrichtsforschung über die Wirksamkeit von PBL folgende empirischen Ergebnisse:

1. „*Das PBL ist an den Lernenden orientiert, ist für Lernende interessant und herausfordernd, führt zu einer hohen Zufriedenheit bei den Lernenden (und Lehrenden), die Studienabbruchquote ist tief.*
2. *Die Motivation, die Akzeptanz, die Selbstlernfähigkeiten und die Verantwortung für das eigene Lernen der Studierenden sind beim PBL stärker entwickelt.*
3. *Kommunikative und soziale Fähigkeiten und Teamfähigkeit sind beim PBL tendenziell besser.*
4. *Die Analysefähigkeit, die Transferwirksamkeit, das nachhaltige Lernen und die Integration des Lernens sind beim PBL tendenziell überlegen.*
5. *Die Handlungskompetenz bzw. die Performanz der PBL-Studierenden ist deutlich besser.*
6. *Der Stand der Kenntnisse und des Wissens ist in etwa derselbe wie bei einer herkömmlichen Ausbildung, in einigen Fällen ist er beim PBL sogar besser.*
7. *Der Übersicht, der Systematik und den fachlichen Grundlagen muss beim PBL besondere Beachtung geschenkt werden, da in der Regel nicht mehr nach Fächern, sondern interdisziplinär gelernt wid.*
8. *PBL-Lernende erwerben nachweislich hohe Fähigkeiten im Umgang mit Quellen und mit ICT, beides wichtige Voraussetzungen für lebenslanges Lernen.*“

Die Autorin betrachtet eine gute Einführung und Begleitung der Lernenden als wichtige Erfolgsfaktoren.

O'Grady (cf. [22]) untersucht, unter anderem, die Wirksamkeit von PBL im universitären Informatikunterricht anhand von 63 veröffentlichten Studien aus den Jahren 1996 – 2011.

Abbildung 2.6 zeigt die Kategorisierung der untersuchten Studien. Die Mehrheit beschäftigt sich mit den innovativen pädagogischen Aktivitäten der Lehrkraft und den Reaktionen der Studenten. Im Informatikbereich besteht offensichtlich noch erheblicher Forschungsbedarf. Lediglich 37% der

Studien berichten von Versuchen, den PBL-Ansatz systematisch zu evaluieren.

Die Rückmeldungen sind tendenziell positiv, aber sehr unterschiedlich. Dies ist nicht verwunderlich, da es sich in den meisten Fällen um den ersten Kontakt der Studenten (und der Lehrkräfte) mit dem PBL-Ansatz handelte. Unter diesen Umständen würde man das Risiko negativer Rückmeldungen eher höher einschätzen (cf. [22] O'Grady S. 8).

	Category	Course Level	Organization Level	Society Level
1.	Goals & content	-	-	-
2.	Student(s)/community of students/citizens	-	-	-
3.	Teacher(s)/organization/society	-	-	-
4.	Student(s)/community of students/citizen - teachers/organization/society	-	-	-
5.1	Students: understanding of and attitude about goals and content	- (1)	-	-
5.2.	Students: actions of students	-	-	-
5.3.	Students: results of students' actions	2	-	-
6.	Teacher(s)/organization/society - goals and content	5	2	1
7.1.	Teacher's conceptions of student's understanding of goals/ content	6	-	-
7.2.	Teacher's conceptions of students' actions towards achieving goals	2	-	-
7.3.	Teacher's pedagogical activities	20(2)	-	-
8.	Student(s)/community of students/citizens - teacher(s)/organization(s)/society's pedagogical means to enhance learning	24(1)	1	-

*Abbildung 2.6: Studies Classified According to their Primary Didactic Focus and Secondary Focus (*cf. [22] *O'Grady S. 7)*

3 Dokumentierter Einsatz von PBL im Informatikunterricht

Praxisberichte und Studien zum Einsatz von PBL gibt es viele; diese beziehen sich jedoch vorwiegend auf Universitätsniveau, also erwachsene Studenten. Forschungsergebnisse zum Einsatz von PBL im Sekundarunterricht sind kaum vorhanden. Die einzige Studie, die mir bekannt ist, stammt von Wirkala und Kuhn (cf. [23]) und bezieht sich nicht auf Informatik, sondern auf Sozialstudien.

Im Informatikunterricht wurde PBL an einer Reihe von Universitäten eingesetzt. In diesem Kapitel fasse ich die in meinen Augen relevantesten Studien zusammen.

3.1 *Fee und Holland-Minkley*

Fee und Holland-Minkley (cf. [24] S. 1) berichten, dass sie PBL in der Abteilung „Information Technology Leadership“ am Washington & Jefferson College in Washington einsetzen, weil „*students improve their ability to analyze and solve complex computational problems when we pursue pedagogies that support them in developing these skills incrementally*“. Hierzu wird PBL für das gesamte Curriculum des Informatikstudiums eingesetzt.

Aus ihrer Sicht stellt PBL den Kontext für ihre unterschiedlichen Kurse dar, indem es die diversen Konzepte zur Lösung der gegebenen Problemstellungen zusammenbringt. Die Problemstellungen müssen insbesondere am Anfang, wenn die Lernenden ihre Problemlösekompetenz zu entwickeln beginnen, verständlich formuliert sein. Mit zunehmender Erfahrung sollen sie jedoch immer unstrukturierter werden, um die Weiterentwicklung der Studenten sicherzustellen.

Die Unterschiede im Wissen der einzelnen Studenten erschweren die Entwicklung einer adäquaten Problemstellung, welche einen erheblichen Zeitaufwand erfordert. Insbesondere bei Studenten mit wenig oder keiner Erfahrung mit dem PBL-Ansatz ist intensives Mentoring seitens der Lehrkraft erforderlich, um den Einstieg in den Umgang mit Problemen zu ermöglichen. Mit zunehmender Problemlösekompetenz der Schüler verringert sich der Bedarf an Unterstützung.

Die Textbuchvorgehensweise zur Umsetzung von PBL sieht so aus, dass die Lehrkraft zu Beginn kleine Probleme stellt, die dann mit der Zeit immer komplexer werden, während sie ihre Führung reduziert. Am Ende des Kurses sollen die Lernenden dann ihre Problemlösekompetenz soweit entwickelt haben, dass sie sowohl vorgegebene als auch selbst entwickelte Problemstellungen über ein breites Spektrum lösen können. Die praktische Umsetzung dieser idealen Vorgehensweise ist jedoch alles andere als einfach, wie wir in Kapitel 4 sehen werden.

Das Risiko der Überforderung der Lernenden ist sehr real, besonders bei denjenigen, die noch keine Erfahrung im Problemlösen haben. Das konstruktivistische Paradigma und PBL postulieren zwar, dass die Lernenden anhand der Problemstellung sich die benötigten Kompetenzen aneignen sollen, allerdings benötigt dies viel Zeit und vor allem eine feinfühlige Balance zwischen „*mental discomfort and uncertainty*“ und regelmäßigen Erfolgserlebnissen. Letztere sind unabdingbar um zu verhindern, dass der Lernende frustriert aufgibt. Dies bedingt, insbesondere am Anfang, intensive prozessuale Unterstützung durch die Lehrkraft.

Angesichts der Tatsache, dass das Lösen von Problemen das Fach Informatik wie einen roten Faden durchzieht, empfehlen die Autoren eine langfristige Perspektive in Bezug auf die Entwicklung der Problemlösekompetenz. Wenn das gesamte Curriculum im Rahmen von PBL abläuft, können die Studenten nach und nach lernen, sich mit anfangs kleinen Problemstellungen auseinanderzusetzen und Lösungsansätze zu erkunden. Nach der anfänglichen sehr starken Unterstützung durch die Lehrkräfte kann man somit mit der Zeit zu einem „*full problem-based learning*“ (cf. 2.7) übergehen und damit einen reibungslosen Übergang in die Berufspraxis ermöglichen.

PBL erfordert insbesondere zu Beginn einen erheblichen Einsatz seitens der Lehrkräfte, nicht nur bezüglich der Entwicklung der Problemstellungen, sondern auch aufgrund des erforderlichen individuellen Mentorings. Wenn die gesamte Abteilung diesen Unterrichtsansatz teilt und unterstützt, lässt er sich sehr viel effizienter organisieren und umsetzen.

Die Autoren implementieren PBL in unterschiedlichen Ausprägungen in Abhängigkeit des Kursniveaus. In Anfängerkursen werden einzelne Fertigkeiten mittels kleiner Übungsaufgaben erarbeitet und im Anschluss zur Lösung eines komplexeren Problems über einen Zeitraum von zweieinhalb Wochen kombiniert. Hierbei werden klare Vorgaben gemacht, wie das Problem in kleinere Teile zerlegt werden kann, welche in Reichweite der Studenten sind. Des Weiteren wird eine Beschreibung einer korrekten Lösung gegeben.

Auf dem nächsthöheren Kursniveau wird von den Studenten erwartet, dass sie neu eingeführte Fertigkeiten eigenständig üben. Die Problemstellungen sind anspruchsvoller, jedoch weiterhin mit spezifischen Vorgaben versehen. Gearbeitet wird in Teams von zwei bis drei Personen und das Problem wird während sechs Wochen, im Anschluss an herkömmliche kleinere Aufgabenstellungen, bearbeitet.

Auf der zweithöchsten Kursebene sind die Problemstellungen offener und ohne formale Vorgaben. Jedoch sorgt die Lehrkraft mittels Rückmeldung dafür, dass die Lösungsansätze nicht aus dem Ru-

der laufen. Auch hier wird das Problem während sechs Wochen in einer Gruppe von drei bis vier Personen, im Anschluss an kleinere Aufgaben, bearbeitet.

Auf dem höchsten Kursniveau werden realistische Problemstellungen ohne Vorgaben bezüglich des Lösungsansatzes verwendet. Das Problem wird über das gesamte Semester hinweg in Gruppen von drei bis vier Mitgliedern bearbeitet. Die Studenten bringen hier die während ihres Studiums entwickelten Kompetenzen zum Einsatz, insbesondere auch ihre Fähigkeit zur kritischen Analyse und zum eigenständigem Denken. Die Lehrkräfte unterstützen den Übergang in die Berufswelt, indem sie die Studenten auf die Bedeutung von Geld, Organisation, Kundenprioritäten usw. aufmerksam machen.

Die Autoren berichten, dass die 52 Studenten, die innerhalb von 5 Jahren das höchste Kursniveau durchlaufen haben, dieses auch erfolgreich abgeschlossen haben. Von 8 berufstätigen ehemaligen Studenten wurde Rückmeldung erhalten, die sehr positiv ausfiel und die Erwartungen der Autoren in den PBL-Ansatz bestätigten.

3.2 *Hämäläinen*

Hämäläinen (cf. [21]) berichtet über ihre Erfahrungen mit dem Einsatz von PBL im Unterrichten des 10-wöchigen Kurses „Theoretical Foundations of Computer Science“ im Jahr 2003 an der Universität von Joensuu in Finnland.

Der Kurs war auf den traditionellen sieben Schritten, die aus der Anwendung von PBL in der Medizin bekannt sind, aufgebaut. Diese bestehen aus der Bestimmung unklarer Konzepte, Definition des Problems, Brain storming, Hypothesenbildung, Bestimmen der Lernziele, Selbststudium und schließlich Diskussion der Ergebnisse innerhalb der Gruppe. Diese Schritte mussten mittels eines Berichts dokumentiert werden. Zusätzlich wurden die neuen Lerninhalte mittels traditioneller Aufgaben geübt und die Studenten führten ein Lerntagebuch, in dem sie einen Gesamtüberblick über das, was sie gelernt haben, zu geben versuchten und das gleichzeitig zur eigenen Lernüberwachung diente.

Die Autorin erzielte mit diesem Ansatz hervorragende Resultate. In diesem Fach, das wegen seiner sehr theoretischen Natur als sehr schwer und langweilig betrachtet wird und das von der Mehrheit erfolglos abgebrochen wird, arbeiteten die Studenten engagiert und die Abbruchquote war sehr gering. Die Noten und Qualität der Lerntagebücher lassen auf ein sehr tiefes Verständnis der Lerninhalte schließen. Außerdem bereitete der Unterricht allen Beteiligten Freude und unterstützte die un-

terschiedlichen Lerntypen.

Es werden Metaanalysen erwähnt, die darauf hindeuten, dass die Absolventen von PBL Kursen unmittelbar nach dem Kurs besser entwickelte Kompetenzen, jedoch ein etwas schlechteres Wissen haben im Vergleich zu Absolventen eines herkömmlichen Kurses. Zu einem späteren Zeitpunkt ist das theoretische Wissen der ehemaligen PBL-Studenten jedoch mindestens genauso gut, was darauf hindeutet, dass sie das Gelernte besser behalten.

Der Bericht geht jedoch auch auf Probleme ein, die mit dem PBL Ansatz in der Informatik auftreten können. Es wird empfohlen, realistische Problemstellungen zu verwenden. Dies geht auch aus den theoretischen Grundlagen von PBL hervor (cf. 2). In der Praxis sind solche Problemstellungen jedoch oft zu aufwendig und schwierig. Um dem entgegen zu wirken, greift die Autorin auf die Empfehlung von Ellis et al. zurück, Lernende mit keiner PBL Erfahrung an gut strukturierten Problemen arbeiten zu lassen. Für erfahrene PBL-Lerner hingegen sollten die Problemstellungen offener und weniger strukturiert, d.h. realistischer, sein.

Für andere potentielle Probleme, wie die Unfähigkeit der Studenten, sich geeignete Lernziele zu setzen oder ihre Metakognition zu entwickeln, sowie das Risiko, dass sie vom PBL-Ansatz abweichen, schafft das Lerntagebuch effektive Abhilfe, da die Studenten hier ihren eigenen Lernprozess analysieren und es darüber hinaus als Kommunikationsinstrument zwischen Lehrkraft und Student dient.

Der praktische Kursverlauf beinhaltete zwei Stunden klassischer Aufgaben pro Woche. Die Vorlesungszeit war zweigeteilt, wobei die eine Hälfte zur Problembearbeitung und die andere für Vorlesungen diente.

Für jedes behandelte Problem erstellten die Studenten einen Bericht. Die Abschlussbewertung setzte sich aus der Bewertung der Berichte, des Lerntagebuchs und der Aufgaben zusammen.

Die Studenten konnten wählen, ob sie am PBL-Teil teilnehmen oder aber den Kurs mittels Aufgaben und Prüfungen absolvieren wollten.

Es wurden 14 Problemstellungen bearbeitet, die Studenten wurden jedoch nicht über die offiziellen Lernziele informiert, sondern sollten ihre eigenen Lernziele definieren. Am Kursende erlaubte dies den Studenten ihre eigenen Lernziele mit den offiziellen zu vergleichen und sich somit selbst zu evaluieren.

In der folgenden Tabelle (cf. [21] S. 4) fasst die Autorin die Ergebnisse des Experiments zusammen:

	PBL	TRAD
registered	63	14
female	23	4
male	40	10
participated	63 (61)	12 (14)
female	23	4
male	40 (38)	8 (10)
dropped	5	7
female	-	2
male	5	5
failed	1	-
female	1	-
male	-	-
passed	55	7
female	22	2
male	33	5

Es fällt auf, dass die Abbruchquote beim PBL-Teil nur 7% beträgt, gegenüber 50% beim herkömmlichen Ansatz. Die Erfolgsquote beim PBL-Teil lag bei 90% und 50% beim Rest. Hinzu kommt, dass diejenigen die den PBL-Teil abbrachen dies in der ersten Woche taten und somit nicht Zeit bis zur ersten Prüfung verschwendeten, wie dies beim traditionellen Ansatz oft der Fall ist.

Die Rückmeldungen der Studenten mittels Fragebogen, die nur von der knappen Hälfte ausgefüllt wurden, waren sehr unterschiedlich. 26 Studenten schrieben freie Kommentare zu PBL, die mehrheitlich positiv ausfielen. Die Rückmeldungen in den Lerntagebüchern waren ebenfalls positiv.

3.3 *Ellis et al.*

Die Autoren (cf. [19]) illustrieren anhand von 2 Fallstudien den Einsatz von PBL im Informatikunterricht auf Universitätsniveau.

3.3.1 Fallstudie 1

Die erste Fallstudie basiert auf zum Teil einsatzerprobten Ideen, zum Einsatz von PBL bei der Einführung in die Programmierung, um die Lernkompetenzen der Studenten zu entwickeln. Die Problemerstellung beginnt im mittleren Bereich des in 2.7 besprochenen Spektrums (cf. Abbildung 2.2), d.h. die Lehrkraft gibt eine relativ offene Problemstellung vor, bewegt sich jedoch im Verlauf der vier Aktivitäten zusehends nach links, d.h. die Studenten entwickeln die Problemstellung selbst.

Die erste Aktivität dauert nur eine Unterrichtseinheit und findet in Gruppen von vier bis fünf Studenten statt. Es geht hier darum, erste Erfahrungen in der gemeinsamen Entwicklung einer Lösung

für eine offene, d.h. nicht präzise spezifizierten, Problemstellung zu sammeln. Das gestellte Problem besteht darin, die Schritte die zum Binden der Schnürsenkel erforderlich sind, aufzuschreiben. Hierfür steht die halbe Unterrichtseinheit zur Verfügung. In der zweiten Hälfte präsentiert eine Gruppe ihren Lösungsvorschlag, der von den anderen kommentiert wird.

Die zweite Aktivität geht über zwei Unterrichtseinheiten und ähnelt der ersten, jedoch entwickeln die Gruppen ihre Problemstellung selbst. Anschließend entwickeln sie einen Lösungsvorschlag für ihr Problem. Hierbei spielt die Zeiteinteilung eine wichtige Rolle. Hier sollte die Lehrkraft unterstützend eingreifen indem sie z.B. vorgibt, bis wann das Problem bestimmt sein muss. Auch die Aufteilung des Gesamtproblems in kleinere Probleme ist wichtig und sollte in der Klasse besprochen werden. Das Ergebnis sollte elektronisch veröffentlicht oder mündlich vorgetragen werden.

Die dritte Aktivität dauert eine Unterrichtseinheit und dient dazu, die Studenten für die Einführung von Schleifen und Feldern zu motivieren. Hierzu sollen sie Beispiele nennen und beschreiben, wo Programme wiederholt die gleichen Operationen auf Datensätzen durchführen. Nachdem die Studenten ihre Ergebnisse vorgestellt haben folgt eine kurze Vorlesung zur Schleifenverwendung im Allgemeinen sowie in der unterrichteten Programmiersprache.

Die letzte Aktivität dauert zwei Wochen und findet in Gruppen von jeweils 8 Studenten statt. Das Problem wird vorgegeben. Die Autoren geben als Beispiel die Untersuchung der Datenanwendung in medizinischen Anwendungen. Die einzelnen Gruppen können unterschiedliche Rollen übernehmen um den Datenbedarf zu analysieren. Die Studenten präsentieren ihre Arbeiten vor der Klasse. Aus der anschließenden Diskussion können sich Minivorträge über Datentypen, Datenbanken usw. ergeben.

3.3.2 Fallstudie 2

Hier wird der Einsatz von PBL in der Mitte eines zweijährigen Einführungskurses in die objektorientierte Programmierung mittels Java und in die Konzepte der Softwareentwicklung beschrieben. Das Ziel ist die Bearbeitung eines anspruchsvolleren Problems, wobei der gesamte Softwarelebenszyklus, von der Bestimmung der Anforderungen über das Design, die Implementierung und die Testphase, durchlaufen wird.

Die Studenten arbeiten in Zweiergruppen während 4 Wochen und durchlaufen die folgenden 7 Schritte:

1. Das Problem, z.B. die Entwicklung eines spezifischen Datenverwaltungssystems, wird vor-

gestellt. Um die Motivation zu erhöhen wird behauptet, dass die zu entwickelnde Software vom Abteilungssekretariat benötigt wird.

2. Das Problem wird in Gruppen von jeweils 24 Studenten besprochen um die Anforderungen zu bestimmen.
3. Ein „reales“ Anforderungsanalysedokument wird von den Lehrkräften präsentiert. Allerdings wurden zwei Kapitel (das Benutzerhandbuch sowie die Testvorhersage) entfernt, die von den Studenten selbst erstellt werden müssen.
4. Das Designdokument wird in der Klasse besprochen. Mittels Rollenspielen werden Objekte visualisiert.
5. Die noch fehlenden Informationen zur Implementierung der 12-15 Java Klassen werden unter der Leitung der Lehrkraft in der Klasse besprochen.
6. Die Klassen werden programmiert, dokumentiert und getestet.
7. Die produzierten Dokumente und die Präsentation des Projekts jeder Zweiergruppe werden bewertet. Zusätzlich wird ein individuelles Gespräch geführt.

Über das Intranet werden über 1100 HTML-Seiten (davon 150 Seiten aus eigener Produktion) an Nachschlagematerial zur Verfügung gestellt. Diese reichen von Beispieldokumenten zur Softwarequalitätsplanung bis zu komplett dokumentierten und getesteten Java-Anwendungen.

3.4 *Nuutila, Törmä und Malmi*

Die Autoren (cf. [31]) der technischen Universität in Helsinki berichten über ihre Erfahrungen, aus den Jahren 1999 bis 2003, mit PBL in einem Einführungskurs in die objektorientierte Programmierung mittels Java der im ersten Studienjahr angeboten wird.

Die traditionellen Vorlesungen werden durch zehn Problemfälle ersetzt. Pro Woche wird etwa ein Problem behandelt. Hierzu werden die Studenten in Gruppen von sieben bis zehn Personen aufgeteilt. Jeder Gruppe steht ein Tutor zur Seite. Beim Tutor handelt es sich entweder um eine Lehrkraft oder einen Studenten, der diesen Kurs bereits absolviert hat. Er übernimmt die Rolle des Fachexperten und des Gruppenmoderators.

Opening session – half an hour, in the group
Step 1: Examination of the case. The group gets familiar with the case material. Step 2: Identification of the problem. An initial title for the case is specified. Step 3: Brainstorming. The students present their associations and ideas about the problem to find out what is already known and how does the case relate to the previous knowledge. The ideas are said aloud and written on self-stick notes, which are organized on a white board. Step 4: Sketching of an explanatory model. An initial version of the explanation for the problem is constructed and most important concepts and their relations are identified. Step 5: Establishing the learning goals. Those parts of the explanatory model that are mysterious, fuzzy, or simply unknown are identified and the central ones are chosen as learning goals for the group.
Study period -- one week, each student working independently
Step 6: Independent studying. Each student independently studies to accomplish *all* learning goals. This phase includes information gathering and usually a substantial amount of reading (e.g., 50--150 pages).
Closing session -- one to two hours, in the group
Step 7: Discussion about learned material. Equipped with the newly acquired knowledge, the group reconvenes to discuss the case. The discussion includes *explanation* of central concepts and mechanisms, *analysis* of the material, and *evaluation* of its validity and importance.

Abbildung 3.1: Die sieben Schritte der Problembehandlung (cf. [30] Nuutila, Törmä & Malmi S. 5)

Jede Gruppe trifft sich wöchentlich für drei Stunden zu einer PBL-Sitzung. Zu Beginn wird die vorherige Problemstellung abgeschlossen. Dann wird ein neues Problem bearbeitet, indem die folgenden sieben Schritte durchlaufen werden: Die Autoren berichten, dass der dritte Schritt, bei dem Kritik an den Ideen anderer Studenten untersagt ist, gewöhnlich die wenigsten Probleme bereitet. Der vierte Schritt bereitet jedoch wesentlich mehr Schwierigkeiten. Hier beschränkt sich die Aktivität der Schüler meist auf eine Neuordnung der Notizen. Konzeptkarten, auf denen ein Lösungsmodell entworfen wird, helfen, neue Abstraktionen und Zusammenhänge zutage zu fördern.

Der sechste Schritt ist der wichtigste und sollte von Anfang an betont werden. Ungenügender Einsatz seitens der Studenten beim Selbststudium wird unweigerlich im letzten Schritt ersichtlich und führt zum Scheitern des PBL-Ansatzes.

Die Autoren unterscheiden drei Kategorien von Problemstellungen:

1. Wissensorientierte Probleme konfrontieren den Lerner mit etwas, das er mit seinem aktuellen Wissen nicht erklären oder lösen kann. Dies erzeugt die intrinsische Motivation die Ursachen zu erforschen und zu verstehen. In der Programmierung eignen sich diese Probleme gut zur Erarbeitung von Grundwissen und Konzepten. Ein Beispiel wäre die Simulation eines Countdowns für einen Raketenstart um das Konzept einer Schleife zu lernen.
2. Designprobleme stellen komplexere Problemstellungen dar, mit anspruchsvolleren Lernzielen als bei den wissensorientierten Problemen. Sie dienen dazu die Problemlösekompetenz und das Abstraktionsvermögen zu entwickeln.
3. Analyseprobleme dienen dazu die Unvollständigkeit respektive Fehlerhaftigkeit vorhandener Denkmodelle aufzuzeigen und diese durch bessere Modelle zu ersetzen. Hierzu kann man z.B. ein scheinbar völlig korrektes Programm vorführen, das jedoch eine falsche Ausgabe liefert. Durch das Erforschen der Ursache für die fehlerhafte Ausgabe passt der Lernende sein eigenes Denkmodell an. Diese Art der Problemstellung sollte, nach den Erfahrungen der Autoren, nicht in der Gruppe sondern individuell bearbeitet werden. Der Grund besteht darin, dass in einer Gruppe oft eine Person den Fehler sofort erkennt, da sie nicht die gleiche Fehlvorstellung wie die anderen Gruppenmitglieder hat. Durch die sofortige Bekanntgabe des Fehlers haben die anderen nicht die Chance, durch ihre eigene Auseinandersetzung mit dem Problem die Fehlerhaftigkeit ihres Denkmodells zu erkennen und zu korrigieren.

Die einmalige Erarbeitung eines Konzepts mittels eines Problems reicht nicht aus, da hier die Wiederholungskomponente fehlt. Daher bearbeiten die Studenten neben den Problemfällen auch Programmieraufgaben. Diese werden individuell gelöst und ermöglichen es das Gelernte wiederholt anzuwenden und somit zu festigen. Als Unterstützung bei der Lösung der Programmieraufgaben finden wöchentliche Treffen mit den Kursassistenten statt. Gegen Ende des Kurses wird ein individuelles Programmierprojekt realisiert.

Eine Studentenbefragung hat ergeben, dass die Programmieraufgaben durchschnittlich fünf Stunden pro Woche beanspruchen, während die Studenten nur gut zwei Stunden aufbringen um sich mit den Problemfällen zu beschäftigen. Hier stellt sich die Frage der richtigen Balance.

Die Studenten schreiben auch Aufsätze und zeichnen Konzeptkarten für die wichtigsten Mechanismen der Programmiersprache. Darauf aufbauend entwickeln sie ihr Portfolio, in dem sie ihre Lernerfahrungen zusammenfassen und analysieren. Die Aufsätze dienen dazu, das Lernen abstrakter Konzepte zu erleichtern, insbesondere in den Fällen wo es schwierig ist, sinnvolle Problemfälle zu erstellen. Die Konzeptkarten wurden eingeführt, um die Studenten stärker auf die Konzepte und deren Zusammenhänge zu fokussieren. Bei den Aufsätzen besteht die Gefahr, dass jemand sein mangelndes konzeptuelles Verständnis hinter vielen Worten versteckt. Die Qualität der erstellten Konzeptkarten hat die Erwartungen der Autoren übertroffen. Letztere lassen sich zudem nicht so leicht aus dem Internet besorgen und lassen Verständnisprobleme leichter erkennen als Aufsätze. Die Evaluation basiert auf den Programmieraufgaben, dem Programmierprojekt, der Prüfung und dem Portfolio.

Die Autoren experimentierten wie ich (cf. 4.1.4.2) mit PBL-Berichten um die Metakognition zu fördern. Die Studenten sollten ihren eigenen Lernprozess analysieren und weiterentwickeln. Bis auf wenige Ausnahmen erwiesen diese Berichte sich jedoch als sehr oberflächlich, sodass der Zeitaufwand sie zu erstellen in keinem vertretbaren Verhältnis zum Ergebnis standen. Folglich wurden die Berichte aus dem Kursablauf entfernt.

Um die Studenten beim Einstieg in die Programmierung und insbesondere die Verwendung der Entwicklungswerkzeuge zu unterstützen wurden Programmiervorführungen sowie überwachte Laborzeiten eingeführt, welche sich großer Beliebtheit erfreuten.

Die Autoren betonen die Wichtigkeit der Reihenfolge in der die verschiedenen Aspekte des Lerngebiets eingeführt werden. Die Studenten sollten jeden Problemfall mit vorherigem Wissen in Verbindung bringen können, ansonsten entsteht Verwirrung und Demotivation.

Die Studentenbefragung hat ergeben, dass Gruppenarbeit die Motivation erhöht und emotionale Unterstützung sowie einen sozialen Kontext bietet. Die gesamte Bewertung basiert jedoch auf individueller Basis. Hierbei ist es wichtig, dass der Tutor der Gruppe klare Rückmeldung über ihre Lernerfolge gibt und sie darauf aufmerksam macht, falls sie nicht genügend Zeit ins eigene Studium investieren.

Das Erlernen der Programmierung erfordert das Aneignen von einer großen Anzahl von abstrakten Konzepten. Darüber hinaus stellen das Lösen von komplexen Problemen und das Design von Lösungen (deren es oft viele gibt, die sich jedoch qualitativ stark unterscheiden), welche eigentlich die Hauptaktivitäten bei der Softwareentwicklung darstellen, kognitive Aktivitäten auf recht hohem Ni-

veau dar. Hinzu kommen die strikte Syntax und Semantik von Programmiersprachen, sowie die Beherrschung der Entwicklungsinstrumente wie Editoren, Compiler, usw.

Gut durchdachte Problemstellungen eignen sich sowohl für das Lernen von Konzepten, Syntax und Semantik als auch für die Entwicklung der Problemlöse- und Designkompetenzen. Sie veranlassen den Lerner sich Fragen bezüglich der Konzepte zu stellen, was ihn zum Selbststudium ermutigt. Die abschließende Gruppenbesprechung kann, durch die Auseinandersetzung mit unterschiedlichen Sichtweisen, zum besseren Verständnis beitragen. Dies gilt auch für die Aufsätze und insbesondere die Konzeptkarten. Das Beherrschen der Konzepte erfordert jedoch neben der Problembehandlung intensive Übungspraxis. Zur Erarbeitung der Syntax und Semantik einer Programmiersprache können Probleme gestellt werden, wo die Studenten den Quellcode eines funktionsfähigen Programms sowie das Ergebnis einer Ausführung erhalten und die Funktionsweise des Programms herausfinden sollen. Die Verwendung der Programmierwerkzeuge und die Entwicklung der notwendigen Programmierroutine lässt sich hingegen nicht mit Problemstellungen, sondern mit Vorführungen und Trainingsaufgaben erlernen.

PBL bietet weitere Vorteile indem es die Sozialkompetenz, das kreative Denken, die Arbeitsplanung sowie die Kommunikations- und Forschungskompetenz fördert. Die Autoren haben auch beobachtet, dass die erhöhte Eigeninitiative, die PBL entwickelt, in die nachfolgenden Kurse mitgenommen wird.

Es fällt auf, dass die Abbruchquote bei diesem Kurs nur bei 17% lag, gegenüber 45% beim herkömmlichen Unterrichtsansatz.

In der subjektiven Einschätzung der Autoren lernen die Studenten viel in diesem Kurs. Dies zeigt sich an der Qualität ihrer Projekte und wie sie ihre Implementierungen erklären können. Es hat sich auch gezeigt, dass die Noten im folgenden Javakurs für Fortgeschrittene bei den PBL-Absolventen etwas höher liegen.

3.5 *Bunch*

Der Autor (cf. [32]) beschreibt das Curriculum und die Unterrichtsmethode für ein Bachelor-Studium in Webanwendungsentwicklung auf postsekundärer Ebene.

Im Gegensatz zum herkömmlichen Ansatz, wo ein Thema eingeführt und abgeschlossen wird ehe das nächste Thema in Angriff genommen wird, verwendet der Autor einen sogenannten Spiralablauf, d.h. der Lerner beherrscht ein Thema teilweise um dann das nächste Thema teilweise zu erar-

beiten bis alle Themen teilweise erschlossen sind ehe er zum ersten Thema zurück kehrt um es zu vertiefen, usw. Dies ermöglicht die Entwicklung von Webanwendungen mit steigender Komplexität wobei ein immer größerer Teil der Lernziele abgedeckt wird. In der Literatur zur Unterrichtspsychologie finden sich Hinweise, dass ein spiralförmig organisiertes Curriculum wesentlich besser mit den psychologischen Prozessen der Schema-Entwicklung und der Bildung von mentalen Modellen übereinstimmt als der traditionelle modulare Ansatz.

Jeder Kurs besteht aus einer Sequenz von realistischen „goal-based scenarios“ (GBS), die der Student mittels der Entwicklung einer Webanwendung löst. Diese Szenarien beginnen recht einfach und werden dann schrittweise immer anspruchsvoller, sodass alle Themenbereiche nach und nach immer weiter vertieft werden. Dies ermöglicht es dem Studenten kontinuierlich die Entwicklung seiner berufsrelevanten Kompetenzen zu beobachten und somit die für ihn sehr wichtige Frage „Wozu brauche ich dies?“ überzeugend zu beantworten.

Dem potentiellen Problem der kognitiven Überforderung der Studenten aufgrund der Problemkomplexität begegnet der Autor mittels scaffolding, d.h. maßgeschneiderter Unterstützung. So werden beinahe fertiggestellte Beispiele verwendet, in denen der Lernende zunächst nur kleine Teile ergänzen muss. Anschließend werden die zu ergänzenden Teile immer größer. Mit diesem Vorgehen haben zahlreiche Autoren positive Ergebnisse erzielt.

Die praktischen Kurse mit den Titeln „Introduction to Server Side Programming“, „Intermediate Server Side Programming“ und „Advanced Server Side Programming“ finden während der ersten drei Semester statt und bestehen jeweils aus einer Reihe von Problemstellungen, die eine stetig steigende Beherrschung der Lernbereiche erfordern.

Die Lehrkraft führt die komplette Lösung einer Problemstellung vor und erklärt das erforderliche Wissen und die benötigten Kompetenzen. Im Anschluss erhalten die Studenten die Musterlösung und müssen dann eine andere Problemstellung vom gleichen Schwierigkeitsgrad selbst lösen.

Die nächste Problemstellung ist dann etwas anspruchsvoller. Dieser Prozess wird fortgesetzt, bis alle Themenbereiche bis zur gewünschten Tiefe erarbeitet wurden. Hierdurch entsteht eine Spirale durch die Lernbereiche. Abbildung 3.2 stellt diesen Verlauf grafisch dar.

Jeder praktische Kurs wird von einem Vertiefungskurs begleitet, wo Problemstellungen mit einem schmaleren Themenfokus behandelt werden., z.B. Grundlagen der Programmierung oder Computernetzwerke. Die Verteilung der Vertiefungskurse über das Curriculum ist aus Abbildung 3.3 ersichtlich.

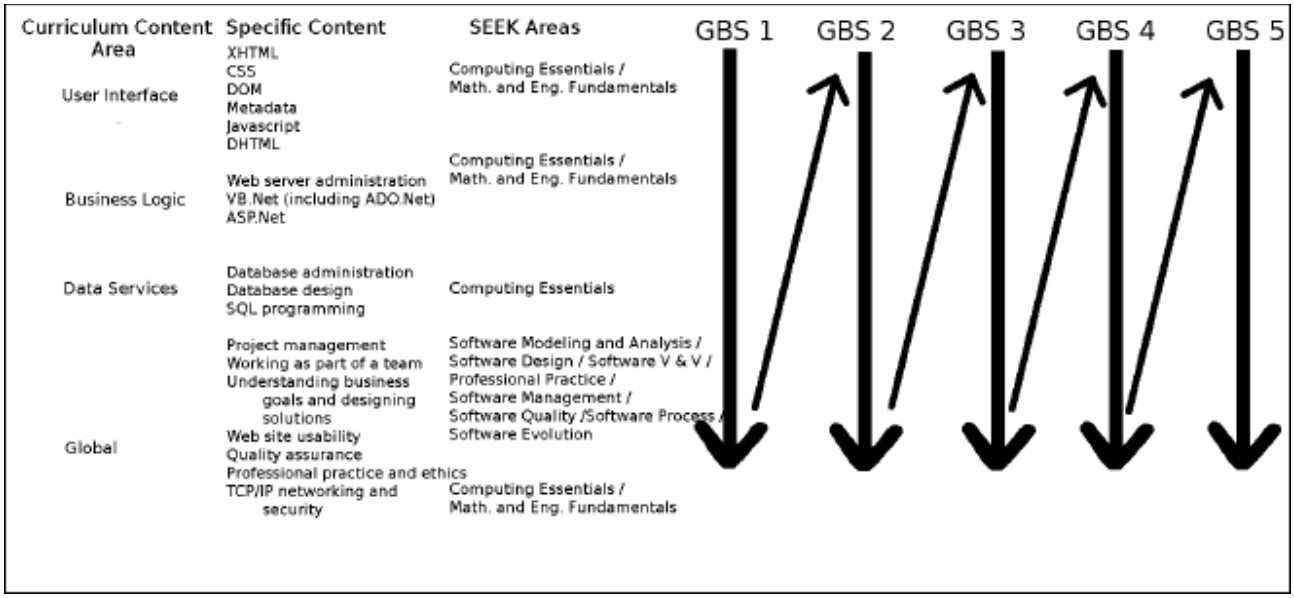


Abbildung 3.2: Relative emphasis and distribution of content areas across curriculum (cf. [32] Bunch S. 9)

		Quarter	1	1	2	2	3	3	4	5
		Course	Programming Concepts	Introduction to Server-Side Programming	Computer Networking	Intermediate Server-Side Programming	Database Administration and Programming	Advanced Server-Side Programming	Project Management	Special Topics in Web Development
Curriculum Content Area	**Specific Content**	**SEEK Knowledge Areas**								
User Interface	XHTML	Computing Essentials / Math. and Eng. Fundamentals	2	2	1	1		1	3	5
	CSS		2	2	1	1		1		5
	DOM		2	2	1	1		1		5
	Metadata		2	2	1	1	1	1		5
	Javascript		5	1	1	3		3		5
	DHTML		2	3	1	2		2		5
Business Logic										
	Web server administration	Computing Essentials / Math. and Eng. Fundamentals			10	9	5	3		7
	VB.Net (including ADO.Net)		35	30	10	30	10	30	5	8
	ASP.Net		30	35	10	30	5	34		8
Data Services										
	Database administration	Computing Essentials	1	1	5	5	15	2		8
	Database design		5	5	2	5	25	5	5	8
	SQL programming		5	4	3	5	30	10		8
Global										
	Project management	Software Modeling and Analysis / Software Design / Software V & V / Professional Practice / Software Management / Software Quality / Software Process / Software Evolution	2	2		1	1	1	25	3
	Working as part of a team			2		1		1	20	3
	Understanding business goals and designing solutions		2	2	3	1	2	1	25	5
	Web site usability			2		1		1	2	5
	Quality assurance			2		1	1	1	5	4
	Professional practice/ethics		1	1	1	1	1	1	5	1
	TCP/IP networking and security	Computing Essentials / Math. and Eng. Fundamentals	4	2	50	1	4	1	5	2

Key: Numbers in cells are approximate percentage; the darker the cell, the greater the relative emphasis.	0%	5%	10%	15%	20%	25%	30%	35%	50%

Abbildung 3.3: Distribution of specific content areas across quarters and courses (cf. [32] Bunch S. 10)

Die Klassengröße für jeden Kurs ist auf 20 Studenten begrenzt. Die Szenarien werden abwechselnd in der Klasse und zuhause bearbeitet. Die Lehrkraft überwacht die Lösung der ersteren und gibt individuelle Hilfestellung falls nötig. Die zuhause zu bearbeitenden Probleme werden mit einer Note bewertet. In beiden Fällen kann ein Student die nächste Problemstellung erst beginnen wenn er die vorherige vollständig gelöst hat. Am Ende der Problemsequenz steht eine Problemstellung deren Lösung den Einsatz aller Lernziele verlangt. Nur wer einen Kurs bestanden hat darf am nächsten teilnehmen.

Die Evaluation hat drei wichtige Funktionen. Erstens dient sie dazu sicherzustellen, dass der Student das nötige Wissen und die erforderlichen Kompetenzen hat um das nächste Problem zu lösen. Zweitens ermöglicht sie es der Lehrkraft Bereiche zu entdecken, in denen der Student gezielte Unterstützung benötigt. Drittens ist es sehr wichtig, dass der Student so schnell wie möglich Rückmeldung bezüglich der Problemlösung bekommt, damit er die Information mit dem Problem assoziiert speichern und somit später besser wiederverwenden kann.

Neben der Studentenbewertung ist es jedoch auch wichtig, die Wirksamkeit des Unterrichtsprogramms zu bewerten um gegebenenfalls Änderungen vorzunehmen. Für jede Problemstellung geben die Studenten mittels einer einfachen Ratingskala und einem beliebigen Kommentar Rückmeldung. Dies ermöglicht es die einzelnen Szenarien anzupassen. Die Lösequote des Abschlussproblems jedes Kurses gibt Auskunft darüber, inwiefern die Lernziele erreicht wurden. Zusätzlich verwendet die Lehrkraft eine Checkliste für alle Problemstellungen die zuhause bearbeitet wurden. Für jeden Studenten wird über die nicht erreichten Lernziele Buch geführt. Sollte sich hier ein Muster abzeichnen werden die entsprechenden Szenarien überdacht. Weitere Einblicke in die Effektivität des Unterrichts lassen sich aus den Praktikumsberichten der Studenten gewinnen. Die ultimative Rückmeldung stellt die Einstellungsquote nach dem Studium dar. Dazu werden die Studenten drei, sechs und zwölf Monate nach Abschluss des Studiums nach ihrem aktuellen Arbeitsstatus gefragt.

4 Anwendung im technischen Sekundarunterricht

4.1 *Erstes Semester*

4.1.1 Kurswebseite

Die Kurswebseite http://foxi.ltam.lu/COURS/cliss/ besteht aus fünf Teilen plus einem Einstellungsdialog. Letzterer dient dazu, die Hintergrundfarbe sowie die Schriftgrößen für das Tutorial und das Training einzustellen. Diese Einstellungen werden lokal im Browser gespeichert.

4.1.2 Bibliothek

Dieser Bereich dient dazu, der Klasse die nützlichsten Ressourcen für ihre Recherchen zentral zugänglich zu machen. Dazu gehören die wichtigsten Internetseiten, sowie eine Reihe von Büchern zum Thema JavaScript, die ich der Klasse am ersten Schultag mit gebracht hatte.

Ich hatte nicht den Eindruck, dass dieser Bereich von vielen Schülern benutzt wurde. Ein Schüler hat sich zwei Bücher aus der Liste gekauft und sich sehr positiv entwickelt (cf. 4.2.3.2).

4.1.3 Motivierende Problemstellung

Wie in 2.4 beschrieben ist die Wahl der Problemstellung von entscheidender Bedeutung. Da ich aus meiner eigenen Erfahrung sowie der Beobachtung meiner Klassen weiß, dass Jugendliche hoch motiviert sind, wenn es um Computerspiele geht, war mir von Anfang an klar, dass das Hauptproblem die Programmierung eines Spiels, in diesem Fall des klassischen Space Invaders, sein sollte. Dies bringt den Vorteil mit sich, dass alle Lernziele des Unterrichtsprogramms mit einem Problem abgedeckt werden können.

4.1.4 Tutorial

Da die Klasse keine Erfahrung mit PBL und nur sehr begrenzt (9e Scratchkurs) mit Programmierung hatte, konnte ich sie nicht einfach mit dem Hauptproblem konfrontieren, in der Hoffnung, dass sie das schon irgendwie hinkriegen würde. Daher habe ich ein Tutorial entwickelt, um mittels *guided problem-based learning* (cf. [21] Hämäläinen S. 2) den Einstieg in PBL zu erleichtern.

4.1.4.1 *Einführung*

Im Einführungsabschnitt wird erläutert, was man mit JavaScript machen kann und wie es auf dem vorherigen Semester, in dem statische Webseitenentwicklung behandelt wurde, aufbaut. Des Weiteren wird der Zweck des Tutorials erläutert, nämlich sich eine effektive Methode zur Lösung von Problemen anzueignen.

4.1.4.2 *Erste Schritte*

Der „Erste Schritte“ Teil des Tutorials dient dazu, die Schüler anhand von zwei einfachen Beispielen mit den Grundelementen sowohl von PBL wie auch von JavaScript vertraut zu machen. Das erste Beispiel fordert den Schüler auf, ein vorgegebenes zweizeiliges Skript auszuprobieren und anschließend eine Reihe von Fragen zu beantworten, zwischen denen grundlegende Erklärungen im Text gegeben werden. Die Antworten auf die Fragen sind ebenfalls vorhanden, werden allerdings erst sichtbar, wenn man auf das Pinguinsymbol vor der Frage klickt. Hierauf habe ich die Schüler erst später aufmerksam gemacht, um sie zu ermutigen, die Fragen durch eigenes Recherchieren und Ausprobieren zu beantworten.

Die erste Frage wird im Text beantwortet. Anschließend werden die Begriffe Variable, Methode, Funktion und Objekt kurz erklärt. Die folgenden Fragen werden indirekt mit Bezug auf das Lehrbuch beantwortet. Hier ging es mir darum, die Schüler daran zu gewöhnen, in ihrem Buch Begriffe gezielt mittels des Indexes oder des Inhaltsverzeichnisses nachzuschlagen. Da die Lehrbücher entgegen meiner Erwartungen erst drei Wochen nach Semesterbeginn zur Verfügung standen, verwies ich die Schüler auf die Bibliothek der Website, die Links zu den wichtigsten Webseiten zum Thema JavaScript enthält. Hierbei bestätigten sich meine Beobachtungen aus meiner bisherigen Unterrichtspraxis, dass es große Unterschiede in der Internetsuchkompetenz der einzelnen Schüler gibt. So fanden einige Schüler korrekte Antworten im Handumdrehen, während andere sich hierbei sehr viel schwerer taten. Letzteren versuchte ich einen effektiveren Forschungsprozess zu vermitteln, indem ich fragte, wonach sie gesucht hatten und beobachtete, wie sie mit den Suchergebnissen umgingen. Ich zeigte ihnen dann, wie ich vorgehen würde und ermutigte sie diesen Ansatz auszuprobieren.

In der Antwort auf Frage 3 habe ich versucht, den Fokus auf den Lösungsprozess zu lenken, indem ich schriftlich festgehalten habe, mit welchen Fragen und Überlegungen man sich dem Ziel Schritt für Schritt nähern kann. Dabei habe ich die Klasse auch zum Ausprobieren angeregt, da ich beobachtete, wie einige Schüler versuchten, der Lösung rein theoretisch näher zu kommen, ohne dabei

ein Gefühl für die Materie zu entwickeln.

Die weiteren Fragen und Anweisungen dienten dazu, das einfache Beispiel aus verschiedenen Blickwinkeln zu analysieren und damit zu experimentieren.

Die zweitletzte Anweisung zielte darauf, die Sozialkompetenz der Schüler zu fördern und zugleich sie daran zu gewöhnen, Rückmeldung von ihren Kameraden zu erhalten und an dieselben zu geben. Dies war für die meisten ungewohnt und ich beobachtete hier eine ausgeprägte Zurückhaltung in Bezug auf die kritische Hinterfragung der gegebenen Erklärungen. Letzteren Teil übernahm ich dann des Öfteren, auch um den Wert einer kritischen Rückmeldung für den Erklärenden vorzuführen.

In der Literatur zu PBL wird viel Wert darauf gelegt, dass die Schüler dokumentieren, was sie wie gelernt haben. Des Weiteren soll die soziale Kompetenz durch Gruppenarbeit gefördert werden. Dies betrachte ich insbesondere auch im Rahmen der Reform der Technikerausbildung und der Vorbereitung auf die Berufspraxis als sehr wichtig (cf. 4.1.6).

Auf der Grundlage des in 2.7 besprochenen Lernablaufs habe ich fünf Etappen vorgegeben, nach denen jedes der sieben Probleme bearbeitet werden sollte, wobei die Schüler in Gruppen von jeweils etwa vier Personen eingeteilt wurden:

1. Analyse des Problems in der Gruppe.
2. Entwicklung von Lösungsideen.
3. Definition der Lernziele.
4. Erarbeitung der Lernziele in Einzelarbeit.
5. Besprechung der Lösungen in der Gruppe und gegebenenfalls Coaching der Gruppenmitglieder, die das Problem nicht vollständig lösen konnten.

Obwohl ich diesen Ablauf mehrfach, anhand der oben beschriebenen Beispiele, erklärt hatte, taten sich die meisten Schüler sehr schwer damit. Hier zeigte sich, neben den Schwierigkeiten der meisten Schüler, sich schriftlich auszudrücken, dass die Auseinandersetzung mit dem eigenen Lernprozess für sie vollkommen ungewohnt war. Daraufhin stellte ich der Klasse nach der Besprechung der Vorgehensweise zur Lösung des ersten Problems eine Musterlösung (cf. 9.1.1) mit den fünf Etappen zur Verfügung. Da dies jedoch bei der zweiten Problemstellung keine erkennbare Verbesserung brachte, obwohl einige Schüler viel Zeit damit verbrachten, kam ich zum Schluss, dass ich mit mei-

ner Erwartung, die Klasse könnte sich gleichzeitig eine neue Programmiersprache und die metakognitiven Prozesse zur Analyse des eigenen Lernprozesses erarbeiten, zu hoch gegriffen hatte. Dies ging auch klar aus dem Feedback vieler Schüler hervor. Hinzu kam, dass einige Schüler dazu tendierten, sich hinter der Gruppe sozusagen zu verstecken, d.h. sie gaben an, das Problem gelöst und verstanden zu haben, konnten die Lösung jedoch nicht erklären. Vier Wochen nach Semesterbeginn beschloss ich daher den Aspekt der Analyse des eigenen Lernprozesses vorerst zurück zu stellen und den Fokus auf das eigenständige Erlernen von JavaScript anhand der Problemstellungen zu lenken.

4.1.4.3 *Problemstellungen*

Im Folgenden gehe ich detaillierter auf die einzelnen Problemstellungen ein um die Lernziele und die gemachten Erfahrungen zu erläutern.

Problem 1

Um den Einstieg möglichst motivierend (cf. 2.4) zu gestalten, habe ich eine Problemstellung gewählt, die es ermöglicht, mit geringem Programmieraufwand, einen Zirkusaffen auf einem Skateboard mittels Mausklick auf dem Bildschirm zu bewegen:

```
<!Dml>
<html lang="de">
    <head>
        <title>Problem 1</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="width: 500px; height: 300px; overflow: hidden">
            <img src="gorilla236x256.png" id="gorilla"
                 title="Source: http://www.iconfinder.com/search/?q=Gorilla"
                 onclick = "move();" width="236" height="256" alt="gorilla236x256.png"
                 style="position: absolute; top: 0px; left: 260px">
        </div>
        <script>
            function move() {
                var gorilla = document.getElementById("gorilla");
                gorilla.style.left = parseInt(gorilla.style.left) - 10 + "px";
            }
        </script>
    </body>
</html>
```

Da im zweiten Beispiel bereits die Einbindung von einem Event Handler für Mausklickereignisse sowie der Zugriff auf HTML-Elemente mittels JavaScript behandelt wurden, bestand die Neuigkeit bei diesem Problem darin, die horizontale Position des HTML-Elementes zu verändern. Da Positio-

nen in CSS aus einer Zahl gefolgt von der Endung „px“ bestehen, musste hier ein Weg gefunden werden, eine solche Position in JavaScript zu generieren. Dies erforderte Kenntnisse des +-Operators zur Zusammensetzung von Zeichenketten, der in Kapitel 3.8 des Buches behandelt wird.

Um über die aktuelle Position des Affen zu verfügen gab es zwei Möglichkeiten: bei jedem Mausklick die aktuelle Position auslesen, wozu die Funktion `parseInt` aus Kapitel 5.6 benötigt wird, um die Endung „px“ zu entfernen oder die Ausgangsposition in einer globalen Variablen speichern und diese dann bei jedem Mausklick ändern.

Im Rahmen der Besprechung von Beispiel 1 waren die Begriffe Konsole und Debugger bereits eingeführt worden. Ich ließ der Klasse Zeit, um auf das Problem mit der Endung „px“ zu stoßen. Dies gelang einigen Schülern von selbst, als sie anfingen die Konsole zu verwenden, wo die Fehlermeldungen angezeigt werden. Die anderen erinnerte ich daran, wie Positionen in CSS gespeichert werden und verwies sie auf die Tabelle im Buch, wo die benötigte Funktion beschrieben wird. Da das Konzept der globalen Variable bereits in Beispiel 2 behandelt worden war, fiel es den meisten Schülern leichter, diesen Weg zu gehen anstatt sich mit der neuen Funktion auseinander zu setzen.

Diese Problemstellung wurde am 16.10.12 abgeschlossen. Wie oben beschrieben, konnten die schwächeren Schüler mir die Lösung ihrer Gruppe nicht vollständig erklären, woraufhin ich sie ihnen einzeln erklärt habe.

Problem 2

Diese Problemstellung zielt darauf ab, die Alternativstruktur `if else` einzuführen:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 2</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="width: 500px; height: 300px">
            <img src="gorilla236x256.png" id = "gorilla"
                 title = "Source: http://www.iconfinder.com/search/?q=Gorilla"
                 onclick="move();" width="236" height="256" alt = "gorilla236x256.png"
                 style="position: relative; top: 0px; left: 260px">
        </div>
        <script>
            var gorilla = document.getElementById("gorilla");
            var direction = -10;

            function move() {
                var left = parseInt(gorilla.style.left);
                if (left < 10) direction = 10;
```

```
                else if (left + gorilla.width > 490) direction = -10;
                gorilla.style.left = left + direction + "px";
            }
        </script>
    </body>
</html>
```

Diese wird benötigt, um zu verhindern, dass der Affe den Käfig verlässt. Dies erfordert jedoch auch, dass die aktuelle Bewegungsrichtung gespeichert wird, was nur von wenigen Schülern aus eigener Überlegung erkannt wurde. Dies liegt meines Erachtens daran, dass das Konzept der Variable noch neu und nicht in seiner ganzen Tiefe verstanden worden war.

Nachdem ich auf diese Erfordernis aufmerksam gemacht hatte und ein Beispiel gegeben hatte, wie man dies lösen könnte, konnten die meisten das Problem meistern.

Die Analyse und Dokumentation des Problemlöseprozesses erwies sich wieder als überaus unergiebig und zeitraubend, so dass, wie oben beschrieben, für die folgenden Problemstellungen dieser Ansatz ausgesetzt wurde.

Hinzu kam, dass die besorgte Mutter eines Schülers mich am 16.10.12 per Email fragte, ob ich nicht der Klasse die Materie auf traditionelle Art und Weise erklären könnte, da ihr Sohn selbst nicht mehr zurechtkäme. Am darauffolgenden Tag habe ich mit dem Schüler gesprochen, um herauszufinden, was ihm Schwierigkeiten bereitete. Ich habe ihm die relevanten Punkte erklärt und ihm vorgeführt, wie man sich dieses Wissen eigenständig aneignen kann. Ich habe ihm auch das wiederholt, was ich der ganzen Klasse von Anfang an gesagt hatte, nämlich dass sie mich jederzeit per Email oder Skype kontaktieren könnten. Das Gespräch hat sich sehr positiv ausgewirkt, da dieser Schüler seitdem große Fortschritte gemacht hat, und auch öfters Kontakt mit mir aufgenommen hat um seine Lösungen mit mir zu besprechen.

Diese Problemstellung wurde am 26.10.12 korrigiert und abgeschlossen.

Problem 3

Hier ging es darum herauszufinden, wie man auf Tastaturereignisse reagieren kann, um den Affen mittels der Pfeiltasten horizontal und vertikal bewegen zu können:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 3</title>
        <meta charset="UTF-8">
    </head>
    <body>
```

```
        <img src="gorilla236x256.png" id="gorilla"
               title="Source: http://www.iconfinder.com/search/?q=Gorilla" width="236" height="256"
               alt="gorilla236x256.png" style="position: relative; top: 0px; left: 260px">
        <script>
            var gorilla = document.getElementById("gorilla");

            function move(x, y) {
                var left = parseInt(gorilla.style.left), top = parseInt(gorilla.style.top);
                if (left + x >= 0 && left + x + gorilla.width <= 1200)
                    gorilla.style.left = left + x + "px";
                if (top + y >= 0 && top + y + gorilla.height <= 800)
                    gorilla.style.top = top + y + "px";
            }

            function keyHandler(event) {
                if (event.keyCode === 37) move(-10, 0);
                else if (event.keyCode === 38) move(0, -10);
                else if (event.keyCode === 39) move(10, 0);
                else if (event.keyCode === 40) move(0, 10);
            }

            onkeydown = keyHandler;
        </script>
    </body>
</html>
```

Im Buch werden Tastaturereignisse nur im Anhang aufgelistet. Die meisten Schüler fanden jedoch mittels Internetrecherche heraus, wie Tastaturereignisse behandelt werden können. Um den Affen auch in vertikaler Richtung bewegen zu können, musste man entweder eine Funktion mit Parametern definieren, was erst Bestandteil des CLISS 2 Lehrplans ist, oder aber eine oder mehrere globale Variablen definieren und dann mit mehreren `if else` Abfragen entscheiden, in welche Richtung der Affe bewegt werden soll. Dies stellte eine gute Wiederholung der vorher erlernten Konzepte dar.

Diese Problemstellung war am 6.11.12 von 17 der 21 Schüler gelöst und wurde dann besprochen.

Problem 4

Dieses Problem stellt eine Verfeinerung von Problem 3 dar, da hier auf zwei gleichzeitig gedrückte Pfeiltasten reagiert werden muss. Dies erfordert nicht nur das Drücken einer Taste zu erkennen, sondern auch das Loslassen. Darüber hinaus muss der Zustand jeder der vier Pfeiltasten in Variablen gespeichert werden:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 4</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <img src="gorilla236x256.png" id="gorilla"
```

```
            title="Source: http://www.iconfinder.com/search/?q=Gorilla" width="236" height="256"
            alt="gorilla236x256.png" style="position: relative; top: 0px; left: 260px">
        <script>
            var moveRight = moveLeft = moveUp = moveDown = false, step = 10, cageWidth = 1200,
                cageHeight = 800;
            var gorilla = document.getElementById("gorilla");

            function move() {
                var left = parseInt(gorilla.style.left), top = parseInt(gorilla.style.top);
                if (moveLeft) {
                    if (left - step >= 0) gorilla.style.left = left - step + "px";
                    else gorilla.style.left = "0px";
                }
                if (moveRight) {
                    if (left + step + gorilla.width <= cageWidth)
                        gorilla.style.left = left + step + "px";
                    else gorilla.style.left = cageWidth - gorilla.width + "px";
                }
                if (moveUp) {
                    if (top - step >= 0) gorilla.style.top = top - step + "px";
                    else gorilla.style.top = "0px";
                }
                if (moveDown) {
                    if (top + step + gorilla.height <= cageHeight)
                        gorilla.style.top = top + step + "px";
                    else gorilla.style.top = cageHeight - gorilla.height + "px";
                }
            }

            function keyDownHandler(event) {
                if (event.keyCode === 37) moveLeft = true;
                else if (event.keyCode === 38) moveUp = true;
                else if (event.keyCode === 39) moveRight = true;
                else if (event.keyCode === 40) moveDown = true;
                move();
            }

            function keyUpHandler(event) {
                if (event.keyCode === 37) moveLeft = false;
                else if (event.keyCode === 38) moveUp = false;
                else if (event.keyCode === 39) moveRight = false;
                else if (event.keyCode === 40) moveDown = false;
            }

            onkeydown = keyDownHandler;
              onkeyup = keyUpHandler;
        </script>
    </body>
</html>
```

11 Schüler hatten dieses Problem am 8.11.12 gelöst.

Problem 5

Bei diesem Problem geht es wiederum darum, ein Bild, diesmal einen Fußball, zu bewegen. Allerdings wird dieser nicht per Mausklick oder Tastendruck bewegt, sondern bewegt sich scheinbar von

selbst, sobald der Startknopf gedrückt wurde. Dies erfordert den Einsatz eines Timers:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 5</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div>
          <button id="start" onclick="start();">Start</button>
            <button id="stop" onclick="stop();" style="display: none">Stop</button>
        </div>
       <div style="position: absolute; top: 0px; left: 0px; z-index: -5">
            <img src="football352x352.png" id="football" width="352" height="352"
                 alt="football352x352.png" style="position: relative; top: 10px; left: 10px">
        </div>
        <script>
            var xDirection = 1, yDirection = 1, timerID, timeout = 20, step = 10, ballWidth = 352,
                ballHeight = 352;
             var ball = document.getElementById("football");

             function start() {
            /* Der Ball bewegt sich um step Pixel horizontal und vertikal alle timeout Millisekunden.
             * Wird ein Rand berührt ändert sich die horizontale resp. vertikale Bewegungsrichtung.
             * Drücken des Startknopfes startet die Bewegung des Balls, lässt den Startknopf
             * verschwinden sowie den Stoppknopf erscheinen.
             * Drücken des Stoppknopfes stoppt die Ballbewegung, lässt den Stoppknopf verschwinden
             * und den Startknopf erscheinen. */
                // Die CSS-Attribute "left" und "top" enthalten Zeichenketten mit "px"
                var x = parseInt(ball.style.left);
                // am Ende. Wir müssen sie in Ganzzahlen umwandeln.
                var y = parseInt(ball.style.top);
                // Der Ball darf das Spielfeld nicht verlassen.
                if ((x + ballWidth + xDirection * step) >= innerWidth) xDirection = -1;
                else if ((x + xDirection * step) <= 0) xDirection = 1;
                ball.style.left = x + xDirection * step + "px";
                if ((y + ballHeight + yDirection * step) >= innerHeight) yDirection = -1;
                else if ((y + yDirection * step) <= 0) yDirection = 1;
                ball.style.top = y + yDirection * step + "px";
                document.getElementById("start").style.display = "none";
                document.getElementById("stop").style.display = "block";
                timerID = setTimeout("start()", timeout);
             }

             function stop() {
                clearTimeout(timerID);
                document.getElementById("start").style.display = "block";
                document.getElementById("stop").style.display = "none";
             }
        </script>
    </body>
</html>
```

Obwohl ich die zu verwendenden Methoden erklärt hatte, tat die Klasse sich sehr schwer, diese einzusetzen. Das Abstraktionsniveau war hier offensichtlich zu hoch und es fehlte den Schülern an ausreichend Erfahrung um diese Problemstellung zu lösen. Dies überraschte mich, da außer dem Ein-

satz des Timers nichts konzeptionell Neues erfordert ist. Aus meinen Gesprächen mit den Schülern erkannte ich jedoch, dass dies noch zu früh war.

Nur 4 Schülern gelang es innerhalb einer Woche dieses Problem zu lösen. Ich erklärte die Lösung des Problems am 16.11.12 und gab der Klasse mehr Zeit um das Problem zu lösen. Am 23.11.12 hatten noch immer eine Reihe Schüler keine funktionierende Lösung, so dass ich die Lösung nochmals erklärte.

Problem 6

Aufbauend auf Problem 5 soll hier ein Eingabefeld zur Geschwindigkeitsangabe sowie ein Ausgabebereich zur Anzeige der Ballpositionen hinzugefügt werden:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 6</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <button id="start" onclick="start();" style="position: absolute; top: 0px;
                 left: 0px">Start</button>
        <button id="stop" onclick="stop();" style="position: absolute; top: 0px; left: 0px;
                 display: none">Stop</button>
        <button id="clearLog" onclick="document.myForm.output.value='';"
                style="position: absolute; top: 30px; left: 0px">Clear log</button>
        <div style="position: absolute; top: 60px; left: 0px">
            <form name="myForm" style="position: fixed">
                <label>Speed:</label>
                <input type="range" step="1" min="0" max="100" value="10" name="speed">
                <br>
                <textarea cols="30" rows="25" name="output"></textarea>
            </form>
        </div>
        <div style="position: absolute; top: 0px; left: 0px">
            <img src="football352x352.png" id="football" width="352" height="352"
                 alt="football352x352.png" style="position: absolute; top: 50px; left: 50px">
        </div>
        <script>
            var xDirection = 1, yDirection = 1, timerID, timeout = 20, step = 10, ballWidth = 352,
                ballHeight = 352;
            var ball = document.getElementById("football");

            function start() {
            /* Der Ball bewegt sich um step Pixel horizontal und vertikal alle timeout Millisekunden.
             * Wird ein Rand berührt ändert sich die horizontale resp. vertikale Bewegungsrichtung.
             * Drücken des Startknopfes startet die Bewegung des Balls, lässt den Startknopf
             * verschwinden sowie den Stoppknopf erscheinen.
             * Drücken des Stoppknopfes stoppt die Ballbewegung, lässt den Stoppknopf verschwinden
             * und den Startknopf erscheinen. */
                // Die CSS-Attribute "left" und "top" enthalten Zeichenketten mit "px"
                var x = parseInt(ball.style.left);
                // am Ende. Wir müssen sie in Ganzzahlen umwandeln.
```

```
                var y = parseInt(ball.style.top);
                step = document.myForm.speed.value;
                // Der Ball darf das Spielfeld nicht verlassen.
                if ((x + ballWidth + xDirection * step) >= innerWidth) xDirection = -1;
                else if ((x + xDirection * step) <= 0) xDirection = 1;
                ball.style.left = x + xDirection * step + "px";
                if ((y + ballHeight + yDirection * step) >= innerHeight) yDirection = -1;
                else if ((y + yDirection * step) <= 0) yDirection = 1;
                ball.style.top = y + yDirection * step + "px";
                document.getElementById("start").style.display = "none";
                document.getElementById("stop").style.display = "block";
                document.myForm.output.value += "top: " + ball.style.top + " left: " +
                            ball.style.left + "\n";
                timerID = setTimeout("start()", timeout);
            }

            function stop() {
                clearTimeout(timerID);
                document.getElementById("start").style.display = "block";
                document.getElementById("stop").style.display = "none";
            }
        </script>
    </body>
</html>
```

Die einzige wirkliche Schwierigkeit bestand hier darin herauszufinden, wie man Text im Ausgabebereich hinzufügt, ohne den vorhandenen Text zu löschen.

Aufgrund meiner Betonung der Wichtigkeit des Operatorenkapitels im Buch hatte ich nicht erwartet, dass dies für viele Schüler eine derartige Hürde darstellen würde. Am 27.11.12 erklärte ich die Musterlösung. 8 Schüler hatten das Problem korrekt gelöst.

Problem 7

Dieses Problem soll zur Einführung der Felder (arrays) dienen. Anstatt eines einzigen Balls kann hier der Benutzer die Anzahl der Bälle dynamisch mittels dem Eingabefeld festlegen. Ein Feld ist hierfür zwar nicht zwingend erforderlich, jedoch die eleganteste Lösung:

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Problem 7</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <button id="start" onclick="start();" style="position: absolute; top: 0px;
                left: 0px">Start</button>
        <button id="stop" onclick="stop();" style="position: absolute; top: 0px; left: 0px;
                display: none">Stopp</button>
        <script>
            var div1 = document.createElement("div");
            div1.style.cssText = "position: absolute; top: 30px; left: 0px";
```

```
var form = document.createElement("form");
form.name = "myForm";
form.style.position = "fixed";
var label = document.createElement("label");
label.setAttribute("for", "numBalls");
label.innerHTML = "Number of balls:";
var input = document.createElement("input");
input.type = "range";
input.step = "1";
input.min = "0";
input.max = "10";
input.value = "5";
input.name = "numBalls";
input.onchange = (function() {stop();start();});
var body = document.getElementsByTagName("body")[0];
body.appendChild(buttonStart);
body.appendChild(buttonStop);
label.appendChild(input);
form.appendChild(label);
div1.appendChild(form);
body.appendChild(div1);

var xDirection = 1, yDirection = 1, timerID, timeout = 20, step = 10, ballWidth = 352,
    ballHeight = 352, text="";
var balls = [], init = true, body = document.getElementsByTagName("body")[0];
var img = new Image(), loaded = false;
img.src = "football352x352.png";
img.onload = function() {loaded = true;};

function start() {
    if (!loaded) return;
    if (init) {
        for (var i = 0; i < document.myForm.numBalls.value; i++) {
            var x = document.createElement("img");
            x.src = "football352x352.png";
            x.width = "352";
            x.height = "352";
            x.style.left = i * 50 + "px";
            x.style.top = i * 50 + "px";
            x.style.position = "absolute";
            x.alt = "football352x352.png";
            body.appendChild(x);
            balls.push(x);
        }
        init = false;
    }
    else {
        for (var i = 0; i < balls.length; i++) {
            var ball = balls[i];
            // Die CSS-Attribute "left" und "top" enthalten Zeichenketten mit "px"
            // am Ende. Wir müssen sie in Ganzzahlen umwandeln.
            var x = parseInt(ball.style.left);
            var y = parseInt(ball.style.top);
            step = Math.random() * (i + 2) * 10 + 1;
            // Der Ball darf das Spielfeld nicht verlassen.
            if ((x + ballWidth + xDirection * step) >= innerWidth) xDirection = -1;
            else if ((x + xDirection * step) <= 0) xDirection = 1;
            ball.style.left = x + xDirection * step + "px";
            if ((y + ballHeight + yDirection * step) >= innerHeight) yDirection = -1;
```

```
                    else if ((y + yDirection * step) <= 0) yDirection = 1;
                    ball.style.top = y + yDirection * step + "px";
                    document.getElementById("start").style.display = "none";
                    document.getElementById("stop").style.display = "block";
                }
            }
            timerID = setTimeout("start()", timeout);
        }

        function stop() {
            for (var i = 0; i < balls.length; i++) body.removeChild(balls[i]);
            balls = [];
            init = true;
            clearTimeout(timerID);
            document.getElementById("start").style.display = "block";
            document.getElementById("stop").style.display = "none";
        }
    </script>
</body>
</html>
```

Dieses Problem wurde von 3 Schülern gelöst. Ich nutzte die Gelegenheit um der Klasse anhand der Musterlösung nochmals zu zeigen, wie man mittels JavaScript neue Elemente im DOM erstellen kann.

4.1.5 Coaching

Da ich mit der Lernfortschrittsentwicklung nicht zufrieden war und eine wachsende Diskrepanz zwischen den Kompetenzen der stärksten und schwächsten Schüler fest stellte, beschloss ich ein Tutorensystem einzuführen. Ich erklärte der Klasse den Zweck und präsentierte meinen Vorschlag einer neuen Sitzordnung, der positiv aufgenommen wurde. Die 8 besten Schüler habe ich so platziert, dass sie jeweils einen oder zwei schwächere Schüler neben sich sitzen haben, für deren Weiterentwicklung sie mitverantwortlich sind.

Da ich die Klasse nur 4 Stunden die Woche sehe und mir bei 21 Schülern nicht viel Zeit auf individueller Basis zur Verfügung steht, hat mir dieses Tutorensystem ermöglicht, mehr Zeit mit den schwächsten Schülern zu verbringen.

Meine Beobachtungen während und meine Nachfrage am Ende des ersten Semesters haben bestätigt, dass das Tutorensystem, bis auf die Ausnahme eines Schülers, der seinem Tutor nicht eine einzige Frage stellte, den schwächeren Schülern geholfen hat.

Obwohl ich wiederholt darauf hingewiesen habe, haben nur 3 Schüler die Möglichkeit mir außerhalb der Unterrichtsstunden Fragen per Email, Facebook oder via Skype zu stellen, regelmäßig genutzt. Diejenigen, die es getan haben, haben deutlich davon profitiert, was ich an ihren Fortschritten

erkennen konnte.

4.1.6 Gruppenarbeit

Wie in 4.1.4 beschrieben, gehört die Förderung der sozialen Kompetenz zu den Zielen von PBL. Da die systematische Vorgehensweise nach dem 5-Punkte Plan sich als nicht umsetzbar erwies, wurden die formalen Gruppen aufgelöst. Allerdings setzte sich die Zusammenarbeit auf informeller Ebene, vorrangig mit den Banknachbarn, fort. Ich beobachtete bei den meisten Schülern einen regelmäßigen Austausch. Dies bestätigt auch die Auswertung der Schülerumfrage (cf. 9.3), wo die Aussage „Ich arbeite gerne in einer Gruppe“ mit 3,9/5 die höchste Zustimmung erhielt.

4.1.7 Trainingsübungen

Aufgrund der immer stärkeren Entwicklungsdiskrepanzen zwischen den einzelnen Schülern sowie der wiederholten Rückmeldung, selbst von sehr guten Schülern, dass sie alleine nicht weiter kämen und angesichts der dahinschwindenden Zeit und Rücksprache mit Kollegen, entschloss ich mich dazu, ein an den einzelnen Lerninhalten orientiertes Trainingsprogramm zu entwickeln.

Hiermit wollte ich sicher stellen, dass die individuellen Lerninhalte verstanden und geübt wurden, um die Lücken zu stopfen, die im Grundlagenwissen vieler Schüler vorhanden waren.

Diese Übungen entsprechen eher den Festigungsübungen, die einer traditionellen Frontalunterrichtsstunde folgen, da sie sich auf einen oder wenige spezifische Lerninhalte konzentrieren und somit eine geringere Komplexität beinhalten.

Dabei ging ich folgendermaßen vor: Ich erklärte den jeweiligen Lerninhalt am Schirm anhand von konkreten Beispielen mittels der Konsole und gab der Klasse anschließend Zeit, die entsprechenden Übungen zu lösen, während ich durch die Reihen ging und Fragen beantwortete. Um mir hierbei zu helfen, hatte ich nach dem ersten Test den besten Schüler zu meinem Hilfsassistenten „befördert“. Er half mir, Fragen zu beantworten sowie Buch darüber zu führen, wer welche Übungen gelöst hatte.

Ich habe viel Aufwand betrieben, um es den Schülern zu ermöglichen, ihre Lösungen auf der Übungswebseite eingeben zu können und mittels der Farben Grün und Rot sofortige Rückmeldung zu erhalten, ohne auf die Lehrkraft warten zu müssen. Dies stellt eine Art der Selbstevaluation (cf. [14] Reich S. 43) dar und hat meines Erachtens und laut Aussage mehrerer Schüler die Motivation erhöht und die Lerneffizienz gesteigert. Allerdings wird diese Einschätzung mit einem Wert von

2,8/5 nicht von der Schülerumfrage bestätigt.

Zwischen dem 20.11.12 und dem 18.12.12 wurden die Übungen 1 – 13 bearbeitet. In diesen Übungen werden Variablen, Operatoren, Kontrollstrukturen, vordefinierte Funktionen, benutzerdefinierte Funktionen ohne Parameter sowie verschachtelte Kontrollstrukturen angewendet. Diese Konzepte waren bereits bei den Problemstellungen behandelt worden:

Übung	Musterlösung
1	`2 undefined 17.5 Hallo`
2	`var x = 7, y = 23.9, z = "JavaScript macht Spaß";`
3	`var z1 = x + y, z2 = x % y;`
4	*true, true, 26, 1, false, true, "no", true, true*
5	`"2613", "26Hallo ", "26Welt", "1326", 2, NaN, "Hallo Welt"`
6	`if (grade >= 50) result = "sehr gut!";`
7	`if (grade >= 30) result = "genügend";` `else result = "ungenügend";`

8	`if (grade >= 1 && grade <= 9) result = "sehr schlecht";`
	`if (grade >= 10 && grade <= 19) result = "schlecht";`
	`if (grade >= 20 && grade <= 29) result = "ungenügend";`
	`if (grade >= 30 && grade <= 39) result = "genügend";`
	`if (grade >= 40 && grade <= 49) result = "gut";`
	`if (grade >= 50 && grade <= 59) result = "sehr gut";`
	`if (grade === 60) result = "exzellent";`

9	`for (var i = 1; i <= u9Num; i++) func();`
10	`var result = 1;` `for (var i = 1; i <= num; i++) result *= i;`

11	`var sum = parseFloat(s1) + parseFloat(s2), difference = parseFloat(s1) - parseFloat(s2);`
	`var product = parseFloat(s1) * parseFloat(s2), quotient = parseFloat(s1) / parseFloat(s2);`
	oder
	`var sum = Number(s1) + Number(s2), difference = Number(s1) - Number(s2);`
	`var product = Number(s1) * Number(s2), quotient = Number(s1) / Number(s2);`

12	`function sum() {return a + b;}`
	oder
	`function sum() a+b;`

13	`function sumEvenBetween() {`
	`  var sum = 0;`
	`  for (var i = a; i <= b; i++) {`
	`  if (i % 2 === 0) sum += i;`
	`  }`
	`  return sum;`
	`}`

Den meisten Schülern gelang es, diese Übungen selbstständig zu lösen, einige benötigten Hilfestellung.

Die Übung 14 behandelt das Thema der verschachtelten Kontrollstrukturen, allerdings in Verbindung mit einem Funktionsaufruf, wobei einer der Parameter für diese Funktion in der inneren Schleife erstellt werden muss:

```
for (var i = 0; i < 128; i++)
  for (var j = 0; j < 128; j++) draw(i, j, "rgb(" + (i+j) + "," + (i+j) + "," + (i+j) + ")");
```

Dies erwies sich, bis auf 2 Ausnahmen, als zu anspruchsvoll, so dass ich diese Übung detaillierter erklärte.

Die Übungen 15-18 behandeln einige vordefinierte JavaScript-Funktionen:

Übung	Musterlösung

15	`var len = s.length;`
	`var lastE = s.lastIndexOf('e');`
	`var cut = s.slice(2, s.length - 3);`
	`var small = s.toLowerCase();`
	`function getMinCharCode() {`
	`  var min = s.charCodeAt(0);`
	`  for (var i = 1; i < s.length; i++)`
	`  if (s.charCodeAt(i) < min) min = s.charCodeAt(i);`
	`  return min;`
	`}`

16	`var minAB = Math.min(a, b);`
	`var maxBC = Math.max(b, c);`
	`var minABC = Math.min(a, Math.min(b, c));`
	`var power = Math.pow(a, b);`
	`function getGrade() {`
	`  return Math.floor(Math.random() * 60) + 1;`
	`}`
	`function getCircumference() {`
	`  return Math.PI * a;`
	`}`

17	`var x1 = a.toExponential(2);`
	`var s1 = b.toString().slice(2, 6);`
	`function getRound() {`
	`  return c.toFixed(1);`
	`}`

18	`function dateFunc() {`
	`  dDayOfMonth = myDate.getDate();`
	`  dDayOfWeek = myDate.getDay();`
	`  dHour = myDate.getHours();`
	`  dMin = myDate.getMinutes();`
	`  dMon = myDate.getMonth();`
	`  dSec = myDate.getSeconds();`
	`  dMilliSec = myDate.getTime();`
	`  dOffset = myDate.getTimezoneOffset();`
	`  dYear = myDate.getFullYear();`
	`  sGreenwich = myDate.toGMTString();`
	`  sLocale = myDate.toLocaleString();`
	`  myNewDate = new Date(dYear, dMon + 1, dDayOfMonth);`
	`}`

Die einzige Schwierigkeit besteht darin, die angegebenen Seiten im Buch zu lesen und anzuwenden. Gerade dies fällt dieser Klasse jedoch sehr schwer. So gut wie niemand hatte die Übungen gelöst. Da es bereits der 11.1.13 war, habe ich die Lösungen am Schirm erklärt und bin sofort zur Einführung der eindimensionalen Felder übergegangen, da dies ein wichtiges aber auch neues Konzept für

die Klasse darstellte.

Nach einer vertieften Erklärung und Besprechung anhand von Beispielen am Schirm habe ich die Klasse aufgefordert, die Übung 19 zuhause zu lösen:

```
function dateFunc() {
    dDayOfMonth = myDate.getDate();
    dDayOfWeek = myDate.getDay();
    dHour = myDate.getHours();
    dMin = myDate.getMinutes();
    dMon = myDate.getMonth();
    dSec = myDate.getSeconds();
    dMilliSec = myDate.getTime();
    dOffset = myDate.getTimezoneOffset();
    dYear = myDate.getFullYear();
    sGreenwich = myDate.toGMTString();
    sLocale = myDate.toLocaleString();
    myNewDate = new Date(dYear, dMon + 1, dDayOfMonth);
}
```

Die Übungen 20 – 23 habe ich aus Zeitgründen übersprungen. Diese behandeln Themen, die von Anfang an bereits in den Problemstellungen ausführlich besprochen und angewendet wurden:

Übung	Lösung

20	`document.getElementById("myButton").onclick = function() {`
	`  result = "You just clicked me!"`
	`};`
	`oder`
	`function myFunc() {`
	`  result = "You just clicked me!";`
	`}`
	`document.getElementById("myButton").onclick = myFunc;`

21	`function eventHandler(event) {`
	`  if (event.keyCode === 88) result = "Taste X wurde gedrückt!";`
	`}`
	`onkeydown = eventHandler;`

22	`document.getElementById("myElement").innerHTML = document.URL + "\n" + document.title + "\n" + document.domain;`

23	`function f() {`
	`  if (document.getElementById("myInput").value === "CLISS1")`
	`  document.getElementById("myImg").src = "green_flag128x128.png";`
	`  else document.getElementById("myImg").src = "red_flag128x128.png";`
	`}`
	`document.getElementById("myButton").onclick = f;`

Als letzter wichtiger Punkt besprach ich die Verwendung von Auswahllisten mittels JavaScript. Hierzu stehen vier detaillierte Beispiele zum Ausprobieren bereit, die in der Klasse besprochen wurden. Die Übung 24, als letzte Übung zur obligatorischen Kompetenz 1, beinhaltet die Sortierung einer Auswahlliste mittels Knopfdruck. Dies erfordert das Kopieren des Inhalts der Auswahlliste in ein Feld, das dann sortiert und wieder zurück in die Auswahlliste kopiert wird:

```
function f() {
    var arr = [];
    for (var i = 0; i < mySelect.options.length; i++) arr[i] = mySelect.options[i].text;
    arr.sort();
    for (var i = 0; i < arr.length; i++) {
        mySelect.options[i].text = arr[i];
        mySelect.options[i].value = arr[i];
    }
}

myButton.onclick = f;
```

Diese Übung war als Hausaufgabe zu lösen. Obwohl das vierte Beispiel die Lösung zum größten Teil bereits beinhaltet, hatten lediglich 11 Schüler die Übung zu lösen versucht, fünf mit Erfolg. Das Zusammenspiel zwischen einer Auswahlliste und einem Feld war offensichtlich noch zu abstrakt, da die Klasse noch nicht Gelegenheit hatte, den Umgang mit Auswahllisten und Feldern jeweils getrennt hinreichend zu üben. Hierzu wäre eine Woche mehr Zeit erforderlich gewesen.

Für die Kompetenz 2, komplexere Skripte, hatte ich vier Übungen erstellt:

Übung	Lösung

25	`for (var i = 0; i < 150; i++)`
	`  for (var j = 0; j < 150; j++)`
	`  if (i % 20 !== 0 || j % 20 !== 0) {`
	`  if (j < 50) draw(i, j, "red");`
	`  else if (j < 100) draw(i, j, "white");`
	`  else draw(i, j, "blue");`
	`  }`

26	`function sumUpTo(arr, pos) {`
	`  var sum = 0;`
	`  if ((typeof arr === 'undefined') || (pos >= arr.length)) return -1;`
	`  for (var i = 0; i <= pos; i++) sum += arr[i];`
	`  return sum;`
	`}`

27	`function getBrowserInfo() {`
	`  var result = navigator.userAgent + "\n";`
	`  result += navigator.javaEnabled() ? "Dieser Browser unterstützt Java." : "Dieser Browser unterstützt Java nicht.";`
	`  result += "\n";`
	`  for (var i = 0; i < navigator.plugins.length; i++)`
	`  result += navigator.plugins[i].description + "\n";`
	`  return result;`
	`}`
	
	`function getDisplayInfo() {`
	`  return "Sichtbare Breite & Höhe: " + screen.availWidth + "x" + screen.availHeight;`
	`}`
	
	`function getInfo() {`
	`  return getBrowserInfo() + "\n" + getDisplayInfo() + "\nLänge der Historie: " +`
	`  history.length + "\n" + location.protocol + location.host + location.pathname;`
	`}`

28	`function createMatrix() {` `  var matrix = new Array(matrixSize);`

Übung	Lösung

	`var counter = 1`
	`for (var i = 0; i < matrixSize; i++) {`
	`matrix[i] = new Array(matrixSize);`
	`for (var j = 0; j < matrixSize; j++) matrix[i][j] = counter++;`
	`}`
	`return matrix;`
	`}`

Für die Kompetenz 3, Debuggertools, hatte ich Übung 29 erstellt, wo es darum geht ein fehlerhaftes HTML-Dokument zu korrigieren. Das korrigierte Dokument sieht folgendermaßen aus:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Fehlerhaftes Skript</title>
        <meta charset="UTF-8">
    </head>
    <body id="myBody" style="background-color: black;">
        <button id="myButton" style="display: none"></button>
        <script>
            var name = prompt("Bitte geben Sie Ihren Namen ein:");
            var myButton = document.getElementById("myButton");
            var myBody = document.getElementById("myBody");
            var currRed = 0, currGreen = 0, currBlue = 0;
            var timerID = null, timeout = 20;
            var RGBColors = [Math.floor(Math.random() * 256), Math.floor(Math.random() * 256),
                 Math.floor(Math.random() * 256)];
            var RGBDirections = [Math.random() >= 0.5 ? 1 : -1, Math.random() >= 0.5 ? 1 : -1,
                 Math.random() >= 0.5 ? 1 : -1];

            myButton.innerHTML = "Hallo " + name + ", bitte klicken Sie um fortzufahren.";
            myButton.onclick = click;
            myButton.style.display = "block";

            function click() {
                if (timerID === null) {
                    timerID = setInterval("blendIn()", timeout);
                    myButton.innerHTML = "Stopp";
                }
                else {
                    clearInterval(timerID);
                    timerID = null;
                    myButton.innerHTML = "Fortfahren";
                }
            }

            function blendIn() {
                for (var i = 0; i < RGBColors.length; i++) {
                    if (RGBColors[i] >= 254 && RGBDirections[i] === 1 || RGBColors[i] <= 1 &&
                        RGBDirections[i] === -1)
                        RGBDirections[i] = -RGBDirections[i];
                    RGBColors[i] += RGBDirections[i] * Math.floor(Math.random() * 2 + 0.5);
                }
                myBody.style.background = "rgb(" + RGBColors[0] + "," + RGBColors[1] + "," +
```

```
                    RGBColors[2] + ")";
            }
        </script>
    </body>
</html>
```

Für die Kompetenz 4, Informationen eigenständig nachschlagen und einsetzen, sollte die Klasse eine Pacman-Animation erstellen:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Pacman</title>
        <meta charset="UTF-8">
    </head>
    <body style="background-color: black; overflow: hidden">
        <script>
            var canvas = document.createElement("canvas");
            canvas.id = "canvas";

            function sizeCanvas() {
                canvas.width = window.innerWidth;
                canvas.height = window.innerHeight;
            }
            sizeCanvas();
            document.getElementsByTagName("body")[0].appendChild(canvas);
            var context = canvas.getContext('2d');
            var pacman = new Image();
            pacman.src = "pacman128x128.png";
            pacman.alt = "pacman128x128.png";

            var TO_RADIANS = Math.PI / 180, IE = document.all ? true : false;
            var mouseX = 0, mouseY = 0, currX = 100, currY = 100;
            var timerID = null, timeout = 20;

            function drawRotatedImage(image, x, y, angle) {
                // save the current co-ordinate system
                // before we screw with it
                context.save();

                // move to the middle of where we want to draw our image
                context.translate(x, y);

                // rotate around that point, converting our
                // angle from degrees to radians
                context.rotate(angle * TO_RADIANS);

                // draw it up and to the left by half the width
                // and height of the image
                context.drawImage(image, -(image.width / 2), -(image.height / 2));

                // and restore the co-ords to how they were when we began
                context.restore();
            }

            function getMouseXY(e) {
                if (IE) { // grab the x-y pos.s if browser is IE
                    mouseX = event.clientX + document.body.scrollLeft;
```

```
                mouseY = event.clientY + document.body.scrollTop;
            }
            else {  // grab the x-y pos.s if browser is NS
                mouseX = e.pageX;
                mouseY = e.pageY;
            }
        }

        function animate() {
            var dX = currX - mouseX, dY = currY - mouseY;
            var a = Math.floor(dX / 20), b = Math.floor(dY / 20);
            currX -= a;
            currY -= b;
            if ((Math.abs(a) >= 1) || (Math.abs(b) >= 1)) {
                canvas.width = canvas.width;
                //context.clearRect(0, 0, canvas.width, canvas.height);
                drawRotatedImage(pacman, currX, currY, (Math.atan2(dY, dX) - Math.PI) * 180 /
                    Math.PI);
            }
        }

        function toggle() {
            if (timerID === null) timerID = setInterval("animate()", timeout);
            else {
                clearInterval(timerID);
                timerID = null;
            }
        }

        onmousemove = getMouseXY;
        onclick = toggle;
        onresize = sizeCanvas;
        timerID = setInterval("animate()", timeout);
        document.getElementsByTagName("body")[0].style.cursor =
           'url("mouseWithCheese64x64.cur"), auto';
    </script>
  </body>
</html>
```

Für die Übungen 25 bis 30 verblieb nur wenig Zeit zwischen der Evaluation der Kompetenz 1 und den restlichen Kompetenzen, so dass ich der Klasse diese Übungen erklärte.

4.1.8 Videotutorials

Obwohl mit dem Tutorial und den dazugehörigen Problemstellungen, den Trainingsübungen und der Bibliothek bereits eine Vielzahl von unterschiedlichen Lernressourcen zur Verfügung standen, war ich davon überzeugt, dass ein Videotutorial ein sinnvolles zusätzliches Lernmedium für einige Schüler darstellen würde. Diese Überzeugung basierte einerseits auf der bekannten Tatsache, dass Menschen unterschiedliche Lernstile und bevorzugte Wahrnehmungspräferenzen haben. Andererseits weiß ich aus eigener Erfahrung, unter anderem mit Onlinekursen von Universitäten und Ausbildungsunternehmen, dass ein gut gemachtes Video es einem erlaubt, sich mit der Materie effizient,

zu einer beliebigen Zeit und in seinem eigenen Rhythmus auseinanderzusetzen. Dies ist besonders bei einer Klasse von 21 Schülern relevant, da es aufgrund der Klassendynamik nicht immer allen Schülern gelingt, sich auf das gerade behandelte Thema zu konzentrieren.

Darüber hinaus erhoffte ich mir, mittels den Videotutorials, die Schüler dazu bewegen zu können, sich intensiver außerhalb der regulären Unterrichtszeit mit der Materie zu befassen.

Die drei Videotutorials, mit einer Dauer von jeweils einer halben bis einer Stunde, sind wie folgt aufgebaut:

1. Eine komplexe Problemstellung wird erläutert und das zu erstellende Endprodukt vorgeführt. Wie bei allen Problemstellungen ist es auch hier wichtig, ein möglichst interessantes Problem zu verwenden, das die Schüler zur Lösung anspornt.

2. Die Lösung wird systematisch Schritt für Schritt im Editor entwickelt, wobei jeder Schritt erklärt wird und der Zusammenhang zum bereits Gelernten und zum Buch hergestellt wird. Hierbei werden gegebenenfalls auch Alternativen erkundet und deren Vor- und Nachteile besprochen.

   Der Bildschirm ist in zwei Bereiche aufgeteilt. Auf der linken Seite wird das Endprodukt vorgeführt oder die Angabe gezeigt oder der aktuelle Stand der Lösung überprüft und die Fehlersuche mittels Console und gegebenenfalls Debugger durchgeführt. Auf der rechten Seite ist das Editorfenster sichtbar, so dass der Zusammenhang zwischen Skript und Ergebnis leichter zu erkennen ist.

   Des Weiteren wird eine systematische Vorgehensweise bei der Überprüfung, Analyse und Fehlersuche vorgeführt, insbesondere unter Verwendung der Console und deren Debugger. Dies betrachte ich als sehr wichtig, da die Erfahrung zeigt, dass die meisten Schüler, trotz mehrfacher Hervorhebung und Vorführung im Unterricht, lange Zeit brauchen, um sich eine strukturierte Vorgehensweise bei der Fehlersuche zu erarbeiten. Insbesondere die Nichtverwendung der Console führt meistens zu Frustration und Zeitverlust, da selbst einfache Fehler nicht gefunden werden und das Erfolgserlebnis eines funktionierenden Skripts somit ausbleibt.

Von den einundzwanzig Schülern haben sich durchschnittlich sechs die drei Videos ganz angeschaut. Absolut betrachtet ist dies wenig, allerdings war mein Ziel auch nur, ein zusätzliches Lernmedium bereit zu stellen. Wenn 6 Schüler davon profitiert hätten, wäre das schon ganz gut. Inwie-

fern die Videotutorials diesen Schülern tatsächlich etwas gebracht haben, lässt sich schwer quantifizieren, allerdings schließe ich aus der Rückmeldung der Schüler sowie aus meinen eigenen Beobachtungen derselben, dass hier tatsächlich positive Entwicklungen statt gefunden haben, was sich in fortgeschritteneren und vor allem schneller korrigierten Lösungen bemerkbar macht.

Da dies meine ersten Erfahrungen mit Videotutorials waren, werde ich diese in der Folge kurz analysieren und Verbesserungsvorschläge erläutern:

1. Viele Schüler haben sich die Videos teilweise angeschaut, hatten aber nicht die Ausdauer sie zu Ende zu betrachten. Dies kann ich aus eigener Erfahrung gut nachvollziehen. Wenn ich mir meine eigenen Tutorials anschaue, dann erscheint mir die Dauer ebenfalls als zu lang. Die Onlinekurse, die ich absolviert habe, verwenden meist eine Vielzahl von Videos, die aber jeweils nur wenige bis maximal etwa zehn Minuten dauern und denen dann meistens eine Übungssitzung folgt. Dies begrenzt die Zeit des passiven Zuschauens und fördert die Festigung des Gelernten.
2. Die drei Videotutorials sind inhaltlich ziemlich anspruchsvoll. Dies war durchaus meine Absicht, da ich die praktische Lösung eines nicht trivialen Problems anhand von JavaScript erläutern wollte. Damit habe ich aber wahrscheinlich die schwächeren Schüler nicht erreicht. Daher betrachte ich es als sinnvoll, auch kürzere und inhaltlich begrenztere Videotutorials zu erstellen, die z.B. das Thema Schleifen oder Felder anschaulich erläutern und vorführen.

Insgesamt war das Feedback der Klasse positiv und ich sehe hier weiteres Potential, die Entwicklung einiger Schüler zu unterstützen.

4.1.9 Evaluation

Ich hatte der Klasse in der ersten Stunde die Bewertung erläutert. Meine damalige Vorstellung war, dass jeder Schüler im Laufe des Semesters das Spiel T1IF Invaders programmieren sollte, weshalb das Endprodukt von Anfang an auf der Webseite zur Verfügung stand.

Beim Vorführen des Spiels war die Begeisterung in der Klasse groß. Die Perspektive selbst etwas entwickeln zu können, was ihnen als Benutzer täglich Freude bereitete, war sichtlich motivierend. Diese Motivation wurde durch die datenbankgestützte Highscoreliste weiter gefördert, die es erlaubte, den Zusammenhang zum Server Side Scripting der Klasse T2IF herzustellen und somit zu illustrieren, wie alles zusammenhängt.

Das Spiel sollte sozusagen das Hauptproblem darstellen, dessen Lösung den Antrieb für die Erarbeitung der Lernziele liefern würde. Das Tutorial sollte dazu Schritt für Schritt die Vorgehensweise und die erforderlichen Grundkenntnisse anhand einer detaillierten Prozesserläuterung sowie kleinerer Problemstellungen mit steigendem Schwierigkeitsgrad entwickeln.

Am Ende des Semesters sollte dann jeder Schüler mir in einem Einzelgespräch seine Lösung für das Problem T1IF Invaders vorführen und detailliert erklären. Damit hätte ich meine Beobachtungen während des Semesters ergänzen können und mir somit einen genauen Überblick darüber verschaffen können, inwiefern der einzelne Schüler die Kompetenzen erreicht hätte. Des Weiteren hätte ein Copy/Paste Versuch wenig Aussicht auf Erfolg, da es insbesondere für Programmieranfänger erfahrungsgemäß so gut wie unmöglich ist, ein fremdes Programm überzeugend zu erklären.

Im Verlaufe des Semesters stellte sich dieser Plan jedoch als zu anspruchsvoll heraus. Somit beschloss ich, die Bewertung aufgrund einer hundert-minütigen Evaluierungsaufgabe für die obligatorische Kompetenz eins und einer zweiten hundert-minütigen Evaluierungsaufgabe für die selektiven Kompetenzen zwei und drei durchzuführen. Bei allen Evaluationen konnten Musterlösungen zu den behandelten Problemstellungen und Übungsaufgaben, Bücher, das Internet sowie eigene Notizen verwendet werden. Für Grenzfälle hatte ich bereits zu Anfang erklärt, dass ich die Arbeit der Schüler während des Semesters berücksichtigen würde.

4.1.9.1 *Formative Evaluation*

Um den Schülern eine kontinuierliche Rückmeldung zu geben, habe ich drei formative Evaluationen unter Prüfungsbedingungen, d.h. in Einzelarbeit, durchgeführt. Auch hierbei durften alle Hilfs- außer Kommunikationsmittel verwendet werden. Zusätzlich habe ich eine vierte formative Evaluation durchgeführt, die zuhause zu bearbeiten war.

Test 1 vom 19.10.12

Jeder Schüler hatte 50 Minuten, um eine von drei, von Herrn Fisch erstellten, einfachen Anwendungen (cf. 9.2.1.1) zu programmieren, die eine Benutzereingabe, eine Berechnung sowie eine Ausgabe beinhalteten. Erforderlich waren hierfür das Verständnis von Variablen, der arithmetischen JavaScript-Operatoren, der String-Konkatenation sowie der Aufruf der vordefinierten Methoden `window.prompt` sowie `document.write`. All dies war Gegenstand des ersten Beispiels im Tutorial, das sehr ausführlich bearbeitet worden war. Somit war das gute Ergebnis von durchschnittlich

6,2 von maximal 8 Punkten nicht verwunderlich, wobei dies zwei Nullergebnisse beinhaltete, von Schülern die nichts abgegeben hatten. Einzelgespräche mit diesen beiden führten sehr schnell zu Tage, dass sie noch nicht ernsthaft begonnen hatten, sich mit der Materie zu beschäftigen. Ich habe sie seitdem verstärkt angespornt, was zumindest bei einem zum Bestehen des Moduls beigetragen hat.

Test 2 vom 5.12.12

Innerhalb von 100 Minuten war eine Webseite (cf. Integrationsübung 1 im Trainingsteil der Webseite sowie 9.2.1.1) zu erstellen, welche die bisher in den Problemstellungen und Übungen behandelten Konzepte integriert. Dies umfasst die meisten Indikatoren der Kompetenz 1: Integrieren von JavaScript-Code beim Erstellen von CSS-formatierten HTML-Seiten, Anwenden von grundlegenden Sprachelementen, Anwenden von vordefinierten Funktionen, Erstellen von einfachen benutzerdefinierten Funktionen ohne Parameter, Benutzen der vordefinierten JavaScript-Objekte, Reagieren auf Maus-Ereignisse sowie Zugreifen und Ändern der DOM-Elemente.

Hier wurde jedoch auch implizit die selektive Kompetenz zum selbstständigen Nachschlagen und Einsetzen von Informationen evaluiert. So konnte die Passwortabfrage, wie sie im Beispiel zur Verschachtelung von Kontrollstrukturen im Trainingsteil kommentiert und mit der Möglichkeit zum direkten Ausprobieren mittels Knopfdruck zur Verfügung stand, eins zu eins übernommen werden. Dies wurde jedoch nur von einem einzigen Schüler entdeckt! Ich betrachte dies als Indiz dafür, dass es nicht einfach ist, trotz großem Aufwand meinerseits, wichtige Konzepte so anschaulich und verständlich zu präsentieren, dass sie die Schüler erreichen. Allerdings haben insgesamt 13 Schüler die Passwortabfrage korrekt gelöst, in vielen Fällen mittels Anpassung von Skriptteilen aus dem Internet.

Die Programmierung der Ballbewegung war für die Klasse nicht neu, bis auf die Verwendung der Breite und Höhe des `div`-Elementes anstatt fest vorgegebener Dimensionen. Die dynamische Randbehandlung richtig zu programmieren gelang nur einem Schüler.

Der Einsatz eines Timers für die Animation war bekannt und wurde von 16 der 19 Schüler (2 fehlten) entsprechend gut gelöst.

Die korrekte Verwendung der Methode `Math.random` zur Berechnung von Zufallszahlen gelang nur 7 Schülern, obwohl auch hier die zu verwendende Formel in einem ausführbaren Beispiel kommentiert im Trainingsteil zur Verfügung stand. Allerdings waren die meisten Schüler noch nicht bei

diesen Übungen angelangt. Die Methode war jedoch im Unterricht im Rahmen der Vorstellung von Problem 7 ausführlich besprochen und vorgeführt worden.

Dieser Test war eine recht anspruchsvolle Problemstellung, angelehnt an Problem 6. Mir wurde erst nach der Hälfte der Zeit bewusst, dass die Schüler, aufgrund des Prüfungslogins, keinen Zugriff auf die Musterlösungen der behandelten Problemstellungen hatten und dass die Mehrheit ohne diese Inspirationsquelle nicht in der Lage war, die Problemstellung in der gegebenen Zeit zu lösen. Ich stellte dann die Musterlösungen zur Verfügung. Die Lösungen von 15 der 19 Schüler waren quasi identisch mit der Musterlösung zu Problem 6. Dies zeigte, dass viele Schüler die Musterlösung nicht wirklich verstanden hatten und an die gegebene Problemstellung anpassen konnten.

Dieser Test bestätigte, was ich bereits zuvor festgestellt hatte, nämlich, dass ich die Mehrheit der Klasse mit den Problemen überfordert hatte und dass sie mehr Zeit und Übung benötigten, um sich die Materie zu erarbeiten.

Test 3 vom 19.12.12

Für diesen Test (cf. Integrationsübung 2 im Trainingsteil der Webseite sowie 9.2.1.1) standen ebenfalls 100 Minuten zur Verfügung. Es geht darum, einen Taschenrechner zu entwickeln. Die Lösung erfordert alle Indikatoren der obligatorischen Kompetenz 1 außer der Verschachtlung von Kontrollstrukturen, Feldern und Tastaturereignissen. 6 Schüler haben eine Lösung abgegeben, die ausreichen würde, die Kompetenz 1 zu bestehen. 5 weitere bewegten sich im Grenzbereich, der Rest hätte die Kompetenz 1 eindeutig nicht erreicht. Erstaunlicherweise bereitete die Passwortabfrage einigen Schülern noch immer Probleme. Als größte Schwierigkeit erwies sich jedoch der Zugriff auf und die Veränderung des Inhalts der Eingabefelder. Einige Schüler glaubten, wenn sie den Wert eines Eingabefeldes in einer Variablen speicherten, dass eine Änderung des Variableninhalts dann automatisch auch den Inhalt des Eingabefeldes ändern würde. Auch dieser Test zeigte mir, dass es noch einige Wissenslücken zu füllen gab.

Test 4 vom 16.1.13 (Hausarbeit)

Es handelt sich um die Beispielevaluationsaufgabe des offiziellen CLISS1-Lehrprogramms, wo es darum geht, einen Währungsrechner zu entwickeln (cf. Beispielevaluationsaufgabe 1 der Webseite, sowie 9.2.1.1). Neu waren in dieser Aufgabe der Einsatz von Feldern und Auswahllisten. Lösungen erhielt ich lediglich von 6 Schülern, wobei 1 Lösung gut, 4 gerade noch ausreichend und 1 nicht

ausreichend zur Bestätigung der obligatorischen Kompetenz 1 waren. Hauptproblempunkt war der Umgang mit Feldern. Dies erklärte und illustrierte ich in der nächsten Stunde erneut im Detail.

4.1.9.2 *Summative Evaluation*

Aufgrund der summativen Evaluation, die im Folgenden detailliert wird, haben 15 Schüler das Modul bestanden, davon einer mit der Auszeichnung „sehr gut“. 6 Schüler haben nicht bestanden.

Kompetenz 1 Aufgabe 1 am 23.1.13

Diese Aufgabe (cf. Evaluation Kompetenz 1 Aufgabe 1 der Webseite, sowie 9.2.1.2) umfasst alle Indikatoren der obligatorischen Kompetenz 1. Sie kombiniert Aspekte aus verschiedenen Problemstellungen und Übungen, die im Unterricht ausführlich behandelt wurden. Um es den Schülern zu ermöglichen, sich während der begrenzten verfügbaren Zeit auf die wichtigen Dinge zu konzentrieren, habe ich ein HTML- und CSS-Gerüst zur Verfügung gestellt, in dem alle Elemente bereits vorhanden waren. Somit blieb „nur“ noch die Funktionalität zu programmieren.

Die Aufgabe beinhaltete das Bewegen eines Raumschiffes mittels Mausklick auf vier Pfeiltasten, wobei die Schrittweite aus einer mit Zufallswerten gefüllten Auswahlliste ausgewählt werden kann.

Als Hauptschwierigkeiten erwiesen sich folgende Punkte:

- Die korrekte Anwendung der Methode `Math.random`. Hier galt es die im Kurs behandelte und auf der Kurswebseite vorhandene und dokumentierte Formel anzuwenden. Dieser Indikator wurde insgesamt nur zu 56% erfüllt.
- Das Verschachteln von Kontrollstrukturen. Das Füllen der Auswahlliste für die Schrittweite der Bewegung des Raumschiffs erforderte eine Auswahl innerhalb einer Wiederholung. Dieser Schritt wurde von neun Schülern gar nicht gelöst. Hierzu hat aber wahrscheinlich auch das Problem beigetragen, dass einige Schüler nicht mehr wussten, dass man mittels dem Modulo-Operator herausfinden kann, ob eine Zahl durch eine andere teilbar ist.
- Das Zugreifen und ändern der DOM-Elemente, in diesem Fall also das Bewegen des Raumschiffes, war eine Standardproblemstellung, wie sie alle Schüler vielfach bewältigt haben. Der Grund, weshalb hier insgesamt trotzdem nur 50% erreicht wurden, liegt darin, dass bis auf eine Ausnahme niemand die Randbehandlung richtig programmiert hatte. Gegenüber den meisten behandelten Problemstellungen war hier die Vorgabe, dass falls ein ganzer

Schritt aus Platzmangel nicht ausgeführt werden konnte, der maximal mögliche Schritt ausgeführt werden sollte, mit anderen Worten das Raumschiff an den entsprechenden Rand gesetzt werden sollte.

18 Schüler haben an dieser Evaluation teilgenommen. Einer hatte die obfuskierte Musterlösung entschlüsselt und sich damit disqualifiziert. In der Folge habe ich daher, anstatt obfuskierte Musterlösungen, Videos erzeugt, um es den Schülern zu ermöglichen, sich das Endprodukt anzuschauen. Dies betrachte ich, angesichts der Schwierigkeiten die viele Schüler mit dem Verständnis auch noch so sorgfältig erarbeiteter Angaben haben, als sehr wichtig, Von den verbleibenden 17 Schülern hatten 6 85% oder mehr der Indikatoren erfüllt (cf. 9.2.1.2), 5 weitere hatten mindestens 70% erreicht.

Kompetenz 1 Aufgabe 2 am 28.1.13

Diese Aufgabe (cf. Evaluation Kompetenz 1 Aufgabe 2 der Webseite, sowie 9.2.1.1) war für die Schüler, die bei der ersten Evaluationsaufgabe gefehlt hatten oder nicht die erforderlichen 70% für die obligatorische Kompetenz 1 erreicht hatten. Der Schwierigkeitsgrad entsprach der ersten Evaluationsaufgabe, allerdings war hier keine verschachtelte Kontrollstruktur erforderlich.

Als Hauptschwierigkeiten erwiesen sich folgende Punkte:

- Anwenden von elementaren Kontrollstrukturen, in diesem Fall die Schleife zum Füllen des Feldes mit den Positionen des Bildes. Dies bereitete fünf der sechs Schüler Schwierigkeiten.
- Das Verwenden eines Feldes bereitete ebenfalls fünf von sechs Probleme.
- Der Zugriff auf die DOM-Elemente wurde von niemandem korrekt gelöst.

4 Schüler erzielten mindestens 70%.

Kompetenz 1 Aufgabe 3 am 29.1.13

Diese Aufgabe (cf. Evaluation Kompetenz 1 Aufgabe 3 der Webseite, sowie 9.2.1.1) war für einen Schüler, der bei der ersten Aufgabe gefehlt und bei der zweiten seine Arbeit falsch abgespeichert hatte, so dass sie nicht mehr zugreifbar war. Er hatte allerdings versäumt mir mitzuteilen, dass er nach nur 50 Minuten wegen einem Arztbesuch gehen musste. Diese Zeit reichte natürlich bei weitem nicht aus, um die Aufgabe zu lösen.

Kompetenz 1 Aufgabe 4 am 31.1.13

Diese Aufgabe (cf. Beispielevaluationsaufgabe 2 der Webseite, sowie 9.2.1.1) war die letzte Chance für den verbleibenden Schüler. Jedoch wurde hier klar, dass er, trotz 100 Minuten Zeit, überfordert war und somit das Modul nicht bestanden hatte.

Kompetenzen 2 und 3 Aufgabe 1 am 30.1.13

Diese Aufgabe (cf. Evaluation Kompetenzen 2 und 3 Aufgabe 1 der Webseite, sowie 9.2.1.1) erforderte die Beherrschung von 2-dimensionalen Feldern, das Optimieren eines vorgegebenen sowie das Debuggen eines fehlerhaften Skripts.

Es gelang lediglich einem Schüler, diese Aufgabe erfolgreich zu meistern. Dieses ernüchternde Ergebnis kam aufgrund der zu kurzen Vorbereitungszeit für diese Kompetenzen nicht überraschend.

Kompetenzen 2 und 3 Aufgabe 2 am 1.2.13

Diese Aufgabe (cf. Evaluation Kompetenzen 2 und 3 Aufgabe 2 der Webseite, sowie 9.2.1.1) entsprach vom Schwierigkeitsgrad weitgehend der vorherigen. Ich hatte sie für einen Schüler erstellt, der bei der vorherigen Evaluation der Kompetenzen 2 und 3 krank war. Aufgrund der Ergebnisse seiner Klassenkameraden bevorzugte er es, die Lösung der Aufgabe erst gar nicht zu versuchen.

4.1.10 Schlussfolgerungen

Ich hatte das erste Semester mit dem Ziel begonnen, PBL konsequent umzusetzen in der Erwartung, hiermit eine höhere intrinsische Motivation und bessere Lernerfolge zu erzielen.

Was die Lernerfolge betrifft, bin ich zufrieden, insbesondere auch im Vergleich zu den anderen Klassen (cf. 5.1.3). Die gestellten Evaluationsaufgaben waren recht anspruchsvoll und es hat sich gezeigt, dass zumindest die obligatorische Kompetenz von der Mehrheit der Klasse erreicht wurde.

Gelernt habe ich, dass PBL, insbesondere bei jungen Schülern mit so gut wie keiner Programmiererfahrung (abgesehen von Scratch auf der neunten Klasse), sehr behutsam eingeführt werden sollte. Meine Vorstellung, dass interessante Problemstellungen die Schüler so stark anspornen würden, dass sie sich die Materie, mit meiner methodischen und prozessualen Unterstützung, selbst aneignen könnten, erwies sich bis auf wenige Ausnahmen als unrealistisch. Dieser Ansatz setzt ein kontinuierliches Arbeiten auch außerhalb der Unterrichtsstunden voraus, was jedoch größtenteils nicht stattgefunden hat. Hier besteht zweifellos auch ein Zusammenhang zum zu schnell ansteigenden

Schwierigkeitsgrad der Problemstellungen, was mir die Klasse bestätigt hat (cf. 5.1.1).

Auch wenn die Schülerbefragung (cf. 5.1.1) dies nicht widerspiegelt, bin ich aufgrund meiner Beobachtungen während des Semesters und den Aussagen der Schüler überzeugt, dass die Problemstellungen die Schüler interessiert und motiviert haben.

Meine Erfahrungen aus dem ersten Semester zeigen, dass Schüler, zumindest in diesem Alter, größtenteils noch nicht selbstständig genug sind um sich anspruchsvolle Materie selbst zu erarbeiten. Es war wohl auch nicht realistisch zu erwarten, dass Schüler, die es gewohnt sind die Materie von der Lehrkraft in kleinen Stücken erklärt zu bekommen, sich innerhalb kurzer Zeit die Prozesse aneignen, um dies selbstständig zu tun.

4.2 *Zweites Semester*

Mein Ziel im zweiten Semester war es, auf den im ersten Semester gelegten Grundlagen aufzubauen, um die Problemlösekompetenzen der Klasse weiter voranzubringen. Dieses Ziel passte in meinen Augen sehr gut zum Lehrplan für das Modul CLISS2, der folgende Kompetenzen vorsieht:

1. Modularen Quellcode erstellen und in Webseiten einbinden (obligatorisch).
2. Erstellen von interaktiven Webseiten (obligatorisch).
3. Einbinden von vorgefertigten Codelösungen in der eigenen Webseite (selektiv).
4. Selbstständig und motiviert arbeiten (selektiv).

Der Fokus des Lehrplans liegt in der Integration der erarbeiteten Konzepte zur Lösung größerer Problemstellungen.

Um das Ziel zu erreichen teilte ich das zweite Semester in drei Teile auf:

1. Ausblick, kurze Wiederholung und Einführung von Schlüsselkonzepten.
2. Entwicklung von T1IF Invaders.
3. Abschlussevaluation anhand einer selbsterstellten Problemstellung.

4.2.1 Ausblick, kurze Wiederholung und Einführung von Schlüsselkonzepten

Funktionen stellen eine klassische Umsetzung der „Teile und Herrsche“-Strategie dar, die es ermög-

licht, komplexe Problemstellungen zu lösen, indem man sie in einfachere Teilprobleme zerlegt und jedes davon einzeln löst. Erstellen und anwenden von Funktionen ist daher einer der Hauptpunkte des Lehrplans für das Modul CLISS2. Im ersten Semester wurden vordefinierte Funktionen mit und ohne Parameter und/oder Rückgabewert eingesetzt und eigene parameterlose Funktionen mit und ohne Rückgabewert erstellt.

In der ersten Stunde des zweiten Semesters erläuterte ich den CLISS2 Lehrplan und das Evaluierungsraster (cf. 9.4.2). Diese beiden Dokumente sind wie die von CLISS1 ganz oben im Trainingsteil zu erreichen. Anschließend erklärte ich den geplanten Evaluationsablauf. Als erste große Problemstellung sollte das Spiel T1IF Invaders in Angriff genommen werden. Als Abschlussevaluation sollte dann jeder Schüler seine eigene Problemstellung entwickeln, mit der einzigen Vorgabe meinerseits, dass die in den zwei Modulen behandelten Schwerpunkte eingesetzt werden sollten.

Anschließend wiederholte ich die Grundlagen und Anwendungsgebiete von Funktionen mit und ohne Parameter/Rückgabewert. Hierzu hatte ich einen neuen Tutorialteil mit Erklärungen und Beispielen sowie fünf Übungen erstellt:

Übung	Lösung

31	`function greetUser() {`
	`  result = "Hello " + user;`
	`}`

32	`function getUserGreeting() {`
	`  return "Hello " + user;`
	`}`

33	`function greetUser(user) {`
	`  result = "Hello " + user;`
	`}`

34	`function getUserGreeting(user) {`
	`  return "Hello " + user;`
	`}`

35	`function getSum(a, b) {`
	`  return a + b + c;`
	`}`
	
	`function getProduct(a, b) {`
	`  return a * b * c;`
	`}`
	
	`function getSumProduct(a, b, c) {`
	`  return getSum(getProduct(a, b), getProduct(b, c));`
	`}`

Die Übungen wurden durchwegs korrekt gelöst.

Die Validierung der Dateneingabe in Formularen, DOM-Modifikationen und das Verändern von CSS-Eigenschaften mittels JavaScript waren bereits im ersten Semester vielfach behandelt und ein-

gesetzt worden, so dass ich dies nur kurz wiederholte.

Ich ging dann auf das Erarbeiten eines Gesamtkonzeptes zum Aussehen und Verhalten einer Webseite ein. Dies war zum Teil eine Wiederholung und zum Teil eine Erweiterung der im Modul zur Erstellung von statischen Webseiten behandelten und verwendeten Aspekte. Hier hob ich die Wichtigkeit einer genauen Planung vor Beginn jeglicher Programmieraktivitäten hervor, indem das Zielpublikum, die gewünschte Funktionalität, das Layout und die zu verwendenden Techniken und Konzepte bestimmt werden. Eine Beispielkonzeption sollte dann später anhand von T1IF Invaders illustriert werden.

Das Anwenden der Gestaltungsregeln von HTML und CSS war aus dem Vorjahr bekannt und wurde routinemäßig eingesetzt.

Unter der Rubrik „Erstellen von komplexerem JavaScript-Code“ führte ich das `canvas`-Tag ein. Zwar hätte das Spiel T1IF Invaders auch ohne dieses Tag, mittels DOM-Modifikationen und Veränderungen von CSS-Eigenschaften, implementiert werden können, jedoch sah ich in der Einführung dieses Tags mehrere Vorteile:

1. Die Schüler mussten sich mit einem neuen Element und seinen Methoden auseinandersetzen, welche ansprechende graphische Anwendungsmöglichkeiten eröffneten und somit die Motivation positiv beeinflussten. Dies insbesondere angesichts der Tatsache, dass immer mehr Spiele dieses Element benutzen, worauf ich die Klasse bereits mehrfach aufmerksam gemacht hatte.
2. Es bot die Gelegenheit, das bisher Gelernte in einem etwas anderen Kontext anzuwenden und somit das bisher Erarbeitete auf eine höhere Abstraktions- und Verständnisebene zu bringen und die Transferkompetenz zu entwickeln.
3. Die resultierende Lösung ist in meinen Augen eleganter und sauberer als die Bewegung von Bildern mittels Verändern der CSS-Eigenschaften.

Ich erklärte das `canvas`-Tag anhand des Tutorialbeispiels.

Die Kompetenzen drei und vier bedurften keiner größeren Erläuterungen.

4.2.2 T1IF Invaders

4.2.2.1 *Zeitplan*

Die Entwicklung des Spiels begann am 20.2.13. Ich hatte als Zeitplan den 5.4.13 für die Abgabe der wichtigsten Funktionen (`init`, `moveAliens`, `moveShotsAndBombs`, `checkCollisions`, `handleKeyDown` und `handleKeyUp`) und den 19.4.13 für die Abgabe des gesamten Spiels vorgegeben.

4.2.2.2 *Problembeschreibung*

Um die Erarbeitung eines Gesamtkonzeptes sowie dessen Umsetzung für eine anspruchsvolle Problemstellung vorzuführen, hatte ich eine detaillierte Problembeschreibung entwickelt. Diese erklärte Ziel, Verlauf und Steuerung des Spiels und illustrierte das Layout anhand eines Videos. Dies war den Schülern oberflächlich bereits vom Spielen des Spiels, das seit Beginn des Jahres auf der Webseite zur Verfügung stand, bekannt. Die Details kannten sie jedoch noch nicht.

Die Beschreibung ging auf die Techniken der Umsetzung mittels dem `canvas`-Tag und insbesondere auf die Kollisionsüberprüfung ein, deren Schwierigkeit ich mir bewusst war. Deshalb illustrierte ich anhand einer Grafik, was die Überlappung von zwei Objekten bedeutet und wie man sie feststellen kann. Ich erklärte dies in den folgenden Stunden mehrere Male an der Tafel.

Im Umsetzungsteil wurden die Planung und Implementierung sehr detailliert beschrieben, da die Mehrheit der Schüler noch keine Erfahrung mit Problemstellungen dieser Größenordnung hatte und es mir darum ging, ihnen vorzuführen, wie man ein großes Problem in Teilprobleme zerlegen und diese einzeln lösen kann.

4.2.2.3 *HTML- und CSS-Struktur*

Als erster Schritt stand die Erstellung der HTML-Datei auf dem Programm, deren vier Bereiche ich genau beschrieben hatte. Es ging hier vor allem darum, das Hauptmenü, das Spielfeld, die Highscoretabelle sowie die Anzeige der Tastenerklärungen mittels HTML zu strukturieren und mittels CSS zu stylen:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF Invaders v1.0</title>
        <meta charset="UTF-8">
```

```
        <style>
            body {background-color: black; color: gold;}
            #splash {display:block; width: 200px; margin: auto;}
            #splash > ul {list-style-type: none;}
            #highScoreSection, #keysSection, #board {display: none;}
            table {border-spacing: 0;}
            table caption {font-size: 2em; font-weight: bold;}
            table thead {text-align: left;}
            table thead th {background-color: #CC2222;}
            th, td {padding: 5px;}
            tr:nth-of-type(odd) {background-color: #222222;}
            tr:nth-of-type(even) {background-color: #444444;}
        </style>
    </head>
    <body>
        <section id="splash">
            <h2>T1IF Invaders v1.0</h2>
            <ul>
                <li><button id="newGame" onclick="newGame();">New Game</button></li>
                <li><button id="highScores" onclick="displayHighScores();">High Scores</button></li>
                <li><button id="keys" onclick="displayKeys();">Keys</button></li>
            </ul>
        </section>
        <section id="highScoreSection">
            <table id="highScoreListTable">
                <caption>Hall of Fame</caption>
                <thead>
                    <tr>
                        <th>Rank</th>
                        <th>Player</th>
                        <th>Score</th>
                        <th>Level</th>
                    </tr>
                </thead>
                <tr>
                    <td>1</td>
                    <td>Gilles Everling</td>
                    <td>2300</td>
                    <td>56</td>
                </tr>
                <tr>
                    <td>2</td>
                    <td>Gilles Everling</td>
                    <td>2290</td>
                    <td>56</td>
                </tr>
                <tr>
                    <td>3</td>
                    <td>Gilles Everling</td>
                    <td>2280</td>
                    <td>56</td>
                </tr>
            </table>
            <button onclick="hideHighScores();">OK</button>
        </section>
        <section id="keysSection">
            <p>Use left and right cursor keys to move your spaceship and the space bar to fire.
               The game can be paused at any time by pressing <code>P</code>.
               Pressing <code>P</code> again will resume.
```

```
            You can exit the game early using <code>Esc</code>.</p>
        <button onclick="hideKeys();">OK</button>
    </section>
    <section id="board">
        <p>
            Score: <span id="score"></span> 
            Level: <span id="level"></span>
        </p>
        <canvas id="canvas" width="600" height="400">This browser does not run this game
           (canvas support missing).</canvas>
        <section>
            <img id="life1" src="playerspaceship36x46.png" alt="playerspaceship36x46.png">
            <img id="life2" src="playerspaceship36x46.png" alt="playerspaceship36x46.png">
        </section>
    </section>
    <script src="index.js"></script>
</body>
</html>
```

Am 22.2.13 hatten 3 Schüler dies vollendet. Bei einigen anderen wurde jedoch ersichtlich, dass das Layout ihnen Schwierigkeiten bereitete. Es gelang ihnen nicht, die drei Schaltflächen im Hauptmenü zu zentrieren. Auch das Styling der Highscoretabelle mit abwechselnden Farben führte zu intensivem Recherchieren. Nach einiger Zeit zeigte ich meine Lösung am Schirm, damit alle mit der eigentlichen Programmierung des Spiels beginnen konnten und nicht zu viel Zeit mit dem Styling verloren.

Meine Hauptaktivität bestand darin, als Tutor denjenigen, die Fragen hatten, zur Seite zu stehen. Aufgrund der Klassengröße und der Zahl der Fragen beauftragte ich die besten Schüler der Klasse, ihre Kollegen bei Bedarf zu unterstützen. Dies funktionierte zusehends besser. Man merkte, dass die meisten Schüler ihre informelle Gruppe gefunden hatten, innerhalb derer sie die Probleme analysierten und die Lösungen besprachen. Der Großteil der Klasse arbeitete sehr aktiv und motiviert, was eine positive Atmosphäre erzeugte.

4.2.2.4 *Globale Variablen*

Als Erstes sollten die globalen Variablen deklariert werden, um die Bilder, die X- und Y-Koordinaten des Raumschiffs, der Aliens, der Kugeln und der Bomben, den aktuellen Score und Level usw. zu speichern. Ich hatte den Zweck aller von mir verwendeten globalen Variablen aufgelistet, ohne jedoch die Datentypen anzugeben. Dies erforderte einiges Nachdenken seitens der Schüler und stellte eine ausgezeichnete Gelegenheit dar, sich im Kontext eines konkreten Problems über die Einsatzgebiete, insbesondere von Feldern, Gedanken zu machen.

```
// Declaration of global variables
var canvas = document.getElementById('canvas');
```

```
var context = canvas.getContext('2d');
var imagePlayer = new Image();
var imageShot = new Image();
var imageAlien = new Image();
var imageAlienBomb = new Image();
imagePlayer.src = 'playerspaceship36x46.png';
imagePlayer.width = 36;
imagePlayer.height = 46;
imageShot.src = 'shot12x23.png';
imageShot.width = 12;
imageShot.height = 23;
imageAlien.src = 'alien42x27.png';
imageAlien.width = 42;
imageAlien.height = 27;
imageAlienBomb.src = 'alienbomb10x10.png';
imageAlienBomb.width = 10;
imageAlienBomb.height = 10;
var playerShotsX = [], playerShotsY = [], minFireThreshold = 300, initialPlayerSpeed = 20,
initialNumOfLives = 2;
var aliensX = [], aliensY = [], alienBombsX = [], alienBombsY = [], numAliens = 40, numAliensPerRow
= 10;
var alienDirection, fireButton, moveLeft, moveRight, timeOfLastShot, currPlayerX, currPlayerY;
var alienXSpeed, alienYSpeed, leftBorderTouched, loseLife, numOfLives, alienBombSpeed, score;
var currLevel, currPlayerSpeed, numShots, pauseButton, adjustmentFactor = 0.2, timeOut = 1000 / 60;
```

4.2.2.5 *Funktionen*

Als nächstes sollten die Funktionen in der gegebenen Reihenfolge implementiert werden. Für alle Funktionen hatte ich auf hohem Abstraktionsniveau angegeben, was sie vollbringen sollten. Die Herausforderung für die Schüler bestand also darin, die abstrakten Anweisungen in konkretes JavaScript umzusetzen.

init

Die erste Funktion dient zur Initialisierung zu Beginn eines neuen Spiels, eines neuen Levels oder nachdem das Raumschiff zerstört wurde und der Spieler noch über mindestens ein Ersatzraumschiff verfügt:

```
/* Called by newGame.
 * Draws aliens and adjusts global variables. */
function init() {
    var x = 1, y = 1;
    for (var i = 0; i < numAliens; i++) {
        aliensX[i] = x;
        aliensY[i] = y;
        context.drawImage(imageAlien, aliensX[i], aliensY[i]);
        x += imageAlien.width + 10;
        if (i % numAliensPerRow == 9) { // 10 aliens per row
            x = 1;
            y += imageAlien.height + 7;
        }
```

```
    }
    document.getElementById("score").innerHTML = score;
    document.getElementById("level").innerHTML = currLevel;
    alienXSpeed = (5 + currLevel) * adjustmentFactor;
    alienYSpeed = currLevel;
    alienBombSpeed = Math.max(1, Math.floor(alienYSpeed / 3));
    leftBorderTouched = false;
    alienDirection = 1; // aliens start moving to the right
    moveLeft = false; // no keys pressed
    moveRight = false;
    playerShotsX = []; // delete shots
    playerShotsY = [];
    alienBombsX = []; // delete bombs
    alienBombsY = [];
}
```

Am 1.3.13 hatten die meisten Schüler ihren ersten Außerirdischen mittels der drawImage-Funktion auf den Canvas gemalt, was sichtbare Erfolgsgefühle auslöste. Dies war eine erfreuliche Leistung, allerdings half hierbei, dass ich im Tutorial ein kommentiertes und ausführbares Anwendungsbeipiel zur Verfügung gestellt hatte. Einige Schüler verwendeten auch die w3schools Webseite.

Deutlich schwieriger wurde es, als es darum ging, die vierzig Aliens auf ihre Anfangspositionen zu malen, da ich ausdrücklich darauf hingewiesen hatte, dass hierfür ein intelligenter Ansatz und nicht einfach vierzig sequentielle Zeichenoperationen zu programmieren wären. Die Ausgangsposition besteht aus vier Reihen mit jeweils zehn Aliens. Das heißt, nachdem zehn Aliens an der gleichen vertikalen, aber unterschiedlichen horizontalen Positionen gemalt wurden, muss die horizontale Position wieder an den linken Rand gesetzt und die vertikale Position erhöht werden. Den sieben schwächsten Schülern habe ich in mehreren Sitzungen das Problem im Detail erklärt und zum Teil vorgelöst, so dass sie die Lösung nur noch vervollständigen mussten. Am 12.3.13 hatten alle die Aliens in ihrer Ausgangsposition auf dem Schirm.

moveAliens

Als nächstes folgte die Funktion `moveAliens`. Die Detektion von Rändern bei der Bewegung von Bildern war der Klasse aus einer Vielzahl von Problemstellungen bekannt. Hier kam jedoch hinzu, dass die Aliens bei der Berührung des linken Randes eine gewisse Distanz nach unten bewegt werden sollten. Um dies zu erreichen, war es notwendig, sich die Berührung des linken Randes mittels einer Variable zu merken und in der Folge die vertikale Position aller Aliens zu erhöhen, sowie die horizontale Position auf den Anfangswert zurückzusetzen:

```
// Called by gameLoop
function moveAliens() {
    // First we determine the minimum and maximum horizontal alien positions.
```

```
    var minX = canvas.width, maxX = 0;
    for (var i = 0; i < aliensX.length; i++) {
        if (aliensX[i] < minX) minX = aliensX[i];
        if (aliensX[i] > maxX) maxX = aliensX[i];
    }
    // We need those to determine whether there is enough room to continue in the current direction.
    // If there isn't enough room, we need to change direction.
    if (minX <= alienXSpeed && alienDirection == -1 || maxX + imageAlien.width > canvas.width &&
         alienDirection == 1) {
                alienDirection = -alienDirection;
                // If we touched the left border, we need to move the aliens down.
                if (alienDirection == 1) leftBorderTouched = true;
    }
    if (leftBorderTouched) {
        leftBorderTouched = false;
        alienYSpeed += 2 * adjustmentFactor;
        for (i = 0; i < aliensY.length; i++) aliensY[i] += alienYSpeed;
    }
    // Move aliens horizontally.
    for (i = 0; i < aliensX.length; i++) {
        aliensX[i] += alienDirection * alienXSpeed;
        context.drawImage(imageAlien, aliensX[i], aliensY[i]);
    }
}
```

Die Mehrheit der Klasse löste dies selbstständig oder in der Diskussion mit anderen. Bei sieben Schülern half ich bei der Lösung, indem ich mittels Fragen versuchte, das Problem zu verdeutlichen. Dies klappte bei fünf, den anderen zwei löste ich einen Teil der Funktion vor. Diese hatten das erste Modul nicht bestanden und waren hier eindeutig überfordert. Ich nutzte die Gelegenheit, um mittels gezielter Fragen, ihre Wissens- und Verständnislücken zu bestimmen und versuchte diese zu reduzieren.

Am 22.3.13, dem letzten Tag vor den zweiwöchigen Ferien, hatten fast alle die Funktionen `init` und `moveAliens` fertig gestellt und ich gab den Zeitplan für die Abgabe des Spiels bekannt. Wie zu Beginn des Kapitels erwähnt, waren die wichtigsten Funktionen am 5.4.13 abzugeben. Ich erhielt jedoch nur von sechs Schülern überhaupt etwas und es stellte sich heraus, dass niemand die Kollisionsüberprüfung gelöst hatte. Ich war mir der Schwierigkeit der Kollisionsüberprüfung sehr wohl bewusst, allerdings hatte ich mal wieder geglaubt, dass die Schüler, aufgrund meiner Erklärungen an der Tafel sowie der algorithmischen Beschreibung und grafischen Illustration im Tutorial, in der Lage sein würden, eine Lösung zu finden. Es zeigte sich jedoch, dass die meisten die Kollisionsüberprüfung noch nicht begonnen hatten, da sie die Bewegung der Kugeln und Bomben mittels der Funktion `moveShotsAndBombs` noch nicht fertig gestellt hatten.

Aufgrund der begrenzten verbleibenden Zeit und um alle wieder in die gleiche Ausgangslage zu bringen, stellte ich der Klasse am 9.4.13 einen Teil der Musterlösung zur Verfügung. Dieser enthielt

die Deklaration der globalen Variablen sowie die vollständigen Funktionen `init`, `moveAliens`, `moveShotsAndBombs`, `handleKeyDown`, `handleKeyUp`, `newGame`, `gameOver`, `restartLevel`, `displayHighScores`, `hideHighScores`, `showSplash`, `displayKeys` und `hideKeys`. Des Weiteren war die Funktion `gameLoop` vorhanden, da ich inzwischen nicht mehr daran glaubte, dass diese Funktion, aufgrund ihrer hohen Komplexität, innerhalb der sehr begrenzten Zeit von einer Mehrheit der Schüler umgesetzt werden könnte.

gameLoop

Die Funktion bildet das Herzstück des Spiels und wird mittels eines Timers sechzig mal in der Sekunde aufgerufen. Sie erfüllt folgende Aufgaben:

1. Falls die P-Taste einmal gedrückt wurde tut die Funktion nichts, da das Spiel im Pausezustand ist.
2. In der Originalversion misst die Funktion die seit dem letzten Aufruf verstrichene Zeit, um eventuelle Verzögerungen zu berücksichtigen und so die Rate von sechzig Aufrufen pro Sekunde möglichst genau einzuhalten. In der Praxis halten sich die hierdurch erzielten Vorteile bezüglich Spielbarkeit in Grenzen. Daher beschloss ich, zwecks Vereinfachung der ohnehin recht komplexen Funktion, diesen Teil der Klasse kurz zu erklären, aber im Code wegzulassen.
3. Der Canvas wird gelöscht.
4. Falls die Leertaste gerade gedrückt ist und eine Mindestzeit seit dem letzten Schuss verstrichen ist, wird eine neue Kugel erzeugt und gemalt. Allerdings hatte ich den Teil, der in Abhängigkeit des aktuellen Levels entscheidet, ob eine, zwei oder drei Kugeln beim Druck der Leertaste abgefeuert werden sollen, entfernt, um den Schülern Gelegenheit zu geben, zu beweisen, dass sie das Prinzip verstanden hatten und mit Anpassungen einsetzen konnten. Die aktuelle Zeit muss gespeichert werden, damit beim nächsten Durchlauf überprüft werden kann, ob ein erneuter Schuss möglich ist.
5. Abhängig vom Wert der globalen Variablen für die rechte und linke Pfeiltaste, die von den Tastaturereignishandlern gesetzt werden, wird das Raumschiff nach links oder rechts bewegt, wobei hier natürlich die Ränder überprüft werden müssen.
6. Für jeden Alien wird aufgrund der in der Problembeschreibung angegebenen Formel mittels

Zufallszahl berechnet, ob er eine Bombe erzeugt. Ist dies der Fall, werden die Koordinaten der Bombe in die entsprechenden Felder eingefügt und die Bombe wird gemalt. Diese Generierung der Bomben hatte ich auch entfernt, da ich erwartete, dass die Schüler dies lösen könnten.

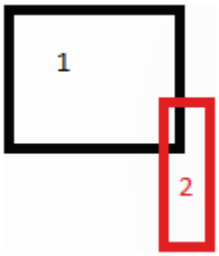
7. Die Funktion `checkCollisions` wird aufgerufen. Anschließend wird mittels der globalen Variable `loseLife` geprüft, ob das Raumschiff zerstört wurde. Ist dies der Fall wird die Zahl der verbleibenden Raumschiffe um eines verringert. Aufgrund der verbleibenden Anzahl von Raumschiffen wird dann entschieden, ob das Spiel mittels der Funktion `gameOver` beendet oder mittels `restartLevel` der aktuelle Level neu gestartet wird.

```
// This is the function that controls the game. Called by newGame and requestAnimationFrame.
function gameLoop(currTime) {
    // if paused, do nothing.  //setTimeout("gameLoop()", timeOut);
    if (pauseButton) requestAnimationFrame(gameLoop);
    else {
        var timeElapsed = currTime - lastAnimationTime;
        if (lastAnimationTime === 0) adjustmentFactor = 0.2;
        else adjustmentFactor = timeElapsed / 100;
        context.clearRect(0, 0, canvas.width, canvas.height);
        // If space key pressed and enough time since the last shot has elapsed, we shoot again.
        if (fireButton && ((currTime - timeOfLastShot) > minFireThreshold)) {
            timeOfLastShot = currTime;
            if (numShots === 3) {
                playerShotsX.push(currPlayerX);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX, currPlayerY);
                playerShotsX.push(currPlayerX + (imagePlayer.width - imageShot.width) / 2);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + (imagePlayer.width - imageShot.width) /
                      2, currPlayerY);
                playerShotsX.push(currPlayerX + imagePlayer.width - imageShot.width);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + imagePlayer.width - imageShot.width,
                      currPlayerY);
            }
            else if (numShots === 2) {
                playerShotsX.push(currPlayerX);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX, currPlayerY);
                playerShotsX.push(currPlayerX + imagePlayer.width - imageShot.width);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + imagePlayer.width - imageShot.width,
                      currPlayerY);
            }
            else {
                playerShotsX.push(currPlayerX + (imagePlayer.width - imageShot.width) / 2);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + (imagePlayer.width - imageShot.width) /
                      2, currPlayerY);
            }
        }
```

```
        if (moveLeft) { // If left arrow key pressed, move spaceship to the left.
            if (currPlayerX > currPlayerSpeed) currPlayerX -= currPlayerSpeed * adjustmentFactor;
            else currPlayerX = 1;
        }
        if (moveRight) { // If right arrow key pressed, move spaceship to the right.
            if ((currPlayerX + imagePlayer.width) < (canvas.width - currPlayerSpeed))
                currPlayerX += currPlayerSpeed * adjustmentFactor;
            else currPlayerX = canvas.width - imagePlayer.width - 1;
        }
        for (var i = 0; i < aliensX.length; i++) { // generate bombs
            if (Math.random() > (1 - (currLevel * adjustmentFactor / 3000))) {
                alienBombsX.push(aliensX[i] + (imageAlien.width - imageAlienBomb.width) / 2);
                alienBombsY.push(aliensY[i] + imageAlien.height);
                context.drawImage(imageAlienBomb, aliensX[i] + (imageAlien.width -
                      imageAlienBomb.width) / 2, aliensY[i] + imageAlien.height);
            }
        }
        checkCollisions();
        if (loseLife) { // If we have been hit or touched...
            if (numOfLives >= 1) document.getElementById("life" + numOfLives).style.display =
                 "none";
            numOfLives--;
            loseLife = false;
            if (numOfLives < 0) gameOver();
            else restartLevel();
        }
        else if (aliensX.length === 0) nextLevel();
        else { // Move everything and continue the fun.
            alienXSpeed = (5 + currLevel) * adjustmentFactor;
            context.drawImage(imagePlayer, currPlayerX, currPlayerY);
            moveShotsAndBombs();
            moveAliens();
            lastAnimationTime = new Date().getTime();
            requestAnimationFrame(gameLoop); //setTimeout("gameLoop()", timeOut);
        }
    }
}
```

checkCollisions

Den Ansatz zur Kollisionsüberprüfung hatte ich selbst in Büchern recherchiert. Ich würde nicht erwarten, dass ein Schüler diesen selbst entwickelt, obwohl er im Internet Beispiellösungen hätte finden können. Daher hatte ich im Tutorial Folgendes zum Thema erläutert: „*Um festzustellen, wann ein Objekt ein anderes berührt (z.B. eine Bombe trifft das Raumschiff oder eine Kugel einen Außerirdischen) betrachten wir jedes Bild als ein Rechteck mit der gegebenen Breite und Höhe. Eine Kollision findet statt, wenn zumindest ein Pixel eines Objektes sich an der gleichen Stelle wie ein Pixel des anderen Objektes befindet. Mit anderen Worten, es findet eine Überlappung der zwei Objekte statt.*



Überlappung bedeutet:

- *Der rechte Rand von Objekt 1 befindet sich rechts vom linken Rand von Objekt 2.*
- *Der linke Rand von Objekt 1 befindet sich links vom rechten Rand von Objekt 2.*
- *Der untere Rand von Objekt 1 befindet sich unter dem oberen Rand von Objekt 2.*
- *Der obere Rand von Objekt 1 befindet sich über dem unteren Rand von Objekt 2.*“

Kollisionen können zwischen Aliens und dem Raumschiff, Bomben und dem Raumschiff oder Kugeln und Aliens stattfinden. Diese drei Ereignisse müssen also von der Funktion `checkCollisions` behandelt werden.

Um eine Kollision zwischen einem Außerirdischen und einem Alien festzustellen, müssen die Felder mit den Alien-Koordinaten mittels einer Schleife durchlaufen werden. Vor der Schleife werden die Ränder des Raumschiffes bestimmt. In der Schleife werden folgende Schritte durchlaufen:

1. Ränder des Aliens bestimmen.
2. Überlappung des gerade betrachteten Aliens mit dem Raumschiff anhand des oben beschriebenen vierfachen Tests überprüfen. Falls eine Berührung stattgefunden hat, wird die globale Variable `loseLife` auf `true` gesetzt, worauf die Hauptfunktion namens `gameLoop` reagieren wird und die Funktion wird beendet.
3. Mittels einer zweiten Schleife wird überprüft, ob eine Kugel den gerade behandelten Alien berührt. Falls dies der Fall ist, müssen sowohl die Kugel als auch der Alien entfernt werden, der Score erhöht und die zweite Schleife verlassen werden. Schwierigkeiten bereitete die Verwaltung der Felder: Wenn eine Kugel einen Alien trifft, werden beide entfernt. In diesem Fall dürfen die Indizes, die zum Durchlaufen der Felder dienen, nicht erhöht werden, da die Felder ja jetzt um ein Element kleiner geworden sind und an den aktuellen Indexpositionen jetzt das Element steht, das vor dem Entfernen eine Position weiter nach rechts stand. Diese Problematik hatte ich zweimal an der Tafel erklärt. Bis zum Abgabetermin gelang es jedoch

nur wenigen Schülern, dies korrekt zu lösen. Realistisch betrachtet, angesichts des Schwierigkeitsgrades dieser verschachtelten Schleife, war dies nicht überraschend. In jeder Schleife müssen Felder durchlaufen, die korrekt indizierten Werte aus diesen Feldern ausgelesen und Entscheidungen aufgrund von Berechnungen mittels dieser Werte getroffen werden. Hierbei müssen mehrere Bedingungen miteinander verknüpft werden. Dass überhaupt jemand dies lösen konnte, obwohl hier Lösungsansätze erforderlich waren, die nirgendwo im Kurs vorgekommen waren, deutet darauf hin, dass die Problemlösekompetenz dieser Schüler ein respektables Niveau erreicht hat.

Zum Schluss werden die Koordinatenfelder der Bomben durchlaufen. Für jede Bombe werden die Ränder bestimmt und anschließend wird überprüft, ob sie das Raumschiff berührt. Ist dies der Fall, wird wieder die globale Variable `loseLife` auf `true` gesetzt, worauf die Hauptfunktion namens `gameLoop` reagieren wird und die Funktion wird beendet.

```
// Called by gameLoop.
function checkCollisions() { // check whether a bullet has hit an alien or an alien or alien bomb
touches the player
    var shotLeft, shotRight, shotTop, shotBottom, alienLeft, alienRight, alienTop, alienBottom,
        playerRight, playerBottom, alienIndex = 0, alienBombLeft, alienBombRight, alienBombTop,
        alienBombBottom, alienBombIndex = 0;

    // first check whether player has been touched by an alien or a bullet has touched an alien
    playerRight = currPlayerX + imagePlayer.width;
    playerBottom = currPlayerY + imagePlayer.height;

    while (alienIndex < aliensX.length) {
        alienLeft = aliensX[alienIndex];
        alienRight = alienLeft + imageAlien.width;
        alienTop = aliensY[alienIndex];
       // if an alien manages to leave via the bottom of the frame, we lose a life
        alienBottom = alienTop + imageAlien.height;
        if (alienRight >= currPlayerX && playerRight >= alienLeft && alienBottom >= currPlayerY ||
            alienBottom > canvas.height) {
            loseLife = true;
            return;
        }
        shotIndex = 0;
        // for each bullet check whether it touches the alien
        while (shotIndex < playerShotsY.length) {
            shotLeft = playerShotsX[shotIndex];
            shotRight = shotLeft + imageShot.width;
            shotTop = playerShotsY[shotIndex];
            shotBottom = shotTop + imageShot.height;
            if (alienRight >= shotLeft && shotRight >= alienLeft && alienBottom >= shotTop &&
               shotBottom >= alienTop) {
                playerShotsX.splice(shotIndex, 1);
                playerShotsY.splice(shotIndex, 1);
                aliensX.splice(alienIndex, 1);
                aliensY.splice(alienIndex, 1);
                alienIndex--;
```

```
                score = ~(~score + 1);
                document.getElementById("score").innerHTML = ~score;
                break;
            }
            shotIndex++;
        }
        alienIndex++;
    }

    while (alienBombIndex < alienBombsX.length) { // check whether player has been touched by a bomb
        alienBombLeft = alienBombsX[alienBombIndex];
        alienBombRight = alienBombLeft + imageAlienBomb.width;
        alienBombTop = alienBombsY[alienBombIndex];
        alienBombBottom = alienBombTop + imageAlienBomb.height;
        if (alienBombRight >= currPlayerX && playerRight >= alienBombLeft && alienBombBottom >=
               currPlayerY && alienBombTop <= playerBottom) {
            loseLife = true;
            return;
        }
        alienBombIndex++;
    }
}
```

4.2.2.6 *Ergebnis und Schlussfolgerungen*

Die Abgabe am 19.4.13 ergab folgendes Ergebnis: Zwei Schüler hatten eine vollkommen korrekte Kollisionsüberprüfung programmiert. Dreizehn Schüler hatten eine fehlerhafte `checkCollisions` Funktion entwickelt. Der häufigste Fehler war, wie oben beschrieben, ein Überspringen der nächsten Kugel und des nächsten Aliens, nachdem ein Alien von einer Kugel getroffen wurde. Dies ist jedoch ein Detail, das erstens selbst für einen erfahrenen Programmierer schwer aus dem Programmcode zu erkennen ist und zweitens beim Spielen nicht auffällt.

Viele Schüler hatten keine Bombengenerierung programmiert und somit auch die entsprechende Kollisionsüberprüfung schlichtweg weg gelassen. Das Feature von mehreren Kugeln ab bestimmten Levels wurde nur von zwei Schülern umgesetzt, das schnellere Schießen nur von einem. Die letzten zwei Punkte erklären sich daraus, dass die Kollisionsüberprüfung die gesamte Zeit beansprucht hat.

Trotz der detaillierten Problembeschreibung sowie meiner Erklärungen und Hilfestellungen, stellte das Lösen des T1IF Invaders Problems eine sehr hohe Herausforderung für die Klasse dar. Da ich das Spiel über mehrere Wochen mit hohem Zeitaufwand selbst entwickelt hatte, war ich mir dessen bewusst. Getreu meines Mottos „der Mensch wächst am Widerstand“ war dies einer der zwei Gründe, wieso ich diese Problemstellung gewählt hatte. Der andere war, dass ich wusste, dass die Programmierung eines Spiels den Schülern Spaß bereiten und sie somit intrinsisch motivieren würde. Die Erarbeitung von Wissen und Kompetenzen anhand einer motivierenden Problemstellung ist die

Grundidee von PBL. Obwohl es der Mehrheit der Schüler nicht gelang, die Kollisionsdetektion absolut korrekt zu implementieren, haben die meisten intensiv daran gearbeitet und ihr Verständnis der Lerninhalte signifikant gesteigert. Dies ging sehr deutlich aus meinen Gesprächen mit den einzelnen Schülern hervor, wo sich zeigte, dass die Mehrheit ein sehr viel tieferes Verständnis der Anwendungsgebiete von Schleifen, Feldern, Funktionen usw. entwickelt hatte. Wenn ich einen Schüler fragte, wie er vorgehen würde, um dieses oder jenes Problem zu lösen, bekam ich immer öfter konzeptionell korrekte Antworten, die sauber strukturiert und klar durchdacht waren. Jeder, der Erfahrung mit dem Unterrichten von Einführungskursen in die Programmierung hat, weiß, dass dies alles andere als selbstverständlich ist.

Auffallend war auch, dass immer mehr Schüler von sich aus dazu übergingen, wichtige Bereiche ihrer Programme zu kommentieren. Wenn man sie bei der Fehlersuche beobachtete, konnte man erkennen, dass die meisten ihre Debuggingkompetenz erheblich weiter entwickelt hatten. Es kam nur selten vor, dass ich um Hilfe gebeten wurde, ehe der Schüler mittels Ausgaben in die Konsole selbst systematisch versucht hatte, der Fehlerquelle auf die Spur zu kommen. Es wurde auch immer schwieriger, den Schülern die klassische Antwort „Ich habe keine Ahnung“ zu entlocken. Stattdessen hatten sie zunehmend gute Ideen und Vorschläge, wie man was angehen könnte und es haperte dann oft nur an einem Detail. Dies ging soweit, dass ich aufgrund der Lösungen von einzelnen Schülern die Musterlösung verbesserte. So hatte z.B. ein Schüler erkannt, dass man, anstatt die Felder für die Alienkoordinaten einmal zur Kollisionsüberprüfung mit dem Raumschiff und einmal zur Kollisionsüberprüfung mit den Kugeln zu durchlaufen, beides in einem Durchgang erledigen könnte, was Rechenzeit spart.

Am 23.4.13 erklärte ich die Musterlösung von T1IF Invaders im Detail.

4.2.3 Abschlussevaluation anhand einer selbsterstellten Problemstellung

Bei der Abschlussevaluation für CLISS2 ging es mir darum, den Schülern eine Gelegenheit zu geben, zu zeigen, was sie in den beiden CLISS Modulen gelernt hatten. Natürlich ist die extrinsische Motivation, das Modul zu bestehen, ein Faktor, der ausreichen sollte, um von den Schülern, die es können, Arbeiten zu erhalten, die die Mindeststandards erfüllen. Um jedoch wirklich herauszufinden, was die Schüler in relativ kurzer Zeit auf die Beine stellen können, bedurfte es einer starken intrinsischen Motivation. Ich bewegte mich daher ans linke Ende des Spektrums der Problemdefiniti-

on (cf. 2.7) und ließ die Schüler ihre eigene Problemstellung vorschlagen. Ihre Idee war bis zum 12.4.13 abzugeben. Als einzige Bedingung hatte ich vorgegeben, dass die Lerninhalte des Moduls so weit wie möglich angewendet werden sollten.

Die Bewertung (siehe unten) wurde anhand eines zehn- bis fünfzehn-minütigen Gespräches vorgenommen, in dem der Schüler mir seine Lösung im Detail erklären sollte. Somit war sichergestellt, dass ein Schüler nicht einfach eine Lösung aus dem Internet oder von seinem großen Bruder, der gerade Informatik studiert, übernehmen konnte.

Bereits bei der Bestimmung der Problemstellung war eine starke Spaltung der Klasse zu erkennen: Elf Schüler hatten mir zum Stichtag ihre Wahl mitgeteilt, während die anderen zehn sich damit sehr schwer taten. Bei ersteren handelte es sich vornehmlich um die guten bis sehr guten Schüler. Diese legten auch sofort mit der Arbeit los, was ihren Einsatz bei der Fertigstellung von T1IF Invaders eindeutig bremste. Die schwächeren Schüler hingegen versuchten, das Thema vor sich her zu schieben, was ich jedoch nicht zuließ. In Einzelgesprächen versuchte ich den Grund für ihr Zögern herauszufinden. Den schwächeren Schülern fiel nichts ein, das sie sich zutrauten, zu tun. Ich fragte nach ihren Interessen, gab Beispielideen und erklärte ihnen, was ich erwartete. Mit zweiwöchiger Verspätung erhielt ich dann die letzten Problembeschreibungen.

Seine eigene Problemstellung zu definieren stellt hohe Ansprüche an die Metakognition, da der Schüler ja sowohl seine eigenen Fähigkeiten als auch die Komplexität einer Problemstellung einschätzen muss, um zu entscheiden, ob er es in der gegebenen Zeit zufriedenstellend lösen kann. Dies ist für jemanden mit wenig Programmiererfahrung eine nicht zu unterschätzende Herausforderung.

Zwei Drittel der Schüler hatten ein Spiel gewählt. Es gab jedoch auch Anwendungen wie eine Restaurantverwaltung oder eine animierte Bildergalerie.

Für die Bearbeitung der Problemstellung standen offiziell drei Wochen zur Verfügung, wobei einige Schüler jedoch schon vorher vor allem zuhause daran gearbeitet hatten.

Ich beobachtete eine hohe Eigeninitiative und Motivation bei der Mehrheit der Schüler. Die meisten waren intensiv damit beschäftigt, ihre Lösung voranzutreiben, wobei sehr viel recherchiert und in der Gruppe diskutiert sowie sich gegenseitig unterstützt wurde. Von Zeit zu Zeit zeigte mir ein Schüler stolz den aktuellen Stand seiner Arbeit. Ich nutzte einen Großteil der Zeit, um die schwächeren Schülern individuell zu unterstützen.

Am 24.4.13 verbrachte ich fast die gesamte Doppelstunde mit einem Schüler, den ich am Vortag erneut darauf aufmerksam gemacht hatte, dass er mir seine Problemstellung noch nicht mitgeteilt hatte. Er hatte daraufhin resigniert gefragt, wann der Nachhilfekurs stattfinden würde. Es stellte sich heraus, dass er das Gefühl hatte, überhaupt nichts von der Materie zu verstehen. Ich ermunterte ihn und erklärte ihm am nächsten Tag die wichtigsten Lerninhalte anhand von der T1IF Invaders Musterlösung. Leider erwiesen sich meine Bemühungen in diesem Fall als vergebens, da dieser Schüler auch weiterhin resigniert da saß und auch nichts abgab.

Es gab drei weitere Schüler, denen ich des Öfteren meine Hilfe angeboten hatte, die jedoch keinerlei Fragen stellten und versuchten, irgend etwas zu programmieren, ohne sichtbare Fortschritte zu erzielen. Sie diskutierten ab und zu mit anderen Schülern, allerdings war auch hier die Resignation groß.

Am anderen Ende des Spektrums gab es sehr viel erfreulichere Beobachtungen. So hatte ich das ganze Jahr über viele Stunden mit zwei Schülern auf Skype verbracht, die hoch motiviert experimentierten und sich kontinuierlich weiter entwickelten.

4.2.3.1 *Bewertungsmethode*

Für die Bewertung habe ich die wichtigsten Kompetenzindikatoren aus den offiziellen Evaluierungsrastern von CLISS1 und CLISS2 (cf. 9.4) gewichtet und durch zwei zusätzliche Indikatoren (Interview und verspätete Abgabe) ergänzt. Von den siebzehn Indikatoren haben elf eine normale, fünf eine doppelte und einer eine zehnfache Gewichtung, in Abhängigkeit von der Wichtigkeit, die ich ihnen beimesse.

Das Beherrschen von Arrays und Funktionen, das Zerlegen von Problemen in Teilprobleme sowie die Fähigkeit komplexeren JavaScript Code zu entwickeln, betrachte ich als besonders wichtig.

Das Interview ist der Zeitpunkt, wo sich das vom Schüler erreichte Kompetenzniveau abschließend feststellen lässt. Daher hat es die mit Abstand höchste Gewichtung.

Eine verspätete Abgabe wurde mit einem Punkt pro zwei Tage geahndet. Eine vorzeitige Abgabe wurde hingegen mit einem Punkt pro zwei Tage belohnt. Dies erklärt, wieso drei Schüler eine Punktzahl von über sechzig erreicht haben.

Die Tabelle zeigt die Ergebnisse, wobei die Schülernamen durch S1 bis S21 anonymisiert wurden.

Sieben Schüler haben nichts abgegeben, weshalb sie keine Einzelbewertungen haben. Die abgege-

benen Lösungen befinden sich auf http://foxi.ltam.lu/COURS/cliss sowie der beigelegten CD. Aufgrund der Größe (hundert Seiten Quellcode) werden sie in den folgenden Abschnitten nur auszugsweise wiedergegeben.

Werte zwischen 0 und 2		Gewicht	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21
K1 + K2	Anwenden von grundlegenden Sprachelementen	1	2			2	2	2				2	2	2	2	2	2	2	2	2		2	2
	Anwenden von elementaren Kontrollstrukturen	1	2			2	2	2				1	2	2	2	2	2	2	2	2		2	2
	Anwenden von vordefinierten Funktionen	1	2			2	2	2				1	2	2	2	2	2	2	2	2		1	2
	Verschachteln von Kontrollstrukturen	1	2			2	2	2				0	2	2	2	0	2	0	2	0		2	2
	Verwendung von Arrays	2	2			2	2	1				2	2	1	2	2	1	1	1	2		1	2
	Funktionen mit und ohne Parameter	2	1			1	1	1				1	2	1	2	2	1	1	2	1		1	2
	Zerlegen des Problems in Teilprobleme die mittels Funktionen gelöst werden	2	2			2	2	2				1	2	2	2	1	2	1	2	2		1	2
	Validieren der Dateneingabe in Formularen	1	0			0	2	0				0	2	0	0	2	1	0	1	0		2	2
	DOM-Modifikationen	1	2			0	2	2				2	2	2	2	2	2	2	2	2		2	2
	Verändern von CSS-Eigenschaften mittels JavaScript	1	2			0	2	2				0	2	2	2	2	2	0	0	0		0	2
	Klar erkennbares Gesamtkonzept bzgl. Design, Funktionalität und Implementierung der Problemlösung	1	2			2	2	2				1	2	2	2	1	2	1	2	2		1	2
	Validiertes HTML und CSS	1	2			1	2	0				2	2	2	0	1	2	2	0	2		1	2
	Komplexerer JavaScript Code	2	2			1	2	1				0	2	1	2	1	2	0	1	1		0	2
Überzeugende Erklärungen im Interview		10	2			1	2	1				2	2	2	2	2	1	2	2	2		1	2
Zwischensumme			50			33	52	34				37	54	46	50	46	39	37	45	44		29	54
K3	Einbinden von vorgefertigten Codelösungen	1	2			1	2	1				1	2	1	2	1	1	1	1	1		1	2
K4	Selbstständig und motiviert arbeiten	2	2			1	2	1				1	2	1	2	1	1	1	1	1		0	2
Verspätete Abgabe		-1	-1			4	-5	1				4	-1	0	0	0	3	3	0	0		6	-1
Gesamtnote		60	57	0	0	32	63	36	0	0	0	36	61	49	56	49	39	37	48	47	0	24	61
Bewertung			++	-	-	0	++	0	-	-	-	0	++	+	++	+	0	0	+	+	-	-	++

	Sehr gut	5	24%
	Gut	4	19%
	Bestanden	5	24%
	Nicht bestanden	7	33%
	Gesamt	21	

4.2.3.2 *Ergebnisse*

Die Abgaben aller Schüler, die das Modul CLISS2 bestanden haben, befinden sich samt ihrer Namen auf der CLISS Homepage, wobei die sehr gut bewerteten Arbeiten größer als die gut bewerteten und diese wiederum größer als die als bestanden bewerteten Lösungen präsentiert sind. Im Sinne der Anerkennung der von den Schülern geleisteten Arbeit war mir dies wichtig. In der Folge beschreibe ich die Problemlösungen auch in dieser Reihenfolge, d.h. zuerst die fünf sehr guten, gefolgt von den vier guten und den fünf genügenden.

Die fünf sehr guten Arbeiten zeichnen sich dadurch aus, dass die jeweiligen Autoren mit hohem Einsatz ein Produkt entwickelt haben, das im technischen Anspruch weit über das zum Bestehen des Moduls erforderlichen Minimums hinausgeht. Darüber hinaus haben sie im Interview auf souveräne Art und Weise keinen Zweifel daran gelassen, dass sie wirklich die Autoren ihres Produktes sind und es syntaktisch und konzeptionell auf allen Ebenen beherrschen.

Die vier guten Arbeiten reichen nicht an die Komplexität und den Anspruch der sehr guten Arbeiten

heran, zeigen aber, dass die Autoren die Lerninhalte der beiden CLISS Module weitgehend beherrschen und dies im Interview auch beweisen können.

Die fünf genügenden Arbeiten sind entweder sehr viel simplerer Natur und wenden nur einen Teil der Lerninhalte an oder aber der Autor konnte mich im Interview nur zum Teil davon überzeugen, dass er alle Lerninhalte wirklich beherrschte.

Dodge the Asteroids

Der Autor dieses Spiels ist der Schüler, dessen Mutter mich anfangs des ersten Semesters per Email gebeten hatte, der Klasse die Materie zu erklären (cf. 4.1.4.3). Seitdem hat er mich alle paar Tage per Skype kontaktiert, um Fragen zu stellen oder um mir seine neuesten Ergebnisse zur Begutachtung zu schicken. Dabei waren seine Fortschritte beeindruckend. Seine ursprüngliche Verunsicherung war einer unübersehbaren Begeisterung gewichen. Er arbeitete täglich an seinen Lösungen und recherchierte solange, bis er seine sehr guten Ideen exakt nach seinen Vorstellungen implementiert hatte. Seine Fragen wurden zusehends anspruchsvoller und man konnte erkennen, dass er sich auf immer höheren Abstraktionsebenen bewegte.

Am 6. März teilte er mir mit, dass er sich zwei („Core HTML5 Canvas“ und „HTML5 Games“) der drei anspruchsvollsten Bücher, von den sechs, die ich der Klasse zu Beginn des Schuljahres mitgebracht hatte, gekauft hatte. Diese Bücher behandeln die Programmierung von Spielen auf recht fortgeschrittenem Niveau, inklusive der physischen Hintergründe der Kollisionsdetektion. Dass er diese Bücher nicht nur gekauft, sondern intensiv studiert hat, erkennt man an seinem Spiel „Dodge the Asteroids“.

Das Spiel besteht aus über siebenhundert Zeilen (cf. 9.2.2.2) sehr sauber strukturiertem JavaScript. Kommentare sind selten aber aussagekräftig und an den relevanten Orten vorhanden. Die Modularisierung ist so logisch und systematisch und die Namensgebung der Funktionen und Variablen so eindeutig, dass man sofort erkennt, wo was passiert.

Obwohl wir jQuery im Kurs gar nicht behandelt hatten, hat der Schüler es hier, wenn auch in nur einer Zeile, verwendet, um zu zeigen, dass er auch die Kompetenz 3 („Einbinden von vorgefertigten Codelösungen in der eigenen Webseite“) beherrscht:

```
function switchToGame() {
    $('#formular').css('display', 'none');
}
```

Eine Besonderheit des Spiels besteht darin, dass alle Spielobjekte per JavaScript nach Bedarf auf dem Canvas erzeugt werden, d.h. es werden keine Bilddateien eingebunden und das gesamte Programm befindet sich in einer einzigen HTML-Datei.

Die Liebe zum Detail erkennt man, unter anderem, an den vorbeiscrollenden Hintergrundsternen, der sehr flüssigen Animation des Raumschiffs, das ohne Beschleunigung langsam nach links gleitet, die Progress Bar, die Flamme, die gemalt wird, wenn das Raumschiff beschleunigt sowie der gesamten „Bossszene“, wo nicht nur der „Boss“ schön aus Grundformen erstellt wird, sondern auch ein animierter Laserstrahl das Raumschiff zu zerstören droht.

Wenn man sich den Code anschaut, erkennt man, dass der Schüler, um all dies zu implementieren, die gesamten Lerninhalte, mit Ausnahme von selbstdefinierten Funktionen mit Parametern, der beiden CLISS-Module in einer Art und Weise angewendet hat, die keinen Zweifel daran lassen, dass er sie beherrscht. Sofort sichtbar wird dies auf Syntaxebene anhand der mühelosen Anwendung von Verzweigungen und Schleifen beliebigen Verschachtelungsgrades. Hier ein Beispiel, wobei dies nur der innere Teil einer größeren Verschachtelung ist:

```
if (gameStarted) {
    getProgression();
    context.fillStyle = '#A0CBDD';
    for (var i = 0; i < maxAsteroidsAmount; i++) {
        asteroidsArrayX[i] -= asteroidSpeed;

        if (asteroidsArrayX[i] <= 0) {
            asteroidsArrayX.splice(i, 1);
            asteroidsArrayY.splice(i, 1);
            radiusAsteroidsArray.splice(i, 1);
            if (asteroidsOnBoard) {
                asteroidsArrayX.splice(i, 0, canvas.width);
                asteroidsArrayY.splice(i, 0, Math.floor(Math.random() * (canvas.width -
                      maxAsteroidsRadius)));
                radiusAsteroidsArray.splice(i, 0, Math.floor(Math.random() * (maxAsteroidsRadius) +
                      minAsteroidsRadius));
            }
        }

        context.beginPath();
        context.arc(asteroidsArrayX[i], asteroidsArrayY[i], radiusAsteroidsArray[i], 0, Math.PI*2);
        context.closePath();
        context.fill();
    }
}
```

Die Fähigkeit, auf konzeptioneller Ebene, ein großes Problem in kleinere Probleme aufzuteilen sowie angemessene Datenstrukturen und Funktionen zu definieren und diese einzusetzen, ist ebenfalls beeindruckend. Darüber hinaus werden zahlreiche Canvasmethoden verwendet, die im Kurs nicht

angesprochen wurden und die der Schüler sich aus den Büchern angeeignet hat. Hier z.B. der Beginn der Funktion, die die Bossszene beendet:

```
function stopBossAnimationFunction() {
    runBossAnimation = false;
    if (bossX <= canvas.width + 100) {
        bossX += 3;

        context.strokeStyle = 'black';
        context.fillStyle = 'orange';

        context.beginPath();
        context.rect(bossX, bossY, 60, 150);
        context.arc(bossX, bossY + 75, 40, 0.5 * Math.PI, 1.5 * Math.PI);
        context.moveTo(bossX + 75, bossY);
        context.lineTo(bossX + 75, bossY + 40);
        context.lineTo(bossX + 120, bossY + 20);
        context.lineTo(bossX + 75, bossY);
        context.moveTo(bossX + 75, bossY + 110);
        context.lineTo(bossX + 75, bossY + 150);
        context.lineTo(bossX + 120, bossY + 130);
        context.lineTo(bossX + 75, bossY + 110);
        context.rect(bossX - 40, bossY + 20, 40, 10);
        context.rect(bossX - 40, bossY + 120, 40, 10);
        context.fill();
        context.stroke();
        context.closePath();

        context.fillStyle = 'white';
        context.beginPath();
        context.arc(bossX - 5, bossY + 75, 30, 0.5 * Math.PI, 1.5 * Math.PI);
        context.moveTo(bossX - 5, bossY + 45);
        context.lineTo(bossX - 5, bossY + 105);
        context.rect(bossX, bossY + 15, 15, 20);
        context.rect(bossX, bossY + 115, 15, 20);
        context.rect(bossX + 45, bossY + 20, 10, 110);
        context.rect(bossX + 60, bossY, 15, 40);
        context.rect(bossX + 60, bossY + 110, 15, 40);
        context.fill();
        context.stroke();
        context.closePath();
}
```

Die Bücher enthalten nichts, was dem Spiel ähnelt, so dass hier eindeutig ein erfolgreicher Transfer des aus den Büchern Gelerntem auf eine neue Problemstellung stattgefunden hat.

Die Kollisionsüberprüfung, eine der größten Herausforderungen für angehende Programmierer, wurde vom Autor souverän gemeistert:

```
function checkCollision() {
    for (var i = 0; i < maxAsteroidsAmount; i++) {
        var distance = Math.sqrt(
                Math.pow(playerX - asteroidsArrayX[i], 2) +
                Math.pow(playerY - asteroidsArrayY[i], 2));
        //console.log(distance);
```

```
        if (playerY > asteroidsArrayY[i] - radiusAsteroidsArray[i]) {
            if (distance <= radiusAsteroidsArray[i] + 1) {
                gameOver();
            }
        }
        else {
            if (distance - 15 <= radiusAsteroidsArray[i] + 1) {
                gameOver();
            }
        }
    }

    if (shootSuperRock) {
        var distance = Math.sqrt(
                Math.pow(playerX - rockX, 2) +
                Math.pow(playerY - rockY, 2));
        if (playerY > rockY - 12) {
            if (distance <= 12) {
                gameOver();
            }
        }
        else {
            if (distance - 15 <= 12) {
                gameOver();
            }
        }
    }

    if (runBossAnimation) {
        for (var i = 0; i < bossLaserArrayX.length; i++) {
            if (playerX + 20 >= bossLaserArrayX[i] && playerX <= bossLaserArrayX[i] + 10 && playerY
               + 7.5 >= bossLaserArrayY[i] && playerY + 7.5 <= bossLaserArrayY[i] + 5) {
                gameOver();
            }
        }
        for (var i = 0; i < bossLaserArrayBottomX.length; i++) {
            if (playerX + 20 >= bossLaserArrayBottomX[i] && playerX <= bossLaserArrayBottomX[i] + 10
               && playerY + 7.5 >= bossLaserArrayBottomY[i] && playerY + 7.5
               <= bossLaserArrayBottomY[i] + 5) {
                gameOver();
            }
        }
        if (playerX <= 40) {
            gameOver();
        }
    }
}
```

Das Interview hat den sehr positiven Eindruck zweifelsfrei untermauert. Der Schüler konnte innerhalb kürzester Zeit das gesamte Programm sehr detailliert erklären und konnte auch argumentieren, wieso er welche Entscheidungen bezüglich Datenstrukturen und Funktionen getroffen hatte.

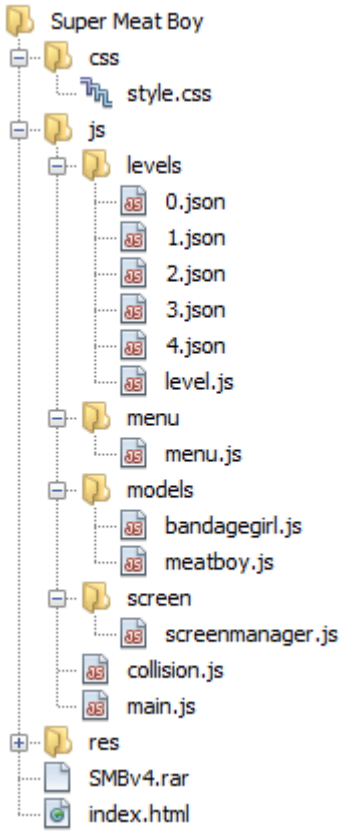
Aus meinen zahlreichen Interaktionen mit diesem Schüler weiß ich, dass er seine Metakognition zunehmends entwickelt hat. Er kann erklären, wie er bei der Lösung eines Problems vorgeht und welche Quellen er für welche Nachforschungen bevorzugt verwendet und weshalb. Es ist nicht verwun-

derlich, dass dieser Schüler sich zu einem der kompetentesten und aktivsten Berater in der Klasse entwickelt hat, dessen Rat und Meinung von vielen Schülern oft ersucht wurden.

Die Komplexität von Dodge the Asteroids übersteigt alles, was im Kurs behandelt wurde, inklusive T1IF Invaders. Dass ein Schüler, der zu Beginn des Kurses über keinerlei Programmiererfahrung (abgesehen von Scratch) verfügte, dies erreicht, illustriert zum einen zweifelsohne das Talent dieses Schülers, darüber hinaus jedoch auch die Grundidee von PBL, nämlich, dass Motivation ein entscheidender Faktor beim Lernerfolg ist und wir anhand von Problemstellungen uns die nötigen Kompetenzen aneignen.

Super Meat Boy Remake

Der Autor dieses Spiels hatte bereits Programmiererfahrung in C++ und anderen Programmiersprachen. Um eine Vorstellung davon zu bekommen, auf welchem Niveau er angelangt ist, braucht man sich nur die Verzeichnisstruktur des Spiels anzuschauen:



Das Programm ist in funktionale Module aufgeteilt, die alle zu Beginn eingebunden werden. Ein Startmenü gibt es keins, dafür aber geheime Modifier Codes, mit denen man nach dem Drücken der Taste M sofort in einen bestimmten Level vorstoßen oder die tödlichen Sägezahnblätter ausschalten

kann. Diese beschleunigen und bremsen mittels geschickter Canvasrotationsprogrammierung sehr geschmeidig, eins von vielen raffinierten Details. Sogar Blutspuren sind mit von der Partie. Das Spiel besteht aus fünf Levels, die in jeweils getrennten Dateien im JSON Format gespeichert sind, so dass man sie sehr einfach ändern oder einen Leveleditor entwickeln kann. Das JSON Format wurde im Kurs nie erwähnt.

Dieses Programm besteht ebenfalls insgesamt aus über siebenhundert Zeilen hoch strukturiertem JavaScript. Kommentare sind noch seltener als bei Dodge the Asteroids, die Namensgebung sowohl der Variablen wie auch der Funktionen und Dateien ist jedoch selbsterklärend.

Mehrdimensionale Felder werden hier ebenso selbstverständlich verwendet wie tief verschachtelte Verzweigungen und fortgeschrittene Canvasoperationen, die im Kurs nicht behandelt wurden. Hier der erste Teil einer Funktion:

```
function DrawLevel() {
    LevelLoaded = true;
    resetCollisions();

    for (var l = 0; l < CurLevelSawBlade.length; l++) {
        var CurSawBladePosX = CurLevelSawBlade[l].PosX;
        var CurSawBladePosY = CurLevelSawBlade[l].PosY;
        var CurSawBladeWidth = CurLevelSawBlade[l].Width;
        var CurSawBladeHeight = CurLevelSawBlade[l].Height;

        var CurSawBladeHalfWidth = CurSawBladeWidth / 2;
        var CurSawBladeHalfHeight = CurSawBladeHeight / 2;

        var CurSawBladeColBottom = CurSawBladePosY + CurSawBladeHeight;
        var CurSawBladeColTop = CurSawBladePosY;
        var CurSawBladeColLeft = CurSawBladePosX;
        var CurSawBladeColRight = CurSawBladePosX + CurSawBladeWidth;

        if (!SawBladesOff) {

            if (SawBladeRotatationAdd > -0.4) {
                SawBladeRotatationAdd -= 0.00007;
            }

            if (checkCollision(CurSawBladeColBottom, CurSawBladeColTop, CurSawBladeColLeft,
               CurSawBladeColRight) == 1) {
                SawBladeMat[l] = BloodySawBlade;
                MeatBoyDieAndSpawn();
                return;
            }
        } else {
            if (SawBladeRotatationAdd < 0) {
                SawBladeRotatationAdd += 0.0004;
            } else {
                SawBladeRotatationAdd = 0;
            }
        }
```

```
        context.save();
        context.translate(CurSawBladePosX, CurSawBladePosY);
        context.translate(CurSawBladeHalfWidth, CurSawBladeHalfHeight);
        SawBladeRotatation += SawBladeRotatationAdd;
        context.rotate(SawBladeRotatation);
        context.drawImage(SawBladeMat[1], -CurSawBladeHalfWidth, -CurSawBladeHalfHeight,
               CurSawBladeWidth, CurSawBladeHeight);
        context.restore();
    }
```

Selbsterstellte Funktionen mit Parametern werden sinnvoll eingesetzt. Hier ein Beispiel:

```
function checkCollision(BottomCol, TopCol, LeftCol, RightCol) {
    if (ColMeatBoyBottom + 1 >= TopCol && ColMeatBoyLeft + 8 <= RightCol && ColMeatBoyRight - 8 >=
       LeftCol && ColMeatBoyTop + 8 <= BottomCol) {
        MeatBoybottomCollision = true;
        MeatBottomCol = true;
    }

    if (ColMeatBoyLeft <= RightCol && ColMeatBoyTop + 8 <= BottomCol && ColMeatBoyBottom - 8 >=
       TopCol && ColMeatBoyRight - 8 >= LeftCol) {
        MeatBoyleftCollision = true;
        MeatLeftCol = true;
    }

   if (ColMeatBoyRight >= LeftCol && ColMeatBoyTop + 8 <= BottomCol && ColMeatBoyBottom - 8 >=
       TopCol && ColMeatBoyLeft + 8 <= RightCol) {
        MeatBoyrightCollision = true;
        MeatRightCol = true;
    }

    if (ColMeatBoyTop <= BottomCol && ColMeatBoyLeft + 8 <= RightCol && ColMeatBoyRight - 8 >=
       LeftCol && ColMeatBoyBottom - 8 >= TopCol) {
        MeatBoytopCollision = true;
        MeatTopCol = true;
    }
    if (!MeatTopCol && !MeatBottomCol && !MeatLeftCol && !MeatRightCol) {
        return 0;
    } else {
        return 1;
    }
}
```

Darüber hinaus verwendet der Autor Objekte, welche als solche im Kurs nur konzeptionell angesprochen und nicht verwendet wurden. Hier ein Beispiel:

```
var Level4 = {
    "name": "Level 04",
    "width": 3840,
    "height": 600,
    "playerSpawnX": 200,
    "playerSpawnY": 75,
    "targetPosX": 120,
    "targetPosY": 280,
};
```

```
var LevelName = [Level0, Level1, Level2, Level3, Level4];

CurLevelPlayerSpawnX = LevelName[Level].playerSpawnX;
```

Nicht nur am fehlenden Startmenü (die Funktionen `initMainMenu` und `UpdateMainMenu` sind leer) erkennt man, dass dieser Schüler noch sehr viel mehr leisten könnte, wenn er mehr Zeit investiert hätte. So fehlt an manchen Stellen das abschließende Semikolon, das zwar nicht obligatorisch, aber empfehlenswert ist. Um Werte miteinander zu vergleichen, wird der Operator == verwendet anstatt des zu bevorzugenden ===. Im Spiel werden auch keine Punkte angezeigt. Dies ändert jedoch nichts an der beeindruckenden Qualität der Lösung.

Das Interview verlief erwartungsgemäß souverän. Der Schüler kannte sein Kunstwerk in- und auswendig und erklärte nicht ohne Stolz, wieso er dies und jenes so und nicht anders gemacht hatte. Allerdings hatte die ursprüngliche Version noch gelegentlich Probleme mit der Kollisionsüberprüfung, so dass die Spielfigur ab und zu einfach hängen blieb. Nachdem ich den Autor darauf aufmerksam gemacht hatte, erhielt ich prompt eine korrigierte Version.

Man muss natürlich berücksichtigen, dass dieser Schüler durch sein starkes persönliches Interesse an der Programmierung dieses Modul mit viel größerer Vorbelastung antrat als seine Kollegen. Dies führte jedoch auch dazu, dass er oft prioritär an seinen eigenen Projekten arbeitete, z.B. mit OpenGL in C++ zu programmieren, da er nach seinen Experimenten mit JavaScript und WebGL zum Schluß kam, dass dies noch zu langsam sei. Dies ist schade, denn anfangs wollte er ein 3D-Projekt realisieren.

Dieser Schüler hatte keine vorherige Erfahrung mit JavaScript, konnte jedoch seine bereits vorhandenen Kompetenzen so effektiv einsetzen, dass er sich die Eigenheiten dieser Programmiersprache sowie den Zugriff auf das Document Object Model sehr schnell und effektiv aneignen konnte. Dies ist ein Musterbeispiel für die Anwendung von reinem PBL. Er hat sich alles, was für die Realisierung seiner Idee notwendig war, eigenständig angeeignet. Nur äußerst selten hat er mich etwas gefragt. Ich habe ihn jedoch sehr oft gebeten, den anderen zu helfen, was er auch sehr gut getan hat und damit einen erheblichen Beitrag zur Weiterentwicklung seiner Kollegen geleistet hat.

Die Strukturierung seiner Lösung lässt seinen C++-Hintergrund klar erkennen. Er hatte sich beschwert, dass JavaScript nicht objektorientiert sei. Dem widersprach ich. Er ging aber nicht hin und studierte daraufhin die Objektorientierung der Sprache.

Brincks

Dieser Schüler hatte die elfte Klasse bereits im alten Regime besucht, allerdings nicht bestanden. Im alten Regime wurde Pascal unterrichtet. Somit waren die Grundlagen der Programmierung für ihn nicht neu, die Programmierung des Document Object Models in JavaScript hingegen schon.

Dies war der Schüler, mit dem ich die meiste Zeit (insgesamt viele Stunden) auf Skype verbrachte. Wie die meisten tat er sich anfangs sehr schwer mit dem PBL-Ansatz, insbesondere auch, da er es gewohnt war, alles erklärt zu bekommen und dann mittels simpler Aufgaben zu trainieren. Somit verstand er anfangs nicht, wieso ich ihm nicht einfach spezifische Fragen mit Lösungsvorschlägen beantwortete, die er dann nur noch einzutippen brauchte. Es war ein zähes Ringen, ihn dazu zu bringen, seine Forschungskompetenz zu entwickeln und zu lernen, selbst Antworten auf seine Fragen zu suchen. Mit der Zeit änderte sich seine Vorgehensweise jedoch erheblich. Er begann vertrauter mit den besten Internetressourcen zu werden, hatte aber noch immer die Tendenz, die gefundenen Seiten nur zu überfliegen anstatt sie gründlich zu lesen. Es kam aber auch vor, dass er zwar die Antwort auf seine Frage gelesen hatte, sich dessen aber nicht bewusst war. Aus den zahlreichen Gesprächen mit ihm konnte ich beobachten, dass seine metakognitive Kompetenz sich im Verlaufe der beiden CLISS Module sehr stark steigerte. Er konnte immer präziser formulieren, wie er welches Teilproblem angegangen war, seine eigene Vorgehensweise kritisch kommentieren und selbst Alternativen und Verbesserungsvorschläge entwickeln und dann auch in seinen eigenen Lösungsprozess integrieren.

Das Ziel, sein eigenes Spiel nach seinen Vorstellungen zu entwickeln, motivierte ihn unnachgiebig weiter zu experimentieren. So erarbeitete er sich nach und nach ein tiefgehendes Verständnis der DOM-Programmierung, das deutlich über die im Kurs behandelten Aspekte hinaus ging. Er begann verschiedene Anwendungen, unter anderem ein Malprogramm, das recht gut funktionierte. Allerdings verlor er mit der Zeit das Interesse daran, da er das Gefühl hatte, ohne PHP nicht mehr weiter zu kommen. Er dachte dann über diverse Spielideen nach, bis er sich schließlich festlegte.

Das Spiel besteht aus über vierhundert Zeilen JavaScript. Alle Dateien befinden sich im gleichen Verzeichnis, der Code ist jedoch gut strukturiert. Die Namensgebung der Variablen und Funktionen ist selbsterklärend, weshalb Kommentare selten aber an relevanten Orten vorhanden sind. Die im Kurs behandelten Inhalte werden angewendet.

Auch hier entdeckt man viele Details, die zeigen, dass der Schüler sich bemüht hat, seine Ideen umzusetzen und dabei über die Kursinhalte hinausgegangen ist. So verwendet das Spiel Klänge zur

akustischen Rückmeldung sowie den lokalen Browserspeicher zum Speichern des Highscores. Letzteres war ein Tipp von mir, auf die Frage hin, ob man zum Speichern der Highscores eine Datenbank benötige. Die konkrete Implementierung hat der Autor jedoch selbst recherchiert:

```
function setData() {
    if (store.getItem("nickname") === "" || store.getItem("nickname") === null) {
        var name = prompt("Enter your nickname: ");
        while (name.length < 1 || name === " " || name === null) {
            name = prompt("Enter your nickname: ");
        }
        store.setItem("nickname", name);
    }
    if (score * levels[selectLevel.selectedIndex] > highscore) {
        highscore = score;
        store.removeItem("score");
        store.setItem("score", highscore);
    }
}
```

Das Hintergrundbild ändert sich auf Zufallsbasis:

```
var randomNumber = Math.floor((Math.random() * backgrounds.length));
canvas.style.backgroundImage = "url(" + backgrounds[randomNumber] + ")";
canvas.style.backgroundRepeat = "no-repeat";
canvas.style.backgroundSize = "100%";
```

Bei jedem fünften Level bekommt man ein zusätzliches „Leben“ und es wird sogar die aktuelle Framerate berechnet und farblich unterschiedlich angezeigt, je nachdem ob sie 20 Frames pro Sekunde unterschreitet oder nicht:

```
var thisLoop = new Date();
var fps = 1000 / (thisLoop - lastLoop);
lastLoop = thisLoop;
if (fps < 20) {
    fpsspan.style.color = "#DD004d";
    fpsspan.innerHTML = "Fps: " + parseInt(fps);
}
else {
    fpsspan.style.color = "lightgreen";
    fpsspan.innerHTML = "Fps: " + parseInt(fps);
}
```

Des Weiteren kann man den Schwierigkeitsgrad auswählen, wodurch die Ballgeschwindigkeit bestimmt wird.

Einer der schwierigsten Punkte bei diesem Spiel ist die Behandlung der Berührung des Balls mit dem Schläger. Hier müsste man den Berührungswinkel sowie die Geschwindigkeit und Bewegungsrichtung des Schlägers berücksichtigen, was allerdings nicht ganz triviale mathematische Berechnungen erfordern würde. Nachdem der Autor dies recherchiert hatte, entschied er sich für eine ein-

fachere Handhabung, indem er den Schläger in mehrere Segmente einteilte und die Reaktion des Balls in Abhängigkeit des berührten Schlägersegmentes programmierte. Selbst diese vereinfachte Lösung beansprucht hundert siebzehn Zeilen Code, wobei diese Zahl mittels einer einfachen Optimierung etwas reduziert werden könnte:

```
function checkCollision() {
    var brickLeft, brickRight, brickTop, brickBottom, bulletRight, bulletLeft, bulletTop,
       bulletBottom;
    for (var i = 0; i < BrickXPosArr.length; i++) {
        brickLeft = BrickXPosArr[i];
        brickRight = brickLeft + brickWidth;
        brickTop = BrickYPosArr[i];
        brickBottom = brickTop + brickHeight;
        bulletLeft = bulletXpos;
        bulletRight = bulletLeft + bulletWidth;
        bulletTop = bulletYpos;
        bulletBottom = bulletYpos + bulletHeight;
        if (brickRight >= bulletLeft && bulletRight >= brickLeft && brickBottom + 4 >= bulletTop &&
              bulletBottom >= brickTop) {
            blob.play();
            ctx.clearRect(BrickXPosArr[i] - 1, BrickYPosArr[i], brickWidth + 2, brickHeight);
            BrickXPosArr.splice(i, 1);
            BrickYPosArr.splice(i, 1);
            numOfBricks--;
            bulletYStep = bulletYStep * -1;
            score += 1 * levels[selectLevel.selectedIndex];
            document.getElementById("score").innerHTML = "Score: " + score;
        }
    }

    //Collision with Pad or Stick
    var middleStick = playerbrickXpos + (playerbrickWidth / 2);
    var leftStick = playerbrickXpos;
    var rightStick = playerbrickXpos + playerbrickWidth;
    var leftStickLeft, leftStickRight;
    var rightStickLeft, rightStickRight;
    var middleStickLeft, middleStickRight;
    if (bulletBottom - 2 > playerbrickYpos && bulletLeft <= playerbrickXpos + playerbrickWidth &&
       bulletRight >= playerbrickXpos) {
        bulletYStep = bulletYStep * -1;
        if (bulletXpos >= playerbrickXpos && bulletXpos < playerbrickXpos + 45) {
            //Left
            blob.play();
            if (numOfBricks < 1) {
                clearInterval(gameLoopID);
                nextLevel();
            }
            //console.log("Left");
            leftStickLeft = playerbrickXpos;
            leftStickRight = playerbrickXpos + 45;
            if (bulletXpos >= leftStickLeft && bulletXpos < leftStickLeft + 15) {
                //console.log("LeftStickLeft");
                bulletXStep = 4;
            } else if (bulletXpos <= leftStickRight + 45 && bulletXpos > leftStickLeft + 30) {
                //console.log("LeftStickRight");
                bulletXStep = 2;
```

```
            } else {
                //console.log("LeftStickMiddle");
                bulletXStep = 3;
            }
            //Stop Left
        } else if (bulletXpos > playerbrickXpos + 90 && bulletXpos <= playerbrickXpos + 135) {
            //Right
            blob.play();
            if (numOfBricks < 1) {
                clearInterval(gameLoopID);
                nextLevel();
            }
            //console.log("Right");
            rightStickRight = playerbrickXpos + playerbrickWidth;
            rightStickLeft = rightStickRight - 45;
            if (bulletXpos >= rightStickLeft && bulletXpos < rightStickLeft + 30) {
                //console.log("RightStickLeft");
                bulletXStep = -2;
            } else if (bulletXpos <= rightStickRight && bulletXpos > rightStickRight - 15) {
                //console.log("RightStickRight");
                bulletXStep = -4;
            } else {
                //console.log("RightStickMiddle");
                bulletXStep = -3;
            }
            //Stop Right
        } else {
            //Middle
            blob.play();
            if (numOfBricks < 1) {
                clearInterval(gameLoopID);
                nextLevel();
            }
            //console.log("Middle");
            middleStickLeft = playerbrickXpos + 45;
            middleStickRight = playerbrickXpos + 90;
            if (bulletXpos >= middleStickLeft && bulletXpos < playerbrickXpos + 60) {
                //console.log("MiddleStickLeft");
                bulletXStep = 1;
            } else if (bulletXpos <= middleStickRight && bulletXpos > playerbrickXpos + 75) {
                //console.log("MiddleStickRight");
                bulletXStep = -1;
            } else if (bulletXpos >= playerbrickXpos + 60 && bulletXpos <= playerbrickXpos + 75) {
                //console.log("MiddleStickMiddle");
                bulletXStep = 0;
            }
        }
    }
    //Stop Collision with Pad
    //Collision width Borders
    if (bulletTop < bulletYStep)
        bulletYStep = bulletYStep * -1;
    if (bulletXpos < bulletXStep)
        bulletXStep = bulletXStep * -1;
    if (bulletXpos > (100 + (brickWidth * brickPerRow) + 5) - bulletWidth)
        bulletXStep = bulletXStep * -1;
    if (bulletYpos > canvas.height) {
        clearInterval(gameLoopID);
        lives--;
```

```
        if (lives < 0) {
            clearInterval(gameLoopID);
            lives = 3;
            alert("Game Over! Score: " + score);
            setData();
            showSplash();
        } else
            newBall();
    }
    ctx.drawImage(playerbrick, playerbrickXpos, playerbrickYpos);
}
```

Insgesamt zeigen die Komplexität und Strukturierung der Lösung die Tiefe des Verständnisses, das der Autor entwickelt hat.

Im Interview überzeugte er durch sehr klare und gut strukturierte Erklärungen, die nicht nur zeigten, dass er das Programm bis ins kleinste Detail verstand, sondern auch, dass er das Spiel sorgfältig durchdacht und geplant und sich konzeptionelle Gedanken gemacht hatte.

Reach The Exit

Der Autor dieses Spiels verfügte bereits über Programmiererfahrung in C++ und anderen Programmiersprachen. Dies zeigt sich in seiner Lösung, die sauber strukturiert und auf verschiedene Verzeichnisse aufgeteilt ist. Außerdem verwendet er in großem Umfang verschachtelte switch-Anweisungen. Diese Anweisung wurde im Kurs nicht behandelt. Allerdings hat der Autor sich das Leben sehr viel schwerer gestaltet als nötig:

```
switch (worldFromMenu) {
    case "1":
        currWorld = 1;
        switch (levelFromMenu) {
            case "1":
                currLevel = 1;
                switchToGame(false);
                break;
            case "2":
                currLevel = 2;
                switchToGame(false);
                break;
            case "3":
                currLevel = 3;
                switchToGame(false);
                break;
            case "4":
                currLevel = 4;
                switchToGame(false);
                break;
            case "5":
                currLevel = 5;
                switchToGame(false);
                break;
```

```
        }
        break;

    case "2":
        currWorld = 2;
        switch (levelFromMenu) {
            case "1":
                currLevel = 1;
                switchToGame(false);
                break;
            case "2":
                currLevel = 2;
                switchToGame(false);
                break;
            case "3":
                currLevel = 3;
                switchToGame(false);
                break;
            case "4":
                currLevel = 4;
                switchToGame(false);
                break;
            case "5":
                currLevel = 5;
                switchToGame(false);
                break;
        }
        break;
}
```

Diese 53 Zeilen Code könnte man mit identischer Funktionalität folgendermaßen formulieren:

```
currWorld = worldFromMenu;
currLevel = levelFromMenu;
switchToGame(false);
```

Als ich den Schüler fragte, wie man dies kürzer gestalten könnte, fand er auf Anhieb die Lösung.

Das Programm besteht aus über achthundert Zeilen JavaScript. Die Variablen- und Funktionsnamen sind selbsterklärend. Kommentare sind jedoch sehr selten und die vorhandenen sind teilweise redundant, da sie nur den Funktionsnamen wiederholen.

Das Ziel des Spiels ist es, den Ausgang aus verschiedenen Labyrinthen zu erreichen. Im Startmenü kann man wählen, in welchem Labyrinth aus welchem Level man starten möchte. Der Kontrast ist leider denkbar ungünstig gewählt. Der Autor hat, seit ich ihn kenne, eine Schwäche für fast unlesbare Farbkombinationen, lässt sich aber nicht davon abbringen.

Die fünf Level jeder der zwei Welten sind in getrennten Dateien als zweidimensionale Felder mit Nullen und Einsen gespeichert, so dass sie sehr leicht verändert werden können. Verwendet werden sie z.B. wie folgt:

```
switch (currMap[currY - 1][currX]) {
case 0:
        currMap[currY][currX] = 0;
        currY--;
        currMap[currY][currX] = 3;
```

Die Lerninhalte von CLISS1 und CLISS2 kommen zur Anwendung, inklusive tief verschachtelter Verzweigungen und Schleifen und selbsterstellter Funktionen mit mehreren Parametern:

```
function updateInfoWindow(player, level, moves, paused) {
    if (player === true)
        infoPlayer.innerHTML = "Good Luck, " + playerName;
    if (level === true) {
        infoWorld.innerHTML = "World: " + currWorld;
        infoLevel.innerHTML = "Level: " + currLevel;
    }
    if (moves === true)
        infoMoves.innerHTML = "Moves: " + movesLeft;
    if (paused === true) {
        if (SPACE === false)
            infoPaused.innerHTML = "Have Fun !";
        else if (SPACE === true)
            infoPaused.innerHTML = "Game Paused !";
    }
}
```

Darüber hinaus wird jQuery verwendet:

```
$("#titleDiv").css("display", "none");
$("#gameDiv").css("display", "block");
```

Im Bewertungsgespräch konnte der Autor sein Werk detailliert erklären.

Dieser Schüler gibt bei Fragen an die Klasse sehr oft sehr gute Antworten, ist ansonsten aber eher zurückhaltend. Er arbeitet sehr selbstständig und seine Medienkompetenz hat ein sehr hohes Niveau erreicht, so dass er sich sehr schnell in neue Dinge einarbeiten kann. Seine Leidenschaft für und seine Erfahrung mit der Programmierung haben ihm klare Vorteile gebracht. Nichtsdestotrotz gilt erfahrungsgemäß auch für diesen Schüler, dass er sich nur für ihn interessierende Herausforderungen wirklich engagiert. Daher war es auch hier von Vorteil, dass er die Problemstellung und die Lösung komplett selbst bestimmen konnte, was ihn sichtlich motivierte.

Grafische Darstellung von Service Nodes

Diese Anwendung dient dazu, die auf Serverseite generierten grafischen Darstellungen der Daten von Gateway Support Nodes anzuzeigen. Letztere sind Bestandteil des General Packet Radio Service (GPRS), das es mobilen Netzwerken erlaubt, IP-Pakete an externe Netzwerke wie z.B. das Internet zu übertragen.

Es stehen insgesamt 23 Bilder zur Auswahl, in jeweils kleiner und großer Ausführung. Die Bilder sind in drei Kategorien aufgeteilt, die über die Navigationsleiste ausgewählt werden können. Es gibt zwei Anzeigemodi, zwischen denen man mit den Pfeiltasten links und rechts hin- und herschalten kann. Der erste Modus zeigt alle Bilder der gewählten Kategorie in kleiner Größe an. Man kann mit den Pfeiltasten nach oben oder unten scrollen. Der zweite Modus zeigt jeweils nur ein Bild der gewählten Kategorie in Vollgröße an. Mit den Pfeiltasten nach oben und unten kann man das vorhergehende oder das nächste Bild wählen.

Das Programm besteht aus rund zweihundert Zeilen JavaScript und ist somit mit Abstand das kürzeste der sehr gut bewerteten Arbeiten. Jedoch ist die Komplexität der Umsetzung hoch und die Beherrschung der Materie sehr deutlich. Kommentare sind so gut wie nicht vorhanden, allerdings ist das Programm sauber strukturiert und auf mehrere Verzeichnisse und Dateien aufgeteilt, wobei die Namensgebung der Verzeichnisse, Dateien, Variablen und Funktionen das Verständnis stark erleichtern.

Der Autor verfügte bereits über vorherige Programmiererfahrung mit anderen Programmiersprachen.

Die Lerninhalte von CLISS1 und CLISS2 kommen zur Anwendung, inklusive tief verschachtelter Verzweigungen und Schleifen und selbsterstellte Funktionen, wovon eine mit Parametern. Des Weiteren beherrscht er den Umgang mit mehrdimensionalen Feldern und dem Canvas ganz gut. Hier ein Beispiel:

```
function drawOverview() {
    var Xoffset = 0;
    var Yoffset = 0 + scrollY;
    var colCounter = 0;
    for (var i = 0; i < 3; i++) {
        if (i == curCat) {
            for (var j = 0; j < smallImg[i].length; j++) {
                colCounter++;
                if (colCounter > maxCols) {
                    Yoffset += smallImgSize[1] + 10;
                    Xoffset = 0;
                    colCounter = 1;
                }
                context.drawImage(smallImg[i][j], ImgPosX[i][j] + Xoffset, ImgPosY[i][j] + Yoffset,
                      smallImgSize[0], smallImgSize[1]);
                Xoffset += smallImgSize[0] + 20;
            }
        }
    }
}
```

Allerdings ist ersichtlich, dass die Zeit knapp wurde, was mir der Schüler auch mitgeteilt hatte. So

sind zwei der fünf Menüpunkte funktionslos.

Im Bewertungsgespräch konnte er sein Werk gut und detailliert erklären.

Animierte Bildergalerie

Das Programm besteht aus gut hundert Zeilen JavaScript. Mit einem Mausklick auf den rechten schwarzen Bereich wird das gerade angezeigte Bild nach rechts hinaus bewegt und das nächste Bild bewegt sich von links in die Mitte. Bei einem Mausklick auf den linken schwarzen Bereich kann man das vorherige Bild wieder anzeigen lassen, wobei auch hier die geschmeidige Animation eingesetzt wird. Das Design ist ansprechend.

Das Programm ist klar strukturiert und leicht verständlich. Der Komplexitätsgrad ist deutlich geringer als bei den sehr guten Lösungen. Wie bei den meisten Abgaben gibt es keine Formularvalidierung und keine Funktionen mit Parametern. Ansonsten ist aber offensichtlich, dass der Autor die Materie gut beherrscht, was sich im Interview bestätigt hat. Insgesamt hat diese Arbeit nur knapp eine sehr gute Bewertung verpasst.

Hier die Funktion zur Bildanimation:

```
function moveImg(){
    context.clearRect(0,0,canvas.width,canvas.height);
    context.drawImage(arrimg[arrpos], imgPosX ,imgPosY);
    if (imgIsMovingRight){
        imgPosX += 10;
        if(imgPosX === 140){
            imgIsMovingRight = false;
            pressRight = false;
            window.clearTimeout(timer)
        }
        if(imgPosX >= 1000){
            imgPosX = 0;
            change = true;
        }
    }
    if (imgIsMovingLeft){
        imgPosX -= 10;
        if(imgPosX === 140){
            imgIsMovingLeft = false;
            pressLeft = false;
            window.clearTimeout(timer)
        }
        if(imgPosX + 720 <= 0){
            console.log('here');
            imgPosX = 1000;
            change = true
        }
    }
    if (pressRight){
```

```
        if(change){
            if (arrpos >= 3){
                arrpos = 0;
            } else {
                arrpos++;
            }
            change = false;
        }
    }
    if (pressLeft){
        if(change){
            if (arrpos <= 0){
                arrpos = 3;
            } else {
                arrpos--;
            }
            change = false;
        }
    }
    document.getElementById('Infotext').innerHTML = arrtext[arrpos];
    document.getElementById('Infotext').style.color = arrcolor[arrpos];
    timer = setTimeout("moveImg()", 1000/60);
}
```

Napoli

Der Autor hat den Anfang einer Restauranttischverwaltung programmiert, die Buch darüber führt, was an welchem Tisch bestellt wird.

Das Programm besteht aus rund dreihundert Zeilen JavaScript, dies liegt jedoch zu einem großen Teil daran, dass für jeden der vierundsechzig Knöpfe für Getränke und Gerichte ein eigener Event Handler definiert wurde, weil der Autor sich nicht zutraute, dies mit einer einzigen Funktion mit einem Parameter sehr viel eleganter zu lösen. Dies ist um so erstaunlicher, als er eine Funktion mit Parameter entwickelt hat und diese in jedem der vierundsechzig Event Handler verwendet! Hier die ersten drei:

```
function onClick11() {
    display(0);
}
function onClick12() {
    display(1);
}
function onClick13() {
    display(2);
}
```

Der Transfer hat hier nicht geklappt. Dabei hatte ich ihn einige Tage vor der Abgabe darauf aufmerksam gemacht. Es lag wahrscheinlich daran, dass ich ihn mit meinem Vorschlag, anstatt diese vierundsechzig Knöpfe alle einzeln im HTML-Teil zu definieren, dies mittels der Methode `docu-`

`ment.createElement` in JavaScript zu tun, überfordert hatte. Ich hatte ihm einen Teil der Lösung gezeigt, aber er traute es sich nicht zu, dies in der kurzen verbleibenden Zeit fertig zu stellen. So bevorzugte er den sehr platzraubenden, aber für ihn einfacheren Weg zu gehen. Die elegantere Lösung hätte das Programm auf ein Drittel seiner Größe reduziert.

Nach dieser Kritik muss ich allerdings anerkennen, dass er große Fortschritte mit den Grundlagen der JavaScript-Programmierung gemacht hat. Er beherrscht die Erstellung von Funktionen, den Umgang mit Feldern, die DOM-Manipulation, die Validierung von Dateneingaben und hat sich inzwischen auch angewöhnt, Debugginginstruktionen einzubauen. Die verwendeten Verzweigungen und Wiederholungen sind nicht verschachtelt und die Komplexität des Programms ist geringer als bei den sehr guten Arbeiten. Allerdings ist das Problem, abgesehen von dem oben beschriebenen Aspekt, sauber in Teilprobleme zerlegt und mittels Funktionen gelöst. Insbesondere hat er sich erfolgreich bemüht, eine sauber formatierte Textausgabe zu programmieren, welche ein tieferes Verständnis der grundlegenden JavaScript-Elemente illustriert:

```
function display(index) {
    //Set of 'space'
    var spaces = "";
    for (var j = 0; j < spacing - commandTableName[index].length; j++) {
        spaces += " ";
    }
    listCommand[table] = listCommand[table] + commandTableName[index] + spaces +
       commandTablePrice[index] + "€" + "\n";
    textArea.innerHTML = listCommand[table];
    totalPrice[table] += commandTablePrice[index];
    console.log(totalPrice[table]);
    textAreaTotal.innerHTML = "TOTAL: " + totalPrice[table] + "€";
}
```

Das Interview war überzeugend. Der Autor konnte nicht nur sein Werk, sondern auch seine Vorgehensweise bei der Lösung der Problemstellung sowie seine dabei erlebten Schwierigkeiten erklären.

Online Ordering System

Ähnlich wie bei Napoli geht es auch hier um ein System, das Bestellungen verwaltet, respektive sammelt und die Rechnung präsentiert. Das Programm macht einen insgesamt noch nicht ganz ausgereiften Eindruck, sowohl was das Design als auch den JavaScript Code betrifft. In letzterem finden sich noch Anweisungen, die nicht benötigt werden. Andererseits wird ein Feld mit den Bestellungen zwar korrekt befüllt, anschließend aber nicht verwendet. Hier zeigt sich der Zeitdruck. Das Programm besteht aus rund dreihundert sechzig Zeilen Code. Zweihundert davon werden für eine gigantische `switch`-Anweisung verwendet, die wie bei Napoli zeigt, dass der Autor noch nicht das

Abstraktionsniveau erreicht hat, das es ihm erlauben würde, dies eleganter und sehr viel kürzer mit geschickter Feldprogrammierung zu realisieren. Allerdings kann man dies auch nicht erwarten, denn eine Problemstellung von ähnlicher Komplexität in dieser Hinsicht war im Kurs nie vorgekommen. Die `switch`-Anweisung an sich war auch nicht behandelt worden. Da es sich jedoch um den zweiten Schüler der Klasse handelte, der vorheriges Jahr bereits auf einer T1IF das alte Lehrprogramm mit Programmierung in Pascal absolviert hatte, war er mit dieser Anweisung vertraut.

Dies erklärt wahrscheinlich auch seine problemlose Erstellung einer Funktion mit zwei Parametern, die es ihm erlaubt hat, elegant das Problem von Napoli, mit einer separaten Funktion für jeden Knopf, zu umschiffen, indem jeder Knopfdruck die gleiche Funktion mit unterschiedlichen Parametern aufruft. Hier der Anfang dieser Funktion:

```
function sum(x, y) {
    var i = 0;
    dailyorder.push(y);
    i += 1;
    console.log(x + " , " + y);
    dailyresult = dailyresult + x;
    switch (y) {

        case "coke ":
            if (coke === 0) {
                alert("cant order/ inventory empty");
            }
            else
            {
                coke = coke - 1;
                result = result + x;
                document.getElementById("output").value = result;
                document.getElementById("outtext").value += y + "\n";
            }

            break;
        case "fanta ":
            if (fanta === 0) {
                alert("cant order/ inventory empty");
            }
            else
            {
                fanta = fanta - 1;
                result = result + x;
                document.getElementById("output").value = result;
                document.getElementById("outtext").value += y + "\n";
    }
    break;
```

Auch hier erkennt man, dass eine optimierte Lösung nur einen Bruchteil der ursprünglichen Länge aufweisen würde.

Das Programm ist sauber strukturiert, leicht verständlich und von mittlerer Komplexität. Die Zeitan-

zeige wurde aus einem externen Skript übernommen und angepasst, wobei hier noch ein überflüssiger Timeraufruf vorhanden ist:

```
function updateTime() {
    document.getElementById("theTimer").firstChild.nodeValue =
            new Date().toTimeString().substring(0, 8);
}
window.setTimeout("updateTime()", 0);
window.setInterval("updateTime()", 1000);
```

Im Interview konnte der Autor seine Lösung in aller Tiefe erklären und selbstkritisch seine Vorgehensweise und potentielle Verbesserungsmöglichkeiten analysieren.

Seine vorherige Programmiererfahrung ist ihm eindeutig zugute gekommen, allerdings ist auch klar ersichtlich, dass er sich die Besonderheiten von JavaScript und der DOM-Programmierung angeeignet hat. Des Weiteren hat er externe Lösungen, z.B. die Anzeige der Uhrzeit, an seine Bedürfnisse angepasst.

HellBlog

Hier handelt es sich um eine dynamische Erweiterung der statischen Website, die der Autor letztes Jahr im T0IF HTSTA Kurs erstellt hatte. Die Scrollgeschwindigkeit des Laufbands, bestehend aus zehn Logos, respektive Kästchen, kann vom Benutzer eingestellt werden. Wenn der Mauszeiger sich über einem der laufenden Bilder befindet, werden die Logos angezeigt. Wenn der Mauszeiger sich wieder weg bewegt, erscheinen erneut die schwarzen Quadrate.

Das Programm besteht aus nur dreiundsechzig Zeilen JavaScript. Es handelt sich eindeutig um eine Minimalistenarbeit. Allerdings enthält sie trotz der Kürze eine clevere Anwendung der meisten der im Kurs behandelten Konzepte und Elemente, wobei allerdings keine verschachtelten Verzweigungen oder Schleifen und auch keine selbsterstellten Funktionen mit Parametern zum Einsatz kommen.

Hier das Skript:

```
var canvas = document.getElementById('canvas');
canvas.width = 700;
var context = canvas.getContext('2d');
var imageHead = new Image();
var imageHead2 = new Image();
var timeOut = 1000 / 60;
var headDirection = 1;
var mouseX = 0, mouseY = 0
var imageHeadX = [];
var imageHeadY = [];
var headXSpeed = 5;
```

```
var headYSpeed;
var adjustmentFactor = 0.2;
var numHead = 10;
imageHead.src = 'head.png', imageHead.width = 30, imageHead.height = 30;
imageHead2.src = 'head2.png', imageHead2.width = 30, imageHead2.height = 30;
var actualImage = imageHead;

setTimeout("gameLoop()", timeOut);

function gameLoop() {
    context.clearRect(0, 0, 50000, 500000);
    moveHead();
    setTimeout("gameLoop()", timeOut);
}

canvas.onmouseover = change;
canvas.onmouseout = changeToNormal;

function changeToNormal() {
    actualImage = imageHead;
}

function change() {
    actualImage = imageHead2;
}

function init() {
    var x = (numHead * 40) * -1, y = 1;
    for (var i = 0; i < numHead; i++) {
        imageHeadX[i] = x;
        imageHeadY[i] = y;
        context.drawImage(actualImage, imageHeadX[i], imageHeadY[i]);
        x += actualImage.width + 10;
    }
}

function moveHead() {
    if (imageHeadX[0] > canvas.width) init();
    for (i = 0; i < imageHeadX.length; i++) {
        imageHeadX[i] += headDirection * headXSpeed;
        context.drawImage(actualImage, imageHeadX[i], imageHeadY[i]);
    }
}

function enter() {
    headXSpeed = myInput.value;
}

function refreshSpeed() {
    headXSpeed = 5;
}

function logo() {
    actualImage = imageHead2;
}

function hide() {
    actualImage = imageHead;
}
```

```
init();
```

Das Interview war überzeugend. Der Autor konnte über die Details seines Programms hinaus alternative Ideen erläutern und beweisen, dass er die Materie beherrscht.

Pong!

Der Autor hatte ursprünglich geplant, ein Snakespiel zu entwickeln. Im Interview zeigte sich jedoch sofort, dass er seine Lösung fast vollständig aus dem Internet übernommen hatte und nicht überzeugend erklären konnte. Ich bot ihm an, etwas selbst zu entwickeln und es mir dann zu erklären. Daraufhin entwickelte er, allerdings mittels intensiver Unterstützung des Autors von Brincks, eine Variante des Klassikers Pong.

Das Ergebnis besteht aus rund vierhundert achtzig Zeilen JavaScript, wobei die Handschrift des Autors von Brincks nicht zu übersehen ist. Letzteres zeigt sich in der Komplexität, der Strukturierung und den eingebauten Features. Insbesondere stammt auch die Idee zur Implementierung der Kollisionsüberprüfung von ihm. Dementsprechend legte ich im zweiten Gespräch besonderen Wert darauf, das genaue Verständnis des Schülers auszuloten. Er konnte das Programm sowohl vom Gesamtaufbau her wie auch im Detail erklären, wobei er auch auf eigene Initiative hin die Aspekte identifizierte, die er nicht vollständig beherrschte, wie z.B. die Kollisionsüberprüfung. Ich stellte gezielte Fragen dazu, wie genau dies oder jenes funktionierte und kam zur Überzeugung, dass er die grundlegenden JavaScript-Elemente sowie deren Einsatz zur Lösung von Problemstellungen auf einem ausreichend hohen Niveau beherrschte, um das Modul bestanden zu haben.

Pong

Auch dieser Autor hat versucht, ein Pongspiel zu programmieren, ist dabei jedoch an der Kollisionsprüfung gescheitert. Dies ist umso erstaunlicher, als die Kollisionsabfrage in diesem Fall sich nicht wesentlich von der Kollisionsabfrage in T1IF Invaders unterscheidet. Ich habe das ganze Jahr über mich bemüht, auch diesen Schüler bei seinem Lernprozess zu unterstützen, was auch zu sichtbaren Fortschritten geführt hat. In diesem Fall war das Problem jedoch, dass der Autor seine Problemstellung mit Verspätung ausgewählt hat, die Zwischenabgabe ausgelassen und seine Lösung mit acht Tagen Verspätung abgegeben hatte. Er hatte den Aufwand unterschätzt, war jedoch auch nicht motiviert genug, meine Hilfe oder die eines Kollegen in Anspruch zu nehmen.

Das Ergebnis besteht aus hundert Zeilen JavaScript von vergleichsweise geringer Komplexität. Ver-

schachtelte Kontrollstrukturen fehlen ebenso wie selbsterstellte Funktionen mit Parametern, Validierung von Formularen oder programmatische Veränderung von CSS-Eigenschaften. Die grundlegenden Sprachelemente kommen jedoch zur Anwendung und das Programm ist sauber strukturiert:

```
var canvas = document.getElementById("canvas");
var context = canvas.getContext("2d");
var currPlayer1X = 20;
var currPlayer1Y = 20;
var currPlayer2X = 960;
var currPlayer2Y = 480;
var moveUp, moveDown, move2Up, move2Down;
var ballpos = [1, 1];

function moveBall() {
    ballpos[0] = ballpos[0] + 1;
    ballpos[1] = ballpos[1] + 1;

    if (ballpos[0] >= canvas.height) {

        ballpos[1] = ballpos[1] - 2.5;
    }
    /*if (ballpos[1] <= 0){
     ballpos[1] = ballpos[1]++ ;
     }*/

}
function score2() {
    if (ballpos[0] >= canvas.width) {
        gameover();
    }
}

function Ball() {
    context.fillStyle = "white";
    context.fillRect(ballpos[0], ballpos[1], 10, 10);
}
function player1() {
    context.fillStyle = "white";
    context.fillRect(currPlayer1X, currPlayer1Y, 25, 100);
}
function player2() {
    context.fillStyle = "white";
    context.fillRect(currPlayer2X, currPlayer2Y, 25, 100);
}
function move() {
    if (moveDown && currPlayer1Y + 105 < canvas.height) {
        currPlayer1Y += 10;
    }
    if (moveUp && currPlayer1Y > 0) {
        currPlayer1Y -= 10;
    }
    if (move2Up && currPlayer2Y + 105 < canvas.height) {
        currPlayer2Y += 10;
    }
    if (move2Down && currPlayer2Y > 0) {
        currPlayer2Y -= 10;
    }
```

```
}

function gameover() {
    document.alert(Gameover)
}

function handleKeydown(event) {
    if (event.keyCode === 87) { // move up Player1
        moveUp = true;
    }
    else if (event.keyCode === 83) { // move down Player1
        moveDown = true;

    }
    else if (event.keyCode === 38) { // move down Player2
        move2Down = true;
    }
    else if (event.keyCode === 40) {    // move up Player2
        move2Up = true;
    }
}

function handleKeyup(e) {
    if (e.keyCode === 87) {
        moveUp = false;
    }
    else if (e.keyCode === 83) {
        moveDown = false;
    }
    else if (e.keyCode === 38) {
        move2Down = false;
    }
    else if (e.keyCode === 40) {
        move2Up = false;
    }
}

function gameloop() {
    context.clearRect(0, 0, canvas.width, canvas.height);
    player1();
    player2();
    move();
    setTimeout("gameloop()", 1000 / 60);
    Ball();
    moveBall();
}
onkeydown = handleKeydown;
onkeyup = handleKeyup;
gameloop();
```

Im Interview bestätigte sich, dass der Autor das Malen im Canvas und die Handhabung des Timers beherrscht und sein gesamtes Programm überzeugend erklären konnte. Wenn es mir gelungen wäre, ihn stärker zu motivieren, hätte er mit Sicherheit mehr erreichen können. Er hat aber eindeutig ein Niveau erreicht, das das Bestehen des Moduls rechtfertigt.

Optical Illusion

Bei dieser Anwendung handelt es sich um eine Canvasanimation, die eine optische Illusion erzeugt, indem sie Reihen von schwarzen Quadraten in entgegengesetzte Richtungen laufen lässt.

Die ursprünglich abgegebene Version bestand aus zweihundert fünfundsiebzig Zeilen JavaScript. Dies lag daran, dass der Autor kein Feld verwendete und somit für jedes einzelne der hundert acht Quadrate eine Malanweisung im Programm hatte. Hinzu kamen weitere hundert Zeilen für eine gigantische Verzweigungsstruktur, um die Randbehandlung für jedes einzelne Quadrat zu handhaben.

Diese Version konnte der Schüler ohne weiteres erklären. Ich hatte mir jedoch mehr erwartet und fragte ihn, ob es keine elegantere Möglichkeit der Implementierung gäbe. Nach kurzer Zeit kam er darauf, man könnte Felder verwenden. Also bat ich ihn dies zu verbessern.

Ich erhielt zwei Tage später die aktuelle, um zweihundert dreißig Zeilen kürzere, Version:

```
var Canvas = document.getElementById("canvas");
var context = Canvas.getContext("2d");
var width = Canvas.width, height = Canvas.height;

var arrleft = [-100, 0, 100, 200, 300, 400, 500, 600, 700];
var arrleftY = [0, 100, 200, 300, 400, 500];
var arrright = [-50, 50, 150, 250, 350, 450, 550, 650, 750];
var arrrightY = [50, 150, 250, 350, 450, 550];

setInterval(main, 1000 / 36);

function main() {
    clear();
    for (var l = 0; l < arrleft.length; l++) {
        for (var yl = 0; yl < arrleftY.length; yl++) {
            arrleft[l]++;
            context.fillStyle = "#ffffff";
            context.fillRect(arrleft[l], arrleftY[yl], 50, 50);
            if (arrleft[l] > width) {
                arrleft[l] = -100;
                arrleft[l]++;
                context.fillStyle = "#ffffff";
                context.fillRect(arrleft[l], arrleftY[yl], 50, 50);
            }
        }
    }
    for (var r = 0; r < arrright.length; r++) {
        for (var yr = 0; yr < arrrightY.length; yr++) {
            arrright[r]--;
            context.fillStyle = "#ffffff";
            context.fillRect(arrright[r], arrrightY[yr], 50, 50);
            if (arrright[r] < -50) {
                arrright[r] = 850;
                arrright[r]--;
                context.fillStyle = "#ffffff";
                context.fillRect(arrright[r], arrrightY[yr], 50, 50);
```

```
            }
        }
    }
}

function clear() {
    context.fillStyle = "#000000";
    context.fillRect(0, 0, width, height);
}
```

Im zweiten Bewertungsgespräch stellte sich heraus, dass er die Hilfe eines Kollegen beansprucht hatte, was für mich in Ordnung war, solange er das Programm im Detail erklären konnte. Anhand meiner Fragen stellte ich jedoch einige Unsicherheiten bezüglich der Manipulation von Feldern fest. Insbesondere fiel es ihm schwer, zumindest verbal, den Unterschied zwischen einem Feldelement und dem Feld klar darzustellen. Um ein genaueres Bild seines Verständnisses der Grundelemente von JavaScript zu bekommen, stellte ich dann ein paar Fragen, deren Beantwortung Funktionen mit Parametern erforderte. Dies klappte überhaupt nicht, so dass im Endeffekt aus dem kurzen Bewertungsgespräch, das wir über Skype führten, eine hundert zweiunddreißig-minütige Tutoratssitzung wurde, während der ich den Schüler bei der Programmierung von Funktionen mit Parametern unterstützte.

Fairerweise muss man berücksichtigen, dass es sich um einen zurückhaltenden Schüler handelt, dem ein Bewertungsgespräch vergleichsweise schwer fällt, was er auch selbst sagte. Da das Gespräch bestätigte, dass er die anderen Lerninhalte, inklusive Kontrollstrukturen, eindeutig beherrschte, bewertete ich die Arbeit insgesamt als ausreichend.

Comet

Hier handelt es sich um eine Kometenanimation. Mit den Pfeiltasten kann man die Bewegungsgeschwindigkeit oder -richtung der Kometen ändern. Mit der Enter-Taste kann man die Kometen mit dem Mauszeiger verbinden, so dass sie diesem folgen.

Das Programm besteht aus rund hundert dreißig Zeilen JavaScript. Der Autor hatte ein Skript im Internet gefunden und als Inspirationsquelle verwendet. Insbesondere die Berechnungen der Koordinaten, um die Animation der Kometen zu erzeugen, hat er übernommen. Es ist auch ersichtlich, dass er andere Teile aus anderen Quellen versucht hat zu integrieren. Die Integration ist jedoch nicht ganz gelungen, da er die Funktionsweise der integrierten Teile nicht komplett durchblickte. Daher läuft das Skript nicht im Firefox Browser und es befinden sich einige Fehler im Code, die der Autor im Interview nicht korrigieren konnte. Das Programm ist von mittlerer Komplexität und arbeitet

ohne Canvas. Stattdessen wird für jeden Kometen auf inkorrekte Weise ein entsprechend gestyltes `div`-Element erzeugt, wie man aus der Hauptfunktion erkennt:

```
function Comet() { //comet spacement
    var yBase = (ns) ? window.innerHeight / 4 : window.document.body.clientHeight / SpaceY;
    var xBase = (ns) ? window.innerWidth / 4 : window.document.body.clientWidth / SpaceX;
    for (i = 0; i < 14; i++) {
        var randCol = Math.round(Math.random() * 800); // randomize Color
        var layer = (document.layers) ? document.layers['n' + i] : me[i].style;
        layer.top = Ypos + yBase * Math.cos((currStep + i * q) / 12) * Math.cos(0.7 + currStep /
               200); // Comets Y-axe movement
        layer.left = Xpos + xBase * Math.cos((currStep + i * w) / 10) * Math.cos(8.2 + currStep /
               400); // Comets X-axe movement
        if (ns)
            layer.bgColor = Clrs[randCol];
        //apply Color
        else
            layer.background = Clrs[randCol];
    }

    currStep += step;
    setTimeout("Comet()", 5); //fluidity

    if (Speed > 0) { //Speed indicator
        document.getElementById("speed").innerHTML = "Speed: " + Speed + " x";
    }
    else
    {  // if too slow
        document.getElementById("speed").innerHTML = "Comets Stopped";
        Speed = 0;
        step = 0.00004883;
    }

    if (Speed > 19) { // if too fast
        document.getElementById("speed").innerHTML = "Don't be so Stupid!";
        Speed = 20;
        step = 51, 2;
    }
}
```

Der Schüler beherrscht die Grundlagen von JavaScript und konnte den Programmaufbau sowie die Funktionsweise detailliert erklären, bis auf einige der Zeilen, die er aus anderen Quellen übernommen hatte. Dies führte zum Punktverlust beim Interview und verhinderte eine bessere Bewertung.

Permanence Viewer

Diese Anwendung implementiert eine simple Bereitschaftsdienstverwaltung, d.h. eine Tabelle, in der man eintragen kann, wer wann Bereitschaft hat. Das Programm besteht aus rund hundert achtzig Zeilen JavaScript und ist optisch und technisch nicht ausgereift, wie man bei der ersten Benutzung sofort herausfindet.

Die Komplexität des Programms ist gering. Verschachtelte Kontrollstrukturen sucht man vergebens, genauso wie Datenvalidierung. Hundert zehn Zeilen Verzweigungsanweisungen zeigen, dass der Autor die Einsatzgebiete von Feldern noch nicht voll erkannt hat. Letztere werden zwar stellenweise verwendet, jedoch nicht da, wo sie wirklich erhebliche Vorteile bringen würden.

Hier ein kurzer, aber repräsentativer, Code-Ausschnitt:

```
for (var j = 0; j < arrUhr.length; j++) {
    uhrSelect.options[j].text = arrUhr[j];
}

function insert() {
    document.getElementById("mon1").innerHTML = mon1;
    document.getElementById("mon2").innerHTML = mon2;
```

Der Autor konnte das simple Programm überzeugend erklären, obwohl er nicht erkannte, weshalb nach dem ersten Klick auf den Knopf in allen unbenutzten Feldern der Text „undefined“ steht.

4.2.3.3 *Schlussfolgerungen*

Die Abschlussevaluation anhand der Lösung eines vom Schüler selbstbestimmten Problems, sowie einem Bewertungsgespräch, passt in meinen Augen sehr gut zu PBL, da es dem Schüler Gelegenheit gibt, seine Sach-, Methoden-, Sozial-, Personal- und Problemlösekompetenz (cf. 2.7) unter Beweis zu stellen, anstatt einfach Wissen in einer Prüfungssituation wiederzugeben.

Aus pädagogischer Sicht ist der Lerneffekt meines Erachtens deutlich höher, denn der Schüler setzt sich über einen längeren Zeitraum mit einer nichttrivialen Problemstellung aktiv auseinander, die er selbst gewählt hat und die ihn daher wahrscheinlich intrinsisch stärker motiviert als eine vom Lehrer vorgegebene Prüfungsaufgabe. Dies ist eine realistische Situation, wie sie im beruflichen Alltag häufig auftritt.

Der Lösungsprozess erfordert zahlreiche Perspektivenwechsel, den Transfer bestehenden Wissens und Könnens in einen neuen Kontext und somit die Festigung und Flexibilisierung der bereits aufgebauten Denkstrukturen und -muster. Darüber hinaus muss der Schüler für anspruchsvolle Problemstellungen seine Wissenslücken identifizieren und seine Fachkompetenz soweit weiter entwickeln, dass er das Problem lösen kann. Dies erfordert wiederum den Einsatz und die Weiterentwicklung seiner Personal-, Medien-, und Sozialkompetenz.

Die realisierten Problemlösungen illustrieren die erreichte Handlungskompetenz in vielen Fällen recht eindrucksvoll. Dies freut mich natürlich als Lehrer, allerdings darf man den Impakt auf das

Selbstbewusstsein und die mentale Einstellung der Schüler nicht übersehen. Sie haben sich und der Welt bewiesen, dass sie größere Problemstellungen erfolgreich bewältigen können. Dadurch hat sich eine klar erkennbare „Can Do“-Mentalität entwickelt, die die vorherige demotivierte Einstellung (man muss wissen, dass diese Klasse letztes Jahr die Proteste gegen die Reform des technischen Unterrichts organisierte und den neuen Lehrplänen somit mehr als skeptisch gegenüber stand) nach und nach ersetzt hat.

Die Atmosphäre hat sich sehr deutlich gewandelt. Da ich diese Klasse bereits im Vorjahr hatte, kann ich dies gut beurteilen. Das allgemeine Interesse an der Programmierung ist gestiegen, selbst in der letzten Unterrichtsstunde waren zahlreiche Schüler damit beschäftigt, entweder an eigenen Programmierprojekten zu arbeiten oder die Skripte anderer im Internet zu studieren. Darüber hinaus haben mehrere Schüler ihre Vorfreude auf nächstes Jahr geäußert, wo die serverseitige Programmierung und Datenbanken auf dem Programm stehen.

Zufrieden bin ich natürlich nicht mit der Tatsache, dass ein Drittel der Klasse das Modul nicht bestanden hat. Auch die Tatsache, dass dies im Vergleich zu den anderen Schulklassen eine vergleichsweise niedrige Misserfolgsquote ist und der Schwierigkeitsgrad der Abschlussevaluation vergleichsweise hoch war, kann nicht darüber hinwegtrösten.

Ich glaube, dass unter Berücksichtigung der in 4.1.10 beschriebenen Empfehlungen die Erfolgsquote hätte höher ausfallen können. PBL eignet sich meines Erachtens hervorragend für differenzierten Unterricht, allerdings benötigt man eine umfangreiche Ressourcensammlung, insbesondere an Lernmaterial, das auf schwächere Schüler zugeschnitten und dementsprechend sauber strukturiert und mit vielen angemessenen Übungen versehen ist. Schwächere Schüler benötigen eine stärkere Unterstützung durch die Lehrkraft.

5 Daten- und Faktorenanalyse

Aufgrund des umfangreichen Lehrprogramms, der ganz neuen Module, der unterschiedlichen Lehrstile, Schülerausgangspositionen etc. war es schwierig, die Ergebnisse präzise zu messen und alle Faktoren zu berücksichtigen.

5.1 *Erstes Semester*

5.1.1 Schülerfeedback

Obwohl ich während des Semesters des Öfteren die Meinung der Klasse zum Unterrichtsverlauf eingeholt hatte, schien es mir wichtig, am Ende des Semesters eine anonyme und detailliertere Umfrage durchzuführen.

Ich bat die Schüler, mittels eines Google Doc Fragebogens, 25 Aussagen zu bewerten sowie 6 offene Fragen zu beantworten. Ich bat die Kollegen, die CLISS1 auf den anderen Klassen im Land unterrichteten, ihre Schüler ebenfalls den Fragebogen ausfüllen zu lassen. Einige von ihnen hatten PBL versucht einzusetzen, waren aber nach einiger Zeit wieder auf traditionelleren Unterricht umgestiegen.

Ich erhielt Antworten von 54 Schülern, davon 20 aus meiner Klasse und 34 von 4 anderen Klassen.

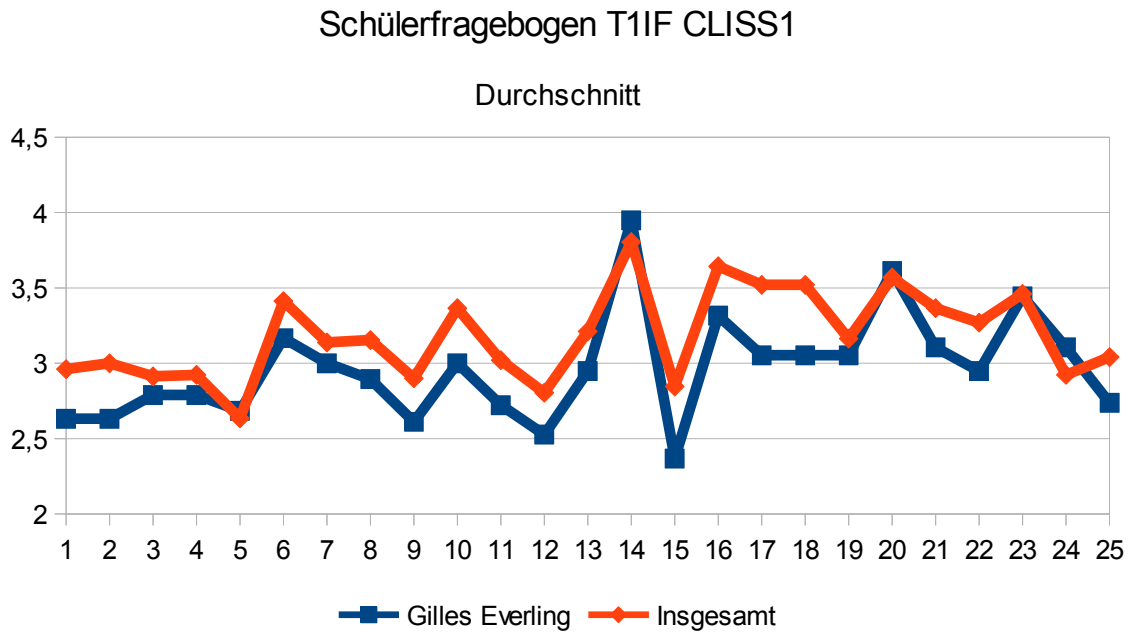
5.1.1.1 *Analyse der Aussagenbewertung*

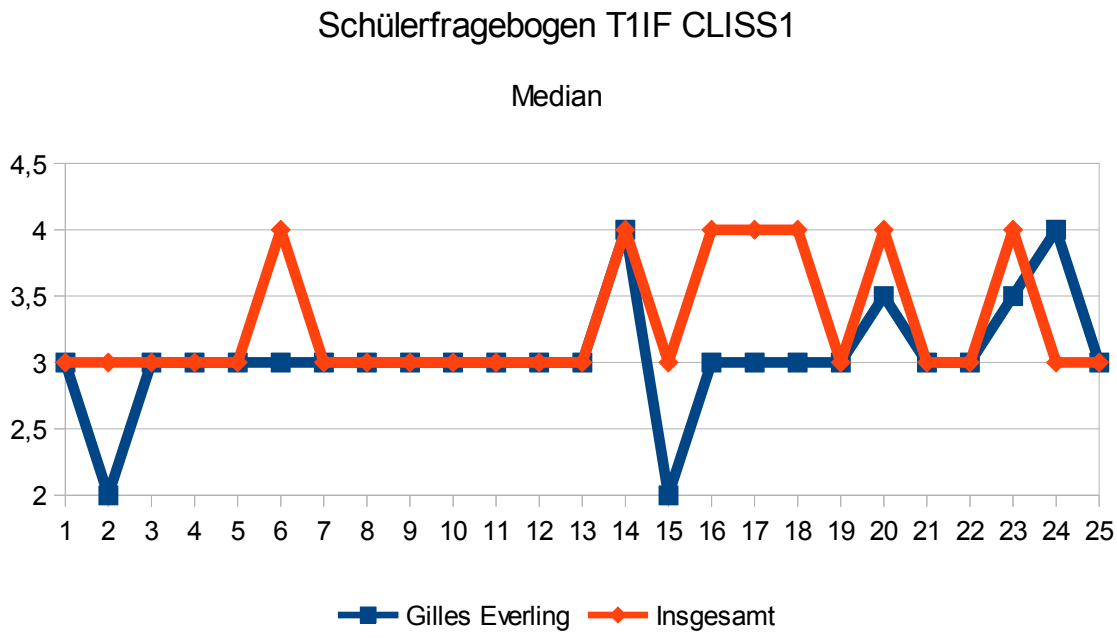
Für jede der folgenden 25 Aussagen sollte ein Wert zwischen 1 und 5 angegeben werden (1 - stimmt gar nicht, 2 - stimmt größtenteils nicht, 3 - stimmt teilweise, 4 - stimmt größtenteils, 5 - stimmt ganz):

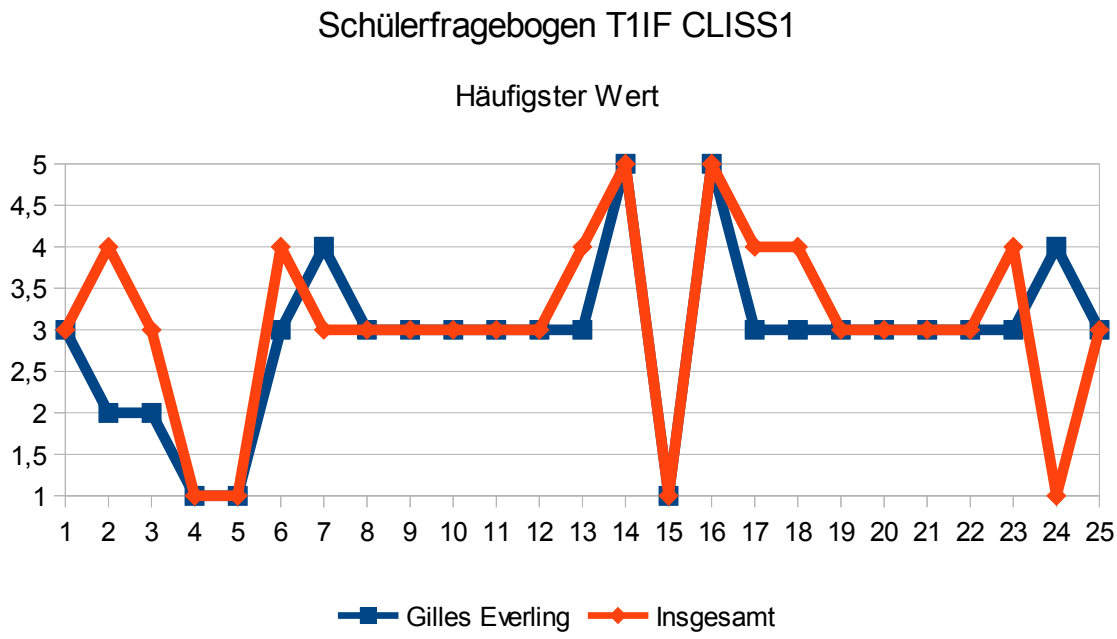
1. Die Problemstellungen haben mich motiviert.
2. Die Übungsaufgaben haben mich motiviert.
3. Die automatische Rückmeldung bei den Übungsaufgaben hat meinen Lernprozess gefördert.
4. Ich habe mindestens soviel für diesen Kurs wie für einen herkömmlichen Kurs gearbeitet.
5. Ich fühlte mich überfordert.
6. Ich habe ein klares Verständnis der grundlegenden Konzepte von JavaScript.

7. Ich kann dieses Verständnis zur Lösung noch nicht behandelter Probleme einsetzen.
8. Meine Fähigkeit, Probleme zu analysieren, hat sich verbessert.
9. Ich kann komplexe Skripte erstellen und optimieren.
10. Ich kann fehlerhafte Programme mittels Debuggertools analysieren.
11. Meine Fähigkeit, Informationen selbstständig in Referenzen nachzuschlagen und einzusetzen, hat sich erhöht.
12. Mein Verständnis meines eigenen Lernprozesses hat sich verbessert.
13. Ich gehe systematischer an die Lösung eines Problems heran.
14. Ich arbeite gerne in einer Gruppe.
15. Ich arbeite am liebsten alleine.
16. Ich hätte die Gruppenarbeit am liebsten fortgesetzt.
17. Das Gespräch mit anderen hat mir geholfen, Probleme besser zu verstehen.
18. Das Gespräch mit anderen hat mir geholfen, Probleme besser zu lösen.
19. Ich kann die Lösung eines Problems effektiv planen.
20. Ich kann meinen Lösungsansatz erklären und begründen.
21. Ich kann meine eigenen Lernziele bestimmen und formulieren.
22. Ich habe mein Wissen und meine Fähigkeiten voll eingesetzt.
23. Ich habe ein klares Verständnis meiner Stärken und Schwächen bezüglich des Lehrplans entwickelt.
24. Ich bin zufrieden mit dem was ich erreicht habe.
25. Ich bevorzuge problembasierten gegenüber herkömmlichem Unterricht.

Die folgenden Grafiken fassen die Bewertung der 25 Aussagen zusammen:







Die detaillierten Statistiken befinden sich in 9.3.1.

Folgendes fällt auf:

1. Die durchschnittliche Standardabweichung liegt bei 1,2 für alle Antworten und bei 1,3 für die Antworten meiner Klasse. Angesichts einer Skala von 1 – 5 zeigt dies eine sehr hohe Variabilität. Die Meinungen der einzelnen Schüler, zu jeder der 25 Aussagen, gehen also weit auseinander.
2. Die Durchschnittsbewertung meiner Klasse beträgt 3,0 gegenüber 3,3 für die anderen Klassen. Bei den ersten zwei Aussagen beträgt der Unterschied zwischen den anderen und meiner Klasse jeweils 0,6 Punkte. Dies betrachte ich als ein sehr ernüchterndes Ergebnis. Hier spielen sicherlich die Tatsache, dass die Übungsaufgaben vor den Problemstellungen hätten durchgeführt werden sollen, eine Rolle (cf. 4.1.7).
3. Die automatische Rückmeldung bei den Übungsaufgaben (Aussage 3), für deren Umsetzung ich viel Aufwand betrieben habe, scheint, entgegen meiner Erwartung und Beobachtung, wenig Einfluss auf den Lernprozess gehabt zu haben.
4. Aussage 5 bzgl. Überforderung wurde nur mit 2,7 bewertet. Die Aussagen (6 - 13) bzgl. der entwickelten Fähigkeiten und des Verständnisses werden von meiner Klasse allerdings auch nur mit 2,9 und von den anderen mit 3,3 bewertet. Aus dieser Trendlosigkeit lässt sich

schwer Rückschlüsse ziehen. Wenn die Problemstellungen und Übungsaufgaben nicht motivierend waren, die Schüler sich nicht überfordert fühlten und trotzdem nach eigener Aussage nicht wirklich die Zielkompetenzen erreicht haben, was müsste dann geändert werden? Ein Hauptschwerpunkt des gesamten Kursaufbaus war gerade die Förderung der Motivation. Dies scheint laut dieser Bewertung fehlgeschlagen zu sein.

5. Was die Metakognition (Aussagen 19 - 21) betrifft, so fällt auf, dass verhältnismäßig viele Schüler der Meinung sind, ihren Lösungsansatz erklären und begründen zu können.
6. Die höchste Zustimmung erhielt die Aussage (Nummer 14) zur Gruppenarbeit und die niedrigste (jedoch nur in meiner Klasse) diejenige zur Einzelarbeit (Nummer 15). Auch die vergleichsweise hohe Bewertung von Aussage Nummer 16 bestätigt den Eindruck, dass Gruppenarbeit insgesamt geschätzt wird. Allerdings ist die Überzeugung, dass das Gespräch mit anderen bei der Lösung und dem Verständnis der Probleme geholfen hat (Nummern 17 und 18) bei meiner Klasse mit einer Bewertung von 3,1 deutlich geringer ausgeprägt als bei den anderen Klassen mit 3,7.

Um dieser Bewertung auf den Grund zu gehen, bat ich die Klasse, mir bei meinem Verständnis zu helfen. Das Ergebnis des Gesprächs:

- Alle stimmten meiner Feststellung zu, dass ich anfangs zuerst die Grundlagen hätte erklären und die Klasse sie anhand der Übungen vertiefen lassen sollen, ehe die Problemstellungen bearbeitet wurden.
- Die Aussage zur automatischen Rückmeldung bei den Übungsaufgaben schien von einigen Schülern nicht richtig verstanden worden zu sein.
- Mehrere Schüler meinten, dass viele den Fragebogen nicht ernsthaft ausgefüllt hätten.

5.1.1.2 *Analyse der Antworten auf die 6 offenen Fragen*

Die offenen Fragen wurden erwartungsgemäß nur teilweise und nicht immer mit dem erhofften Ernst und Tiefgang beantwortet. Nichtsdestotrotz lassen sich einige Tendenzen erkennen.

Wo glaubst du das, was du in CLISS1 gelernt hast, einsetzen zu können?

Bei den Antworten auf diese Frage fällt auf, dass die meisten Schüler die Anwendung der in CLISS1 gelernten Materie lediglich in der Entwicklung von Webseiten sehen. Nur etwa zehn Pro-

zent scheinen erkannt zu haben, dass sie die Grundlagen der Programmierung erlernen, die sie weit über die Webseitenentwicklung hinaus werden einsetzen können.

Welche Aspekte des Unterrichts haben dir gut gefallen? Begründe wenn möglich.

Am häufigsten wird hier der Aspekt der Gruppenarbeit positiv erwähnt. Auch die Übungen wurden mehrfach erwähnt.

Welche Aspekte des Unterrichts haben dir nicht gefallen? Begründe wenn möglich.

Viele Schüler beklagten hier, dass sie sich allein gelassen fühlten und sie es gewohnt waren, alles von ihrem Lehrer erklärt zu bekommen.

Wie bist du bei der Lösung der Probleme vorgegangen?

Die meisten Schüler gaben hier an, systematisch vorgegangen zu sein, indem sie das Problem analysiert und dann im Internet oder im Buch recherchiert haben, um es zu lösen.

Wie bist du bei der Lösung der Übungsaufgaben vorgegangen?

Die Antworten auf diese Frage unterschieden sich nicht nennenswert von den Antworten auf die vorhergehende Frage.

Zusätzliche Anmerkungen

Der Fragebogen sah auch ein Feld vor um zusätzliche Anmerkungen zu machen. Dies wurde jedoch von fast niemandem genutzt und brachte keine neuen Einsichten.

5.1.2 Lehrerfeedback

Aus der Rückmeldung der anderen Lehrer lässt sich erkennen, dass das Fach für die meisten Schüler recht schwierig ist. Das Lehrprogramm wurde als zu umfangreich betrachtet und konnte von niemandem komplett behandelt werden.

Einigkeit herrscht auch darüber, dass das Lehrbuch als solches unbrauchbar ist und höchstens als Referenz dienen kann.

5.1.3 Leistungsergebnisse

Auf den anderen Klassen waren die Ergebnisse wie folgt: 1 x 0%, 2 x 46% und 1 x 63%. Die durchschnittliche Erfolgsquote lag somit bei rund 39%.

5.2 *Zweites Semester*

5.2.1 Schülerfeedback

Wie im ersten Semester führte ich eine anonyme Umfrage durch. Ich bat die Schüler, mittels eines Google Doc Fragebogens, 23 Aussagen zu bewerten sowie 4 offene Fragen zu beantworten. Ich bat die Kollegen, die CLISS2 auf den anderen Klassen im Land unterrichteten, ihre Schüler ebenfalls den Fragebogen ausfüllen zu lassen.

Ich erhielt Antworten von 73 Schülern, davon 20 aus meiner Klasse und 53 von 5 anderen Klassen.

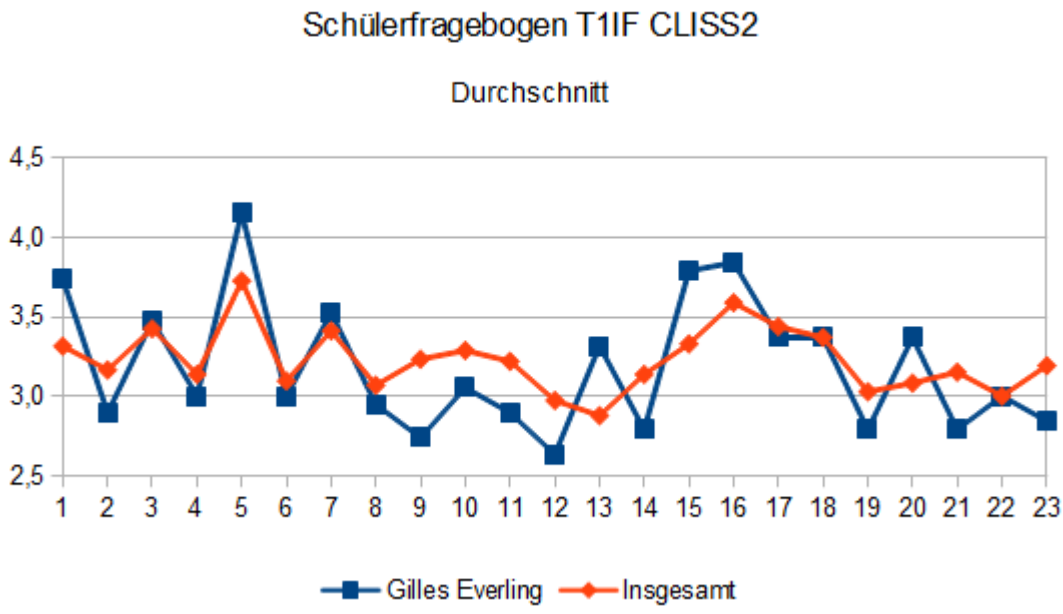
5.2.1.1 *Analyse der Aussagenbewertung*

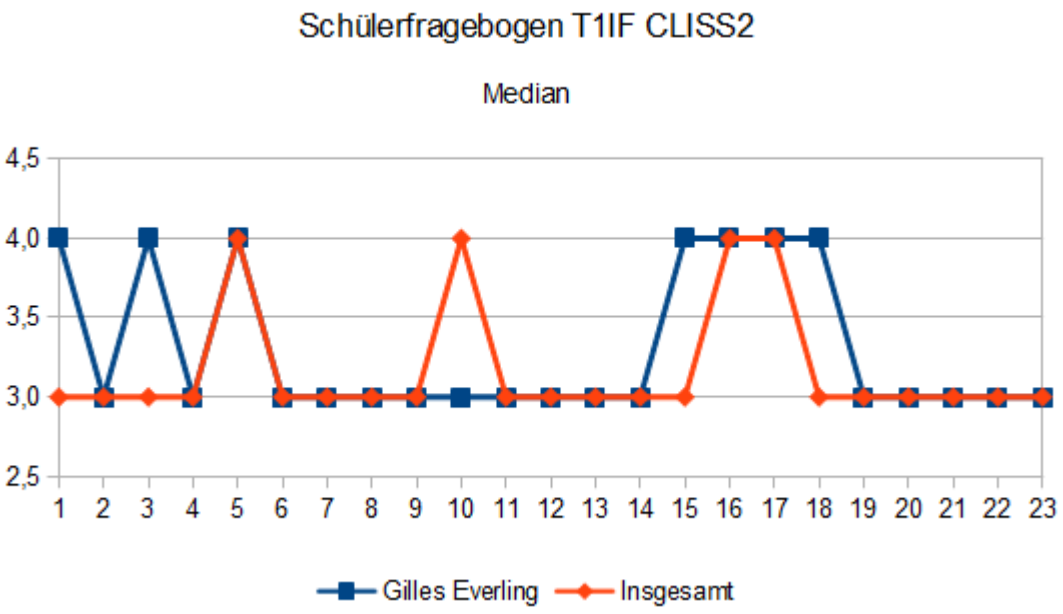
Für jede der folgenden 23 Aussagen sollte ein Wert zwischen 1 und 5 angegeben werden (1 - stimmt gar nicht, 2 - stimmt größtenteils nicht, 3 - stimmt teilweise, 4 - stimmt größtenteils, 5 - stimmt ganz):

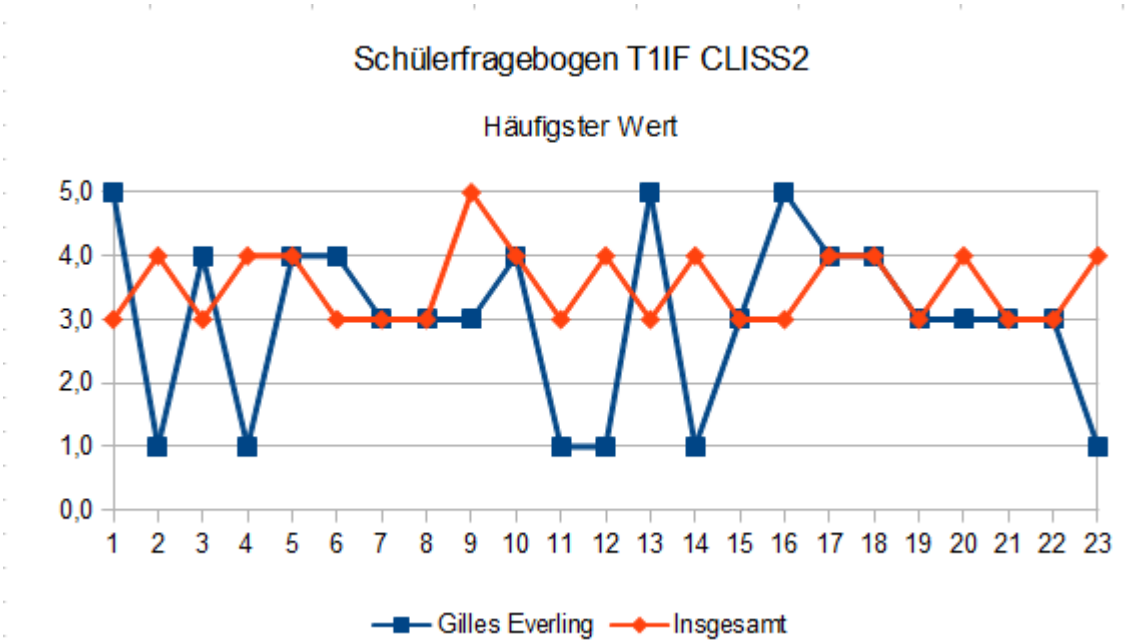
1. Ich kann Probleme in Teilprobleme zerlegen und Funktionen erstellen, welche diese lösen.
2. Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt.
3. Ich beherrsche das Erstellen von interaktiven Webseiten.
4. Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt.
5. Ich kann vorgefertigte Codelösungen an meine eigene Webseite anpassen und einbinden.
6. Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt.
7. Ich arbeite selbstständig und motiviert.
8. Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt.

9. Meine Fähigkeit konstruktiv in einer Gruppe zu arbeiten hat sich erhöht.
10. Ich habe meinen eigenen Stil entwickelt um Probleme zu analysieren und zu lösen und bin mir dessen bewusster geworden.
11. Meine Fähigkeit zu erkennen, welche Informationen mir fehlen und diese gezielt und effizient zu finden hat sich deutlich verbessert.
12. Ich habe ein besseres Verständnis für meinen eigenen Lernstil entwickelt.
13. Die behandelten Problemstellungen haben mich motiviert.
14. Mein Selbstvertrauen auch anspruchsvolle Problemstellungen lösen zu können hat sich gesteigert.
15. Der Lehrer war hauptsächlich als Instruktor tätig und hat mittels Vorlesungen Wissen vermittelt.
16. Der Lehrer war hauptsächlich als Tutor tätig und hat mich dabei unterstützt, meine eigenen Lernprozesse besser zu verstehen und zu entwickeln.
17. Ich habe hauptsächlich mittels Recherche im Internet und in Büchern die mir fehlenden Informationen gefunden.
18. Ich habe hauptsächlich durch das Gespräch mit meinen Klassenkollegen die mir fehlenden Informationen gefunden.
19. Durch die Diskussionen in der Gruppe habe ich andere Perspektiven kennen gelernt, die mich voran gebracht haben.
20. CLISS2 hat mir viel Spaß gemacht.
21. Ich habe hart gearbeitet.
22. Ich bin stolz auf das was ich erreicht habe.
23. Ich habe mir dieses Jahr die Grundlagen der Programmierung erarbeitet, auf denen ich in den nächsten Jahren aufbauen werde.

Die folgenden Grafiken fassen die Bewertung der 23 Aussagen zusammen:







Die detaillierten Statistiken befinden sich in 9.3.2.

Folgendes fällt auf:

1. Die Variabilität ist erneut recht hoch mit einer durchschnittlichen Standardabweichung von 1,3.
2. Die Durchschnittsbewertung meiner Klasse beträgt 3,2 und ist identisch mit der Durchschnittsbewertung der anderen Klassen.
3. Die Aussagen 1 und 5 werden von meiner Klasse um 0,5 respektive 0,6 Durchschnittspunkte höher, während die Aussagen 2 und 6 um 0,4 respektive 0,1 Punkte niedriger bewertet werden. Meine Klasse scheint also stärker davon überzeugt zu sein, bereits zu Beginn des Kurses über eine höhere Problemlösekompetenz sowie eine ausgeprägtere Fähigkeit zur Integration existierender Codelösungen verfügt und diesbezüglich weniger dazugelernt zu haben.
4. Die Aussagen 9 bis 12, 14, 19, 21 und 23 sind von meiner Klasse deutlich niedriger bewertet worden. Insbesondere bei den Aussagen 11 und 14 erstaunt mich dies, da nach meinen Beobachtungen sich diese Aspekte bei den meisten Schülern stark verbessert haben. Interessant finde ich, dass die Klasse im Durchschnitt laut Aussage 21 nicht der Meinung ist, hart gearbeitet zu haben. Ich vermute, dass dies mit der Aussage 20 zusammenhängt, wo meine Klasse um 0,4 Punkte stärker der Meinung war, viel Spaß gehabt zu haben. Was Spaß macht wird weniger als harte Arbeit empfunden. Dies wird durch die Bewertung der Aussage 13 bestä-

tigt, die darauf hindeutet, dass die Problemstellungen die Schüler motiviert haben. Dies entspricht auch meinem Erleben des zweiten Semesters, wo eine sehr konstruktive und positive Atmosphäre herrschte (cf. 4.2). Enttäuschend wiederum, dass meine Klasse deutlich weniger die Meinung teilt, sich die Grundlagen der Programmierung erarbeitet zu haben.

5. Die höhere Zustimmung mit Aussage 16 seitens meiner Klasse entspricht tendenziell meinen Erwartungen, obwohl ich hier eine größere Differenz erwartet hätte. Dass jedoch meine Klasse deutlich stärker der Meinung ist, ich sei hauptsächlich als Instruktor tätig gewesen, verleitet mich dazu zu bezweifeln, dass die Aussagen von einer Reihe von Schülern richtig verstanden wurden, denn meine Vorlesungsaktivitäten waren verschwindend gering.

5.2.1.2 *Analyse der Antworten auf die 4 offenen Fragen*

Die offenen Fragen wurden erwartungsgemäß nur von einem Teil der Schüler beantwortet.

Folgende Aspekte des Unterrichts haben mir gut gefallen

Am häufigsten wird hier die selbstständige Arbeit erwähnt. Auf den anderen Klassen wird auch die Gruppenarbeit positiv vermerkt.

Folgende Aspekte des Unterrichts haben mir nicht gefallen

Einige Schüler meiner Klasse haben zu viel Lärm bemängelt, obwohl andere positiv hervorgehoben haben, dass sich dies gegenüber dem ersten Semester deutlich verbessert habe. In den anderen Klassen wurden zum Teil zu viele Aufgaben respektive ungenügende Erklärungen bemängelt.

Folgendes würde ich am Unterricht ändern

Am häufigsten werden hier mehr und bessere Erklärungen verlangt.

Folgendes möchte ich noch anmerken

Es freute mich, dass 4 Schüler aus meiner Klasse schrieben, dass sie Spaß hatten und sich auf das nächste Jahr freuen. Auf den anderen Klassen gab es teils sehr positive und teils sehr kritische Bemerkungen zum Unterricht im Stile von „ich habe viel dazu gelernt“ bis zu „der Kurs ist nicht nötig und viel zu schwer“.

5.2.2 Lehrerfeedback

CLISS2 wurde im Wesentlichen als Wiederholung und Festigung von CLISS1 gesehen, was es ja auch ist. Dies half einer Reihe von Schülern, insgesamt jedoch reichte es bei vielen Schülern nicht, um das Modul zu bestehen.

5.2.3 Leistungsergebnisse

Auf den anderen Klassen waren die Ergebnisse wie folgt: 1 x 0%, 1 x 39%, 1 x 46% und 1 x 73%. Die durchschnittliche Erfolgsquote lag somit bei rund 40%.

6 Empfehlungen

Nachdem ich in den beiden vorherigen Kapiteln meine Erfahrungen mit PBL als Unterrichtsmethode bei der Einführung der Programmierung auf einer T1IF Klasse beschrieben habe, möchte ich hier meine Empfehlungen erläutern.

6.1 *Problemstellungen*

Problemstellungen bilden das Fundament von PBL. Die in 2.6 detaillierten theoretischen Aspekte spielen in der Praxis eine große Rolle. Wie in 4.1 geschildert ist das Risiko groß, dass man Schüler ohne PBL-Erfahrung mit Problemstellungen überfordert, auf deren Lösung sie kognitiv noch nicht vorbereitet sind.

In meinen Augen ist es wichtig, mit stark strukturierten Problemfällen geringer Komplexität zu beginnen und parallel den Schülern eine starke Unterstützung bei der Entwicklung ihrer Problemlösekompetenz zu bieten. Auf diesen wichtigen Punkt gehe ich in 6.4 ein.

Die Schüler, die ein ausreichendes Kompetenzniveau erreicht haben, können nach und nach offenere und komplexere Problemstellungen in Angriff nehmen und so die relevanten Kompetenzen parallel weiter entwickeln, d.h. nicht nur ihr Fachwissen ausbauen, sondern gleichzeitig auch verstärkt ihren eigenen Lernprozess analysieren und Problemschemata aufbauen, die es ihnen erlauben werden, offenere und anspruchsvollere Probleme immer effektiver zu lösen.

Um dies realisieren zu können, benötigt man eine Vielzahl von Problemstellungen der unterschiedlichsten Struktur- und Komplexitätsgrade, um allen Schülern gerecht werden zu können. Selbstverständlich müssen die Probleme auch von den anvisierten Lerninhalten her stimmig aufeinander und auf das Lehrprogramm abgestimmt sein. Aus Erfahrung weiß ich, dass die Erstellung einer guten Problemstellung viel Zeit beansprucht. Der Zeitaufwand, um einen kompletten, erfolgversprechenden, PBL-Kurs aufzubauen, ist allein schon aufgrund der Vielzahl von erforderlichen Problemstellungen enorm. Daher drängt es sich geradezu auf, eine Problemdatenbank zu erstellen, in der zu jedem Problem auch die relevanten Metadaten gespeichert sind, wie Grad der Strukturierung, Komplexität, Lernziele, Kontext, Voraussetzungen bezüglich Fachwissen etc. Mit der Zeit würde es so möglich, mittels PBL eine echte Unterrichtsdifferenzierung zu realisieren, indem jeder Schüler die für ihn gerade angemessene Problemstellung bearbeitet und somit optimal gefördert wird.

Ideal wäre es hierfür ein automatisiertes System zu entwickeln, mit anderen Worten eine maßge-

schneiderte Lernumgebung.

6.2 *Lernumgebung*

Die Lernumgebung spielt meines Erachtens auch bei PBL eine wichtige Rolle. Sie soll insbesondere den schwächeren Schülern Ressourcen, z.B. Verweise auf geeignete Webseiten und Bücher, zur Verfügung stellen, um ihren Lernprozess zu unterstützen. In dieser Hinsicht sind Entwicklungen von, wie ich meine, fundamentaler pädagogischer Bedeutung im Gange. Sogenannte massive open online course (MOOC) Anbieter arbeiten an maßgeschneiderten Plattformen, um didaktisch und inhaltlich hervorragendes Material einer sehr großen Zahl Studenten weltweit zugänglich zu machen. Ich habe bereits Kurse von den wichtigsten Anbietern ganz oder teilweise absolviert und bin nachhaltig beeindruckt. Plattformen wie http://code.edx.org/ stellen dabei die ganze Infrastruktur als open source zur Verfügung, unter anderem auch die automatischen Bewertungsmodule, die mittels künstlicher Intelligenz versuchen, die Bewertung soweit wie möglich zu automatisieren.

Die CLISS-Website besteht aus über zehntausend Zeilen HTML, CSS, JavaScript und PHP und stellt in meinen Augen erst den Anfang von dem dar, was erforderlich ist, um eine effiziente PBL-Umgebung aufzubauen, die alle Schüler maximal fördert.

Wie in 4.1.7 erwähnt habe ich viel Aufwand getrieben, um die automatische Evaluation für alle Trainingsübungen zu programmieren, damit ein Schüler eine sofortige Rückmeldung bezüglich der Korrektheit seiner Lösung bekommt. Um dem Schüler eine komfortable Codeeingabemöglichkeit, inklusive auto completion und syntax highlighting, zu bieten, habe ich eine JavaScript-Komponente namens CodeMirror für meine Zwecke angepasst.

Bis jetzt musste der Schüler mir aber immer noch seine korrekte Lösung, respektive die Bestätigung durch den automatischen Validator, zeigen, damit ich darüber Buch führen konnte. In einem nächsten Schritt möchte ich die Website so ausbauen, dass jeder Schüler sein eigenes Login hat und damit seine persönliche Lernumgebung von überall erreichen kann. Hier wird er dann einen Überblick über die bereits gelösten Übungen und Problemstellungen haben und das System wird ihm automatisch passende Problemstellungen, abhängig von seinen bisherigen Lernfortschritten, anbieten. Hier könnte man auch ein Punktesystem einführen, so dass ein gewisser Wettkampf in der Klasse entsteht, um die meisten Punkte, d.h. das höchste Kompetenzniveau, zu erreichen. Meine Erfahrungen mit codecademy.com und ähnlichen Websites, die auf das Erlernen der Programmierung ausgerichtet sind, zeigen, dass dies sehr motivierend wirken kann. Hier könnte man auch eine Art Progress

Bar implementieren, die anzeigt, wie weit es noch bis zum Bestehen des Moduls ist etc. Somit hätten der Schüler, die Lehrkraft und eventuell sogar die Eltern jederzeit den Überblick.

Einen großen Pluspunkt würde die Verbindung zu einer umfangreichen Problemdatenbank, wie in 6.1 beschrieben, darstellen, so dass die Lernumgebung jedem Schüler zum Teil andere Problemstellungen anbieten würde, wodurch das Kopieren von Lösungen reduziert werden könnte. Allerdings sollte man dies nicht zu weit treiben, um nicht die Gruppendynamik und somit die Entwicklung der Sozialkompetenz zu gefährden.

Im Sinne eines differenzierten Unterrichts halte ich Videotutorials für sehr wichtig, da sie ein alternatives Lernmedium darstellen und somit für einige Schüler attraktiver sind. Außerdem sorgen sie für Abwechslung und erlauben es, Dinge sehr viel anschaulicher und verständlicher darzustellen als mittels einer Textbeschreibung. Berücksichtigt werden sollten die Erkenntnisse aus 4.1.8, d.h. kurze und prägnante Videos zu spezifischen Themen.

Steigern sollte man auch die Abwechslung der Übungsformate, z.B. mit Kreuzworträtseln.

6.3 *Lehrbuch/Kursunterlage*

Ein gutes Lehrbuch oder eine maßgeschneiderte Kursunterlage sind auch in einem PBL-Kurs sehr wichtig. Da das offizielle Lehrbuch didaktisch nicht überzeugen konnte und obendrein mit dreiwöchiger Verspätung ankam, hatte ich, wie in 4.1.4 beschrieben, ein Tutorial entwickelt, sozusagen als für PBL maßgeschneiderte Kursunterlage. Ich habe jedoch ein Buch gefunden, das meines Erachtens pädagogisch und fachlich hervorragend ist und obendrein sehr gut zum PBL-Ansatz passt: „Head First HTML5 Programming“ von Eric Freeman und Elisabeth Robson. Von allen Büchern, die ich in den Sommerferien studiert habe, um JavaScript zu erlernen, ist dies dasjenige, das mir mit Abstand am meisten Freude bereitet und mich am effektivsten vorangebracht hat. Das Buch ist so unterhaltsam geschrieben und die Materie wird so effektiv vermittelt, dass man nicht aufhören kann es zu studieren, bis man die letzte Seite erreicht hat. Vom Anfang bis zum Ende werden sehr originelle und anspruchsvolle Problemstellungen in einem Spiralablauf (cf. 3.5) gelöst, die einen neugierig machen und sehr stark motivieren. Dabei werden auch fortgeschrittene Techniken, die man in anderen Büchern meist vergeblich sucht, sehr effektiv vermittelt.

Dass dieses Buch mich derart überzeugt, ist kein Zufall, denn es ist gezielt auf die Förderung der Metakognition ausgelegt. Und ganz im Sinne von PBL haben die Autoren erkannt, dass dies über motivierende Problemstellungen zu erreichen ist, da diese die Synapsenaktivität fördern (cf. 2.2).

Daneben werden beide Gehirnhälften und unterschiedliche Lernstile angesprochen, intelligente Redundanz, viele Bilder und Geschichten sowie unterschiedliche Perspektiven verwendet. Und es gibt jede Menge Aufgaben unterschiedlichster Art und Problemstellungen zu lösen. Ich werde daher meine aktuelle T0IF um ihre Meinung zu diesem und einigen anderen Büchern bitten und falls es seitens der Klasse keine stichhaltigen Einwände gibt, werde ich dieses Buch nächstes Jahr für die CLISS-Module verwenden und die Website darauf abstimmen, so dass Lernumgebung und Lehrbuch sich optimal ergänzen.

6.4 *Metaskilltraining*

Wie in 6.1 beschrieben ist es insbesondere bei PBL-Anfängern notwendig, eine starke Unterstützung bei der Entwicklung ihrer Metakognition und Problemlösekompetenzen zu bieten. Die Lehrkraft sollte hierzu die Lösung von leicht offenen Problemstellungen von zunächst geringer Komplexität vorführen und dabei ihren Lösungsprozess verbalisieren. Diese Modellierung ist wichtig, um den Schüler nach und nach mit effektiveren Vorgehensweisen vertraut zu machen und ihm die Wichtigkeit davon vorzuführen, um ihm zu zeigen, dass das Analysieren der eigenen Vorgehensweise keine Zeitverschwendung darstellt, um den Lehrer zufrieden zu stellen, sondern für das eigene Erfolgserlebnis auf Dauer entscheidend ist.

Ich bin jedoch nicht mehr davon überzeugt, dass es, zumindest im Fach Informatik und in der Sekundarschule, sehr sinnvoll ist, von den Schülern zu verlangen, ihren Lösungsprozess zu dokumentieren. In den Worten Spitzers (cf. 2.2) wissen wir wenig und können viel. Dies wird insbesondere bei Informatikexperten sehr deutlich. Sie können sehr anspruchsvolle Probleme auf effektive Weise lösen, wissen aber meist nicht, wie sie es tun. Ihr Können ist das Ergebnis der jahrelangen Auseinandersetzung mit zahlreichen Problemstellungen. Deswegen ist Modellierung wohl auch die effektivste Form der Metaskillförderung durch die Lehrkraft (cf. 2.3).

Des Weiteren ist ein Sekundarstufenschüler nach meiner Erfahrung noch gar nicht in der Lage, seinen eigenen Lernprozess zu dokumentieren. Sehr viel wichtiger scheint mir, dass er sich mit den Problemstellungen auseinandersetzt und dadurch die erforderlichen Denkstrukturen nach und nach in seinem Gehirn entwickelt, die ihm das Können ermöglichen.

6.5 *Unterrichtsführung*

Ein PBL-Kurs unterscheidet sich in Bezug auf die Unterrichtsführung deutlich von einem lehrerzen-

trierten Ansatz. Das Hauptanliegen des Lehrers ist das Aktivieren und Fördern der Denkprozesse der Schüler.

Eine Grundvoraussetzung ist, dass allen Schülern angemessene Problemstellungen zur Verfügung stehen (cf. 6.1). Ist dies nicht der Fall, entstehen unweigerlich Frust und/oder Langeweile, die eine konstruktive Lernatmosphäre zunehmend unmöglich machen.

Die Lehrkraft muss darauf achten, allen Schülern die von ihnen benötigte individuelle Unterstützung zukommen zu lassen. Dies bedeutet nicht notwendigerweise, gleich viel Zeit mit jedem Schüler zu verbringen, sondern intern Buch darüber zu führen, wo welcher Schüler in Bezug auf seine Kompetenzentwicklung steht und welche Unterstützung für ihn gerade am angebrachtesten wäre. Es ist auch wichtig, die Lernstile und Persönlichkeit des Schülers zu berücksichtigen. Manche kommen hervorragend ohne externe Unterstützung zurecht, hier wäre eine aufgezwungene Unterstützung eher kontraproduktiv. Andere Schüler profitieren sehr stark vom regelmäßigen Austausch mit der Lehrkraft. Für diesen Aspekt ist das Fingerspitzengefühl der Lehrkraft entscheidend.

Bei großen Klassen ist es meines Erachtens unabdingbar, die besten Schüler zu Tutoren (cf. 4.1.5) zu „befördern“ damit die schwächsten Schüler ausreichend Unterstützung bekommen. Dies stellt zugleich eine valorisierende und herausfordernde Aktivität für die Tutoren dar, die dies nach meiner Erfahrung gerne und auch sehr gut machen. Sie fördern dadurch ihre eigene Metakognition und Sozialkompetenz und tragen zu einer Teamatmosphäre bei, die sich konstruktiv auswirkt.

Was den Stand der Kompetenzentwicklung eines jeden Schülers betrifft, wäre die Unterstützung durch eine weitgehend automatisierte Lernumgebung sehr hilfreich, da sie es erlauben würde, stets einen Überblick darüber zu haben, wo jeder einzelne Schüler steht und gezielt auf Lücken einzugehen. Durch die automatische Auswahl von angemessenen Trainingsaufgaben und Problemstellungen könnte ein differenzierterer Unterricht umgesetzt werden.

Ein nicht zu unterschätzender Aspekt stellt die Motivation des Lehrers dar. Ich habe das ganze Jahr über die Schüler auf interessante Entwicklungen und gut gemachte, in JavaScript programmierte, Anwendungen und Spiele aufmerksam gemacht, um ihnen zu zeigen, welche Möglichkeiten diese Programmiersprache eröffnet und so ihre Motivation zu schüren. Des Weiteren habe ich sehr viel Aufwand getrieben, um möglichst ansprechende Beispiele, Aufgaben, Problemstellungen und Evaluationsaufgaben sowie die ganze Lernumgebung zu entwickeln, um die Neugier des Schülers zu erwecken und seine Ansprüche an sich selbst möglichst hoch zu schrauben. Darüber hinaus habe ich meine Begeisterung für die dynamische Webprogrammierung stets zum Ausdruck gebracht, was

auch vom Zielpublikum positiv aufgenommen wurde. Dass dies seinen Effekt nicht verfehlt hat, konnte ich an der Begeisterung und an den zahlreichen Fragen der Klasse erkennen.

Wenn die Voraussetzungen stimmen, dann entsteht im PBL-Unterricht die typische Atmosphäre wie ich sie aus Unternehmen kenne, die an der Spitze der technologischen Evolution neue Lösungen entwickeln. Man spürt regelrecht die kognitive Aktivität einer motivierten und aktivierten Klasse. Die Klasse führt sich dann sozusagen selbst, da jeder Schüler intensiv mit der Lösung einer Problemstellung und somit der Entwicklung seiner Problemlösekompetenzen beschäftigt ist, was für alle eine sehr zufriedenstellende Aktivität darstellt. Die Lehrkraft kann sich voll darauf konzentrieren, die angemessene individuelle Unterstützung zu leisten und die Klasse ansonsten ungestört arbeiten zu lassen. Dieser Traumzustand wird selbstverständlich nicht in jeder Unterrichtseinheit erreicht, insbesondere nicht wenn die Lehrkraft noch nicht über ausreichend Erfahrung mit dem PBL-Ansatz verfügt. Mit der Zeit verbessert sich dies jedoch deutlich und die Lehrkraft verlässt den Klassensaal immer häufiger mit dem sehr zufriedenstellenden Gefühl, dass die Klasse mit Freude große Fortschritte erzielt hat.

6.6 *Evaluation*

Wie in 2.8 erwähnt ist es wichtig, die Evaluation dem PBL-Ansatz entsprechend zu gestalten, also nicht statisches Wissen abzufragen, sondern zu bewerten, inwiefern die vorgesehenen Lerninhalte im Rahmen einer ausgeprägten Handlungskompetenz beherrscht werden. Der natürlichste und in meinen Augen beste Weg, dies zu tun, besteht in der Konfrontation des Schülers mit unbekannten Problemstellungen, gegebenenfalls mit einem Bewertungsgespräch kombiniert. Man könnte jedoch auch andere Ansätze verwenden, z.B. die Portfoliomethode. Nicht vergessen sollte man jeden Schüler für gut gelöste Problemstellungen zu loben.

6.6.1 Formativ

PBL beinhaltet „von Haus aus“ eine kontinuierliche formative Evaluation, da der Schüler durch die Bearbeitung der Problemstellungen implizit Rückmeldung zu seinem Kompetenzstand bekommt. Wenn der Lehrplan auf Kompetenzen aufbaut, wie dies bei den CLISS Modulen der Fall ist, fällt es nicht schwer der Klasse zu vermitteln, was erforderlich ist um ein Modul zu bestehen. Dies ist wichtig, damit jeder Schüler immer weiß, wo er steht.

6.6.2 Summativ

Wie in 4.2.3 beschrieben bin ich von der Bewertung mittels der Lösung einer vom Schüler selbst bestimmten Problemstellung, kombiniert mit einem Bewertungsgespräch, überzeugt. Es stellt einen sehr realistischen Ansatz dar, wie man ihn in der beruflichen Praxis sehr oft antrifft, mit der Ausnahme, dass in der Berufspraxis oft der Kunde die Problemstellung vorgibt.

6.7 *Curriculum*

Um ein Maximum an Effektivität zu erreichen, wäre es sehr sinnvoll, wie in 3.1 beschrieben, PBL nicht in vereinzelten Modulen, sondern systematisch einzusetzen. Zum einen ist die Förderung der Handlungskompetenz ja ein Hauptziel der Reform der Berufsausbildung. Zum anderen bin ich auch davon überzeugt, dass die Vorteile von PBL erst durch den systematischen Einsatz richtig zum Tragen kommen und sich die anfängliche Lernkurve erst richtig bezahlt macht. Modulübergreifend könnten noch interessantere Problemstellungen in Angriff genommen werden und damit die Handlungskompetenz noch weiter gefördert werden. Zudem wird ein Schüler, der bis einmal das Erfolgsgefühl erlebt hat, das das Lösen einer anspruchsvollen Problemstellung mit sich bringt, die berechtigte Erwartung haben, dies auch in den anderen Modulen zu erleben.

Darüber hinaus sehe ich das Einsatzgebiet von PBL keineswegs auf die Berufsausbildung beschränkt. Allein schon die Tatsache, dass es bisher in der Informatik hauptsächlich auf Universitätsniveau eingesetzt wurde, zeigt, dass es durchaus im klassischen und technischen Sekundarunterricht eingesetzt werden könnte.

Seit drei Jahren unterrichte ich objektorientierte Programmierung mit Java auf 11TG- und dieses Jahr auch auf 12GE-Klassen. Hier beobachte ich einen generellen Motivationsmangel und werde immer wieder von Schülern gefragt, wieso dieses Fach denn für diese Klassen relevant sei. Diese Beobachtung wird von fast allen Kollegen landesweit geteilt. Dies sind natürlich, im Gegensatz zu TxIF-Klassen, keine Schüler, die Informatik als Vertiefung gewählt haben. Trotzdem bin ich überzeugt, dass PBL hier eine deutliche Verbesserung bringen würde. Diese Überzeugung wird durch meine eigenen Erfahrungen und die anderer untermauert.

Im Rahmen meiner Arbeit mit dem Titel „Travaux pratiques en informatique“ (cf. [4]) hatte ich auf einer 11TG-Klasse ein zweistündiges Experiment mit Lego Mindstorms durchgeführt. Die Klasse hatte noch nie mit Lego-Robotern gearbeitet, kannte die Entwicklungsumgebung nicht und hatte auch keine Erfahrung mit der zu verwendenden Programmiersprache. Trotzdem gelang es den meis-

ten Gruppen, die Problemstellungen zu lösen und dabei Spaß zu haben. Dies wurde inzwischen auf anderen Klassen mit Erfolg wiederholt.

Dieses Jahr habe ich Robocode (http://robocode.sourceforge.net) entdeckt, ein Programmierspiel, wo es darum geht, einen Panzer in Java zu programmieren, der dann in Turnieren gegen andere Panzer antritt. Ein Großteil meiner 11TG-Klasse hat daran mit Begeisterung in ihrer Freizeit gearbeitet. Der Erfolg dieser Mini-PBL-Versuche führt allerdings zu noch mehr Frustration, wenn man zum konventionellen Unterricht zurückkehrt. Dass es zu dieser Rückkehr nicht unbedingt einen zwingenden Grund gibt, zeigen O'Kelly und Gibson (cf. [28]) in ihrem Artikel zu RoboCode und PBL. Die Autoren kommen zum Schluss, dass RoboCode die ideale Problemstellung zum Erlernen der Programmierung darstellt. Eine Meinung, die ich nur teilen kann.

7 Schlussfolgerungen

Die Vorteile von problembasiertem Lernen sind meines Erachtens folgende:

- Die Motivation der Schüler tendiert insgesamt zu einem höheren Niveau, da sie mit anspruchsvollen und realistischen Problemstellungen konfrontiert werden, die sie ansprechen. Dies hat sich insbesondere während dem zweiten Semester gezeigt.
- Die Handlungskompetenz steigt sehr stark, da nicht kontextloses Wissen vermittelt wird, sondern die zur Lösung offener und komplexer Problemstellungen notwendigen Denkstrukturen und -muster erarbeitet werden. Die Abschlussevaluation, anhand einer selbsterstellten Problemstellung, hat, wie ich meine, eindrucksvoll gezeigt, dass dies kein Wunschdenken, sondern durchaus realisierbar ist. Ich kann dies auch im Vergleich zu den „konventionellen“ Programmierkursen, die ich unterrichtet habe und noch immer unterrichte, eindeutig bestätigen.
- Die Schüler lernen selbständiger zu arbeiten. Mit steigender Problemlöseerfahrung gewinnen sie zunehmend Vertrauen in ihre eigenen Fähigkeiten und erwarten nicht mehr alles von einer allwissenden Lehrkraft erklärt zu bekommen. Sie erweitern ihren Vorstellungshorizont in Bezug auf die eigenen Fähigkeiten und gewinnen Freude daran, zunehmend anspruchsvollere Ideen zu generieren und umzusetzen. Dies ist in meinen Augen eine Entwicklung, die ihr ganzes Leben positiv prägen wird.
- Es gibt wohl kaum ein überzeugenderes Argument für Gruppenarbeit als die Lösung eines anspruchsvollen Problems. Ein Schüler, der einmal erlebt hat, wie hilfreich die Ideen anderer sein können, wird seine Sozialkompetenz kontinuierlich weiter entwickeln. Dies hat sich im Laufe des Schuljahres auf ganz natürliche und offensichtliche Art und Weise ergeben. Während zu Beginn eher eine Einzelkämpfermentalität dominierte, konnte man zunehmends beobachten, wie immer mehr spontane Gruppendiskussionen rund um ein Problem stattfanden. Die Sozialkompetenz wird also eindeutig gefördert, was ganz im Sinne von PISA 2015 und dem beruflichen Alltag ist, wo Teamarbeit ganz oben auf der Tagesordnung steht.
- Die Methoden- und Medienkompetenz wird nachhaltig gefördert. Dies zeigt sich daran wie schnell die Schüler inzwischen Informationen finden, die ihnen bei der Problemlösung helfen. Des Weiteren können sie die Qualität der gefundenen Information und Webseiten besser

einschätzen.

- Die Auseinandersetzung mit den Problemstellungen trägt zur Entwicklung der Persönlichkeit der Lernenden bei, dies insbesondere beim Austausch in der Gruppe. So haben viele Schüler eine klarere Vorstellung von dem entwickelt, was sie interessiert und was sie in Zukunft erreichen wollen.
- Die Lehrkraft kann einen wirklich differenzierten Unterricht realisieren. Dies ist mir in meinem ersten PBL-Jahr nur teilweise gelungen. Allerdings habe ich im vorherigen Kapitel die in meinen Augen notwendigen Maßnahmen beschrieben, die diesen sehr wichtigen Aspekt weiter voranbringen können.
- In einer PBL-Umgebung zu arbeiten bereitet auch mir als Lehrkraft viel Freude, insbesondere nachdem die Startschwierigkeiten überwunden sind und eine sehr konstruktive Arbeitsatmosphäre entsteht, wo man die Synapsenaktivität beinahe hören kann.

Als Nachteile würde ich folgende Punkte nennen:

- Der Aufwand für einen erfolgreichen PBL-Kurs tendiert gegen unendlich, wobei dies nur eine leichte Übertreibung darstellt. Um eine ausreichende Anzahl angemessener Problemstellungen sowie eine entsprechende Lernumgebung aufzubauen, sind viele hundert Arbeitsstunden nötig. Ein gutes Lehrbuch zu finden, das mit PBL harmoniert, ist schwierig. Ein Fehlgriff führt hier zu deutlichem Frust und Zeitverlust und ist unbedingt zu vermeiden. Die Abstimmung der Problemstellungen und der Lernumgebung auf das Lehrbuch ist ebenfalls aufwendig.
- Die Lehrkraft und die Schüler brauchen Zeit, um sich auf die Methode einzustimmen, da sie im krassen Gegensatz zum traditionellen Unterricht steht, den die meisten von uns gewohnt sind.

Meine Erfahrung mit PBL erinnert mich sehr stark an meine dreizehnjährige Berufserfahrung in mittleren bis sehr großen Unternehmen im In- und Ausland im Hochtechnologie- sowie im Finanzbereich. In der Lage zu sein bessere Lösungen für alte Probleme, oder wirksame Lösungen für neue Problemstellungen, zu entwickeln stellt in der Berufspraxis einen entscheidenden Vorteil, sowohl für den Mitarbeiter wie auch für das gesamte Unternehmen, dar.

Leute, die das Selbstvertrauen haben, sich Herausforderungen zu stellen sowie innovative und effiziente Lösungen zu entwickeln, fallen positiv auf. Es ist meine Überzeugung, dass man nicht mit

diesem Selbstvertrauen geboren wird, sondern dass man es erlernt indem man beobachtet wie Experten Probleme lösen und man sich mit immer neuen Problemstellungen auseinandersetzt. Dies wird von der pädagogischen Forschung bestätigt und ist ein Kernpunkt der Problemlösekompetenzerhebung in den PISA-Studien. PBL passt hervorragend in das konstruktivistische Lernparadigma und integriert sich meines Erachtens sehr gut in die Reform des technischen Unterrichts.

Informatik eignet sich von Natur aus hervorragend für den PBL-Ansatz. Es gibt jedoch keinen erkennbaren Grund wieso PBL in der Mehrheit der anderen Fächer nicht ebenfalls erfolgreich eingesetzt werden könnte. Die empirischen Hinweise in diese Richtung häufen sich.

Bei breiterem Einsatz von PBL scheint mir die Entwicklung einer Problemdatenbank und einer fortgeschrittenen Lernumgebung angebracht. In den USA ist ein solcher Ansatz bereits in Arbeit (cf. http://inbloom.org/services).

Rückblickend auf das vergangene Jahr stelle ich fest, dass PBL meine hohen Erwartungen, nach anfänglichen Schwierigkeiten, weitgehend erfüllt hat und ich motivierter bin denn je, meinen PBL-Unterricht nach und nach auszubauen und weiter zu entwickeln, mit dem Ziel, dass 100% der Schüler die Module mit viel Freude und Handlungskompetenz bestehen.

8 Bibliographie

1. John D. Bransford, Ann L. Brown & Rodney R. Cocking: How People Learn: Brain, Mind, Experience, and School: Expanded Edition

   2000, Committee on Developments in the Science of Learning

2. Gilles Everling: Klassenführung im Informatikunterricht

   2011, Mémoire professionnel

3. Ministère de l'Éducation nationale et de la Formation professionnelle: Reform der Berufsausbildung: Leitfaden zur Entwicklung von Lehrplänen

   18.05.2012

4. Gilles Everling: Travaux pratiques en informatique

   2011, Pièce de la formation modulaire

5. PISA 2012 Field Trial Problem Solving Framework

   2010

6. PISA 2015 Draft Collaborative Problem Solving Framework

   2013

7. Rolf Dubs: Lehrerverhalten

   2009, Franz Steiner Verlag

8. Dr. Jochen Koubek: Wiki-Didaktik zwischen Kognitivismus und Konstruktivismus

   2008, Humboldt-Universität zu Berlin, Institut für Informatik

   http://waste.informatik.hu-berlin.de/koubek/forschung/KoubekWikiDidaktik.pdf

9. Kersten Reich: Konstruktivistische Didaktik

   2008, Beltz Verlag

10. Herbert P. Ginsburg, Sylvia Opper: Piagets Theorie der geistigen Entwicklung

    2004, Klett-Cotta

11. John R. Savery, Thomas M. Duffy: Problem Based Learning: An instructional model and its

constructivist framework

2001, Center for Research on Learning and Technology, Indiana University

12. Manfred Spitzer: Lernen – Gehirnforschung und die Schule des Lebens

2006, Spektrum Akademischer Verlag

13. Richard E. Mayer: Cognitive, metacognitive, and motivational aspects of problem solving

1998, Instructional Science, Kluwer Academic Publishers

14. Kersten Reich: Problem-Based Learning

http://methodenpool.uni-koeln.de/problembased/frameset_vorlage.html

15. Wikipedia: Problembasiertes Lernen

http://de.wikipedia.org/wiki/Problembasiertes_Lernen

16. Harry A. Shoemaker: The Functional Context Method of Instruction

1960, University of Oregon

17. David H. Jonassen: Toward a Design Theory of Problem Solving

2000, Educational Technology Research and Development Volume 48 Issue 4

18. Renée E. Weiss: Designing Problems to Promote Higher-Order Thinking

2003, New Directions For Teaching and Learning, no. 95

19. Ellis et al., Resources, Tools, and Techniques for Problem Based Learning in Computing

1999, Report of the ITiCSE'98 Working Group on Problem Based Learning

20. David H. Jonassen: All Problems are Not Equal: Implications for Problem-Based Learning

2008, Interdisciplinary Journal of Problem-based Learning Volume 2 Issue 2

21. Wilhelmiina Hämäläinen: Problem-based learning of theoretical computer science

Department of Computer Science, University of Joensuu, Finland

22. Michael J. O'Grady: Practical Problem-Based Learning in Computing Education

2012, University College Dublin

23. Wirkala, Kuhn: Problem-Based Learning in K-12 Education: Is it Effective and How Does it

Achieve its Effects?

2011, American Educational Research Journal

http://aer.sagepub.com/content/48/5/1157.abstract

24. Samuel B. Fee, Amanda M. Holland-Minkley: Teaching Computer Science through Problems, not Solutions

    2010, Information Technology Leadership, Washington & Jefferson College

25. John R. Savery: Overview of Problem-based Learning: Definitions and Distinctions

    2006, Interdisciplinary Journal of Problem-based Learning

    http://dx.doi.org/10.7771/1541-5015.1002

26. Jörg Zumbach, Agnes Weber & Gunter Olsowski: Problembasiertes Lernen - Konzepte, Werkzeuge und Fallbeispiele aus dem deutschsprachigen Raum

    2007, h.e.p. verlag ag, Bern

27. Jörg Zumbach & Peter Reimann, Computerunterstütztes fallbasiertes Lernen: Goal-Based Scenarios und Problem-Based Learning

    https://www.sbg.ac.at/mediaresearch/zumbach/download/1999_2006/book_chapters/zumbach_bookc_13.pdf

28. Jackie O'Kelly & J. Paul Gibson: RoboCode & Problem-Based Learning: A non-prescriptive approach to teaching programming

    2006, ITICSE '06 Proceedings of the 11th annual SIGCSE conference on innovation and technology in computer science education

29. Chris Beaumont, Dr Andrew Sackville & Chew Swee Cheng: Identifying Good Practice in the use of PBL to teach computing

    2003

30. Mark Newman: A pilot systematic review and meta-analysis on the effectiveness of Problem Based Learning

    2003, Newcastle University

    http://www.ltsn-01.ac.uk/resources/features/pbl

31. Esko Nuutila, Seppo Törmä & Lauri Malmi, Helsinki University of Technology, Finland: PBL and Computer Programming — The Seven Steps Method with Adaptations

    2005, Computer Science Education Vol. 15, No. 2

32. John M. Bunch: A Constructivist Approach to Teaching Web Development in Post-Secondary Vocational Settings

    2009, Journal of Information Technology Education

9 Anhang

9.1 *Tutorial*

9.1.1 Problem 1 Musterlösung

Gruppendokument

1. Wir wissen, wie wir:
   - ein Bild in HTML anzeigen und mit CSS positionieren können,
   - ein Bild mittels HTML so konfigurieren, dass die Bildquelle angezeigt wird, wenn sich der Mauszeiger darüber befindet,
   - auf ein Mausklickereignis reagieren können.
2. Um den Gorilla bei jedem Mausklick zu bewegen, können mir mittels dem onclick-Event jedes Mal die Position des Gorillas um 10 Pixel nach links verschieben.
3. Lernziel: mittels JavaScript die Position eines Bildes verändern.
4. Siehe individuelles Dokument.
5. Alle Gruppenmitglieder haben das Problem vollständig gelöst. Unsere Musterlösung sieht wie folgt aus (verschönert mit http://hilite.me):

Individuelles Dokument

Die Position eines Bildes ist als Zeichenkette mit der Endung „px“ gespeichert. Um sie zu verändern müssen wir die Zeichenkette vor dem „px“ in eine Ganzzahl umwandeln, um 10 reduzieren und dann eine neue Zeichenkette mit der Endung „px“ erstellen und diese als neue Position definieren.

Ich habe im Internet nach „Convert string to integer in Javascript“ gesucht und dabei folgende Seiten gefunden:

- http://stackoverflow.com/questions/1133770/how-do-i-convert-a-string-into-an-integer-in-javascript
- http://www.w3schools.com/jsref/jsref_parseint.asp

Die `parseInt`-Funktion hat es mir ermöglicht, das Lernziel folgendermaßen zu erreichen:

```
function move() {
      var gorilla = document.getElementById("gorilla");
      gorilla.style.left = parseInt(gorilla.style.left) - 10 + "px";
}
```

9.2 *Evaluation*

9.2.1 CLISS1

9.2.1.1 *Formative Evaluation*

Test 1 vom 19.10.12

Jeder Schüler hatte eine der drei folgenden, von Herrn Fisch erstellten, Aufgaben zu lösen.

Aufgabe 01: Idealgewicht

Laut Borca kann man das Idealgewicht einer Person anhand ihrer Größe mit folgender Formel berechnen:

Idealgewicht [kg] = Größe [cm] - 100

Lege auf dem Server FOXI das Verzeichnis **Aufgaben** an. Lege in diesem Verzeichnis ein Unterverzeichnis mit dem Namen **01** an. Erstelle ein neues HTML5 Dokument welches du in dem zuletzt erstellen Verzeichnis unter dem Namen **index.html** abspeicherst.

Dateiname: **/Aufgaben/01/index.html**

Programmiere nun die Seite so, dass sie den Benutzer nach seiner Größe (in cm) und seinem Gewicht (in kg) fragt und ihm dann in einem Satz sein Idealgewicht laut Borca's Formel und in einem zweiten die Differenz zu seinem Idealgewicht mitteilt.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Aufgabe 1a</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <script>
            var groesse = prompt("Bitte geben Sie Ihre Größe in cm ein");
```

```
            var gewicht = prompt("Bitte geben Sie Ihr Gewicht in kg ein");
            document.write("Ihr Idealgewicht ist " + (groesse - 100) + " kg.<br>");
            document.write("Sie sind " + (gewicht - (groesse - 100)) +
                           " kg von Ihrem Idealgewicht entfernt.");
        </script>
    </body>
</html>
```

Aufgabe 01: Body Mass Index

Um festzustellen ob eine Person übergewichtig ist kann man den Body Mass Index, kurz BMI, mit folgender Formel berechnen:

BMI = Gewicht [kg] / (Größe [m] * Größe [m])

Lege auf dem Server FOXI das Verzeichnis **Aufgaben** an. Lege in diesem Verzeichnis ein Unterverzeichnis mit dem Namen **01** an. Erstelle ein neues HTML5 Dokument welches du in dem zuletzt erstellen Verzeichnis unter dem Namen **index.html** abspeicherst.

Dateiname: **/Aufgaben/01/index.html**

Programmiere nun die Seite so, dass sie den Benutzer nach seinem Gewicht (in kg) und seiner Größe (in m) fragt und ihm dann in einem Satz seinen BMI laut der obigen Formel anzeigt.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Aufgabe 1b</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <script>
            var gewicht = prompt("Bitte geben Sie Ihr Gewicht in kg ein");
            var groesse = prompt("Bitte geben Sie Ihre Größe in m ein");
            document.write("Ihr BMI lautet " + gewicht / (groesse * groesse) + ".");
        </script>
    </body>
</html>
```

Aufgabe 01: Alter

Um das aktuelle Jahr in der Variable **thisYear** zu speichern kann man in JavaScript folgende Zeile Code schreiben:

```
var thisYear = (new Date()).getFullYear();
```

Lege auf dem Server FOXI das Verzeichnis **Aufgaben** an. Lege in diesem Verzeichnis ein Unter-

verzeichnis mit dem Namen **01** an. Erstelle ein neues HTML5 Dokument welches du in dem zuletzt erstellen Verzeichnis unter dem Namen **index.html** abspeicherst.

Dateiname: **/Aufgaben/01/index.html**

Programmiere nun die Seite so, dass sie den Benutzer nach seinem Geburtsjahr fragt und ihm dann in einem Satz mitteilt wie alt er ist. (Um ganz korrekt zu sein, müsste man auch den Monat und den Tag abfragen, für diese Aufgabe ist das aber nicht nötig.)

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Aufgabe 1c</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <script>
            var birthYear = prompt("Bitte geben Sie Ihr Geburtsjahr ein");
            var thisYear = (new Date()).getFullYear();
            document.write("Sie sind " + (thisYear - birthYear) + " Jahre alt.");
        </script>
    </body>
</html>
```

Test 2 vom 5.12.12

Entwickle die Webseite und beachte folgende Punkte:

1. Das Fußballfeld hat einen grünen Hintergrund, beginnt 50 Pixel unterhalb des oberen Randes und nimmt immer die gesamte Breite und die verbleibende Höhe (Gesamthöhe - 50) des Browserfensters ein. Um dies zu erreichen müssen der linke, obere, rechte und untere Abstand des Fußballfeldes entsprechend gestylt werden. Hierzu können die Attribute **offsetWidth** und **offsetHeight** des Fußballfeld-**div**-Elementes verwendet werden.
2. Über dem Fußballfeld befinden sich ein **label**, ein **input** sowie zwei **button**. Es ist immer nur einer der zwei Knöpfe sichtbar.
3. Nach dem Drücken des Startknopfes bewegt der Ball sich alle 20 ms um eine gewisse Pixelzahl horizontal und vertikal. Diese Pixelzahl wird als Zufallszahl zwischen 1 und dem Wert, den der Benutzer im Eingabefeld vorgegeben hat, berechnet. Beispiel: Der Benutzer hat als maximale Geschwindigkeit 100 Pixel / 20 ms vorgegeben. Alle 20 ms wird eine Zufallszahl zwischen 1 und 100 berechnet. Der Ball bewegt sich dann um diesen Wert horizontal und

vertikal. Falls der Ball bei dieser Bewegung einen Rand überschreiten würde, wird er an diesen Rand bewegt und die Richtung (horizontal oder vertikal) geändert.

4. Beim Laden der Webseite wird der Benutzer mit dem Text "Please enter the magic word:" aufgefordert das Passwort **`CLISS1`** einzugeben. Erst wenn das korrekte Passwort eingegeben wurde kann die Animation gestartet werden.
5. Die Webseite muss fehlerfrei als HTML5-compliant validiert werden.
6. Alle Variablen werden explizit mit **`var`** definiert.
7. Alle JavaScript-Instruktionen werden mit **`;`** beendet.
8. Dein fertiges Script wird mit dem JavaScript Utility validiert.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Integrationstest</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <label>Max speed (pixels / 20 ms):</label>
        <input id="maxSpeedInput" value="10">
        <button id="startButton" onclick="start()">Start</button>
        <button id="stopButton" onclick="stop()" style="display: none">Stop</button>
        <div id="footballField" style="background-color: green; left: 0px; top: 50px; right: 0px;
               bottom: 0px; position: absolute">
            <img id="football" src="football352x352.png" alt="football352x352.png" width="352"
                    height="352" style="position: absolute; left: 0px; top: 0px;">
        </div>
        <script>
            var football = document.getElementById("football"), directionX = 1, directionY = 1,
                 timerID, timeout = 20;
            var footballField = document.getElementById("footballField"), footballWidth = 352,
                 footballHeight = 352;

            function checkPassword() {
                var password = prompt("Please enter the magic word:");
                while (password !== "CLISS1") {
                    password = prompt("Please enter the magic word:");
                }
            }

            function timer() {
                var maxStep = document.getElementById("maxSpeedInput").value;
                var step = Math.floor(Math.random() * maxStep) + 1;
                var left = parseInt(football.style.left), top = parseInt(football.style.top);

                if (left + step * directionX < 0) {
                    football.style.left = "0px";
                    directionX = 1;
                }
                else if (left + step * directionX + footballWidth > footballField.offsetWidth) {
```

```
                football.style.left = footballField.offsetWidth - footballWidth + "px";
                directionX = -1;
            }
            else football.style.left = left + step * directionX + "px";
            if (top + step * directionY < 0) {
                football.style.top = "0px";
                directionY = 1;
            }
            else if (top + step * directionY + footballHeight > footballField.offsetHeight) {
                football.style.top = footballField.offsetHeight - footballHeight + "px";
                directionY = -1;
            }
            else football.style.top = top + step * directionY + "px";

            timerID = setTimeout("timer()", timeout);
        }

        function start() {
            document.getElementById("startButton").style.display = "none";
            document.getElementById("stopButton").style.display = "inline";
            timer();
        }

        function stop() {
            document.getElementById("startButton").style.display = "inline";
            document.getElementById("stopButton").style.display = "none";
            clearTimeout(timerID);
        }
        checkPassword();
    </script>
  </body>
</html>
```

Test 3 vom 19.12.12

Entwickle die Webseite und beachte folgende Punkte:

1. Kopiere den HTML- und CSS-Code aus der Lösung.
2. **DIESEN PUNKT SOLLTEST DU ALS LETZTEN IMPLEMENTIEREN.** Beim Laden der Webseite wird der Benutzer mit dem Text "Please enter the secret key:" aufgefordert, das Passwort **`CLISS1`** einzugeben. Dazu hat er 3 Versuche. Wenn nach 3 Versuchen das richtige Passwort noch immer nicht eingegeben wurde, werden alle Elemente des Dokuments mittels der Anweisung **`document.getElementsByTagName("body")[0].removeChild(document.getElementById("calculator"));`** gelöscht und es passiert nichts mehr. Wird das richtige Passwort innerhalb der 3 Versuche eingegeben, wird der Taschenrechner freigegeben.
3. **`MC`** steht für memory clear, **`MR`** für memory read und **`MS`** für memory store. Der Taschenrechner hat zwei unabhängige Zwischenspeicher, die anfangs den Wert 0 enthalten. Mit **`MS`** wird

der Wert im Eingabefeld darüber zwischengespeichert, mit **MR** wird der zwischengespeicherte Wert im Eingabefeld angezeigt. Mit **MC** wird der Zwischenspeicher auf 0 gesetzt.

4. Die binären Operatoren **+**, **-**, *****, **/** und **%** arbeiten mit den beiden linken Eingabefeldern und schreiben das Ergebnis in das rechte Eingabefeld. Die unären Operatoren/Funktionen **sqrt**, **exp**, **ln** und **!** arbeiten mit dem linken Eingabefeld und schreiben das Ergebnis auch dorthin.
5. Die Fakultätsfunktion überprüft ob der Wert im linken Eingabefeld größer ist als 1000. Falls dies der Fall ist tut sie nichts.
6. Die Webseite muss fehlerfrei als HTML5-compliant validiert werden.
7. Alle Variablen werden explizit mit **var** definiert.
8. Alle JavaScript-Instruktionen werden mit **;** beendet.
9. Dein fertiges Script wird mit dem JavaScript Utility validiert.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 Calculator</title>
        <meta charset="UTF-8">
        <style>
            body {
                background: -moz-linear-gradient(to bottom right, yellow, #772222);
                background-attachment: fixed;
            }
        </style>
    </head>
    <body>
        <section id="calculator">
            <input id="inputA" type="number">
            <input id="inputB" type="number">
            <input id="result" type="number">
            <br>
            <button id="b1">MC</button>
            <button id="b2">MR</button>
            <button id="b3">MS</button>
            <button id="b4">MC</button>
            <button id="b5">MR</button>
            <button id="b6">MS</button>
            <button id="b7">+</button>
            <button id="b8">-</button>
            <button id="b9">*</button>
            <button id="b10">/</button>
            <button id="b11">%</button>
            <br>
            <button id="b12">sqrt</button>
            <button id="b13">exp</button>
            <button id="b14">ln</button>
```

```
        <br>
        <button id="b15">!</button>
    </section>
    <script>
        var memoryA = 0, memoryB = 0;
        var inputA = document.getElementById("inputA");
        var inputB = document.getElementById("inputB");
        var result = document.getElementById("result");

        document.getElementById("b1").onclick = memoryAClear;
        document.getElementById("b2").onclick = memoryARead;
        document.getElementById("b3").onclick = memoryAStore;
        document.getElementById("b4").onclick = memoryBClear;
        document.getElementById("b5").onclick = memoryBRead;
        document.getElementById("b6").onclick = memoryBStore;
        document.getElementById("b7").onclick = add;
        document.getElementById("b8").onclick = subtract;
        document.getElementById("b9").onclick = multiply;
        document.getElementById("b10").onclick = divide;
        document.getElementById("b11").onclick = modulo;
        document.getElementById("b12").onclick = squareRoot;
        document.getElementById("b13").onclick = exponential;
        document.getElementById("b14").onclick = naturalLog;
        document.getElementById("b15").onclick = factorial;

        function memoryAClear() {
            memoryA = 0;
        }

        function memoryARead() {
            inputA.value = memoryA;
        }

        function memoryAStore() {
            memoryA = inputA.value;
        }

        function memoryBClear() {
            memoryB = 0;
        }

        function memoryBRead() {
            inputB.value = memoryB;
        }

        function memoryBStore() {
            memoryB = inputB.value;
        }

        function add() {
            result.value = Number(inputA.value) + Number(inputB.value);
        }

        function subtract() {
            result.value = inputA.value - inputB.value;
        }

        function multiply() {
            result.value = inputA.value * inputB.value;
```

```
            }

            function divide() {
                result.value = inputA.value / inputB.value;
            }

            function modulo() {
                result.value = inputA.value % inputB.value;
            }

            function squareRoot() {
                inputA.value = Math.sqrt(inputA.value);
            }

            function exponential() {
                inputA.value = Math.exp(inputA.value);
            }

            function naturalLog() {
                inputA.value = Math.log(inputA.value);
            }

            function factorial() {
                var fact = 1;
                if (inputA.value >= 1000) return;
                for (var i = 2; i <= inputA.value; i++) fact *= i;
                inputA.value = fact;
            }

            function loginCheck() {
                var password = "", counter = 0, limit = 3;
                while (password !== "CLISS1" && counter <= limit) {
                    password = prompt("Please enter the secret key:");
                    counter++;
                }
                if (password !== "CLISS1") {
                   document.getElementsByTagName("body")[0].removeChild(document.
                      getElementById("calculator"));
                }
            }

            onload = loginCheck();
        </script>
    </body>
</html>
```

Test 4 vom 16.1.13

Erstelle eine HTML Seite zum Umrechnen von Währungen. Der Benutzer wählt seine Quell- und Ziel-Währung aus und gibt den umzuwandelnden Betrag in einem Textfeld ein. Mit dem Knopf **`btnUmrechnen`** führt er die Umrechnung aus.

Erstelle 2 Felder **`arWaehrungName`** und **`arWaehrungKurs`**, welche das Währungskürzel respektive den Umwandlungswert enthalten. Verwende folgende Kurswerte:

1 USD	0,76 EUR
1 GBP	1,23 EUR
1 JPY	0,0087 EUR
1 CHF	0,83 EUR

Die Auswahlelemente für die Währung erhalten ihre Werte aus dem Feld **`arWaehrungName`**.

Wenn die Funktionalität vollständig programmiert ist, style die Seite mittels CSS exakt nach Vorgabe.

Für Fortgeschrittene: Wenn in einem Auswahlelement eine Währung ausgewählt wurde, darf diese nicht mehr in dem zweiten Auswahlelement angezeigt werden.

Musterlösung

```
<!DOCTYPE html>
<html lang="de">
    <head>
        <title>Währungsrechner</title>
        <meta charset="UTF-8">
        <style>
            section {
                background-color: #EEE;
                width: 255px;
                padding: 10px;
            }

            div {
                margin-top: 10px;
            }

            label {
                float: left;
                width: 100px;
                text-align: right;
                padding-right: 10px;
            }
        </style>
    </head>
    <body>
        <h1>Währungsrechner</h1>
        <section>
            <div>
                <label>Quellwährung:</label>
                <select id="quellWaehrungSelect" onchange="quelleChange()">
                    <option></option>
                    <option></option>
                    <option></option>
                    <option></option>
                    <option></option>
                </select>
            </div>
            <div>
                <label>Zielwährung:</label>
```

```
            <select id="zielWaehrungSelect" onchange="zielChange()">
                <option></option>
                <option></option>
                <option></option>
                <option></option>
                <option></option>
            </select>
        </div>
        <div>
            <label>Betrag:</label>
            <input id="betragInput">
        </div>
        <div>
            <button onclick="umwandeln()" style="margin-left: 110px;">Umwandeln</button>
        </div>
    </section>
    <script>
        var arWaehrungName = ["EUR", "USD", "GBP", "JPY", "CHF"];
        var arWaehrungKurs = [1, 0.76, 1.23, 0.0087, 0.83];
        var quellWaehrungSelect = document.getElementById("quellWaehrungSelect");
        var zielWaehrungSelect = document.getElementById("zielWaehrungSelect");
        var betragInput = document.getElementById("betragInput");
        for (var i = 0; i < arWaehrungName.length; i++) {
            quellWaehrungSelect.options[i].text = arWaehrungName[i];
            quellWaehrungSelect.options[i].value = arWaehrungName[i];
            if (i !== 0) {
                zielWaehrungSelect.options[i].text = arWaehrungName[i];
                zielWaehrungSelect.options[i].value = arWaehrungName[i];
            }
        }

        function umwandeln() {
            var quellWaehrungIndex = quellWaehrungSelect.selectedIndex;
            var zielWaehrungIndex = zielWaehrungSelect.selectedIndex;
            var quellWaehrungKurs = arWaehrungKurs[quellWaehrungIndex];
            var zielWaehrungKurs = arWaehrungKurs[zielWaehrungIndex];
            betragInput.value *= quellWaehrungKurs / zielWaehrungKurs;
        }

        function quelleChange() {
            var quellWaehrungIndex = quellWaehrungSelect.selectedIndex;
            // Regenerierung aller Zielwährungen.
            for (var i = 0; i < arWaehrungName.length; i++) {
                zielWaehrungSelect.options[i].text = arWaehrungName[i];
            }
            // Falls eine gültige Quellwährung ausgewählt wurde steht
            // diese nicht mehr als Zielwährung zur Verfügung.
            if (quellWaehrungSelect.options[quellWaehrungIndex].text !== "") {
                zielWaehrungSelect.options[quellWaehrungIndex].text="";
            }
        }

        function zielChange() {
            var zielWaehrungIndex = zielWaehrungSelect.selectedIndex;
            // Regenerierung aller Quellwährungen.
            for (var i = 0; i < arWaehrungName.length; i++) {
                quellWaehrungSelect.options[i].text = arWaehrungName[i];
            }
            // Falls eine gültige Zielwährung ausgewählt wurde steht
```

```
                // diese nicht mehr als Quellwährung zur Verfügung.
                if (zielWaehrungSelect.options[zielWaehrungIndex].text !== "") {
                    quellWaehrungSelect.options[zielWaehrungIndex].text="";
                }
            }
        </script>
    </body>
</html>
```

9.2.1.2 *Summative Evaluation*

Bewertungsraster für Kompetenz 1

Evaluation Kompetenz 1 **Klasse:** T1IF2 **23.01.2013**
Name:

Schwerpunkt	Kompetenz	nicht gelöst	zum Teil gelöst	vollständig gelöst	Bewertung
Integration von JavaScript in CSS-formatiertes HTML	1			1	
Anwenden von grundlegenden Sprachelementen	1			1	
Anwenden von elementaren Kontrollstrukturen	1			1	
Anwenden von vordefinierten Funktionen	1			1	
Erstellen von einfachen benutzerdefinierten Funktionen ohne Parameter	1			1	
Benutzen der vordefinierten JavaScript-Objekte	1	1			
Verschachteln von Kontrollstrukturen	1	1			
Benutzen und erstellen von eindimensionalen Feldern	1			1	
Reagieren auf Mausereignisse	1		1		
Zugreifen und ändern der DOM-Elemente	1		1		

Kompetenz		nicht gelöst	zum Teil gelöst	vollständig gelöst	%
Die grundlegenden Konzepte von JavaScript in der Entwicklung von dynamischen Webseiten einsetzen	1	2	2	6	**70%**
Komplexere Skripte erstellen und optimieren	2	0	0	0	**nicht geprüft**
Debuggertools bei der Fehleranalyse im Programmierprozess einsetzen	3	0	0	0	**nicht geprüft**
Informationen selbstständig in Referenzen nachschlagen und einsetzen	4	0	0	0	**nicht geprüft**

Bewertungsraster für die Kompetenzen 2 und 3

Evaluation Kompetenzen 2 und 3

Klasse: T1IF2 **30.01.2013**
Name:

Schwerpunkt	Kompetenz	nicht gelöst	zum Teil gelöst	vollständig gelöst	Bewertung
Optimieren von gegebenen Skripten	2			1	
Einsetzen von 2-dimensionalen Feldern	2			1	
Analysieren von fehlerhaften Skripten	3		1		

Kompetenz		nicht gelöst	zum Teil gelöst	vollständig gelöst	%
Die grundlegenden Konzepte von JavaScript in der Entwicklung von dynamischen Webseiten einsetzen	1	0	0	0	**nicht geprüft**
Komplexere Skripte erstellen und optimieren	2	0	0	2	**100%**
Debuggertools bei der Fehleranalyse im Programmierprozess einsetzen	3	0	1	0	**50%**
Informationen selbstständig in Referenzen nachschlagen und einsetzen	4	0	0	0	**nicht geprüft**

Kompetenz 1 – Aufgabe 1 23.1.13

Angabe

Erstelle folgende HTML Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst. Arbeite folgende Schritte **der Reihe nach** durch:

1. Das Raumschiff kann mit Hilfe der 4 Pfeile in die entsprechende Richtung bewegt werden, ohne jedoch das Universum zu verlassen. Würde die Bewegung über die Begrenzung hinaus führen, wird das Raumschiff an den entsprechenden Rand positioniert. Verwende die Attribute **`offsetWidth`** und **`offsetHeight`** des Universum-divs um die Grenzen zu bestimmen. Nimm für diesen Punkt an, dass die Anzahl Pixel um die das Raumschiff sich bewegt konstant ist, z.B. 100 Pixel.
2. Definiere ein leeres Feld (Array) mit dem Namen **`optionsArray`**. Fülle dieses Feld mit 50 zufällig berechneten Ganzzahlen aus dem Intervall [1, 200]. Bei der Berechnung dieser Zufallszahlen gilt Folgendes: falls die berechnete Zahl durch 5 teilbar ist, wird sie verdoppelt. Beispiel: Die berechnete Zahl ist 15, also wird der Wert 30 im Feld gespeichert. Jedes berechnete Feldelement wird zugleich auch in die Auswahlliste eingefügt. Die in der Auswahlliste ausgewählte Zahl wird als Schrittweite für die Bewegung des Raumschiffs verwendet. **Für diesen Punkt (also das Füllen der Auswahlliste) darfst du auf keinen Fall mehr als 30 Instruktionen verwenden!**

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 Space Ship</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="position: absolute; left: 0px; top: 0px; width: 100%; height: 100px;
               background-color: gold">
            <div style="position: absolute; left: 0px; top: 0px; width: 150px; height: 90px;">
                <img src="arrowLeft21x29.png" onclick="moveLeft();" alt="arrowLeft21x29.png"
                     style="position: absolute; top: 30px; left: 5px;">
                <img src="arrowUp29x21.png" onclick="moveUp();" alt="arrowUp29x21.png"
                     style="position: absolute; top: 5px; left: 30px;">
                <img src="arrowRight21x29.png" onclick="moveRight();" alt="arrowRight21x29.png"
                     style="position: absolute; top: 30px; left: 63px;">
                <img src="arrowDown29x21.png" onclick="moveDown();" alt="arrowDown29x21.png"
                     style="position: absolute; top: 63px; left: 30px">
            </div>
```

```
        <div style="position: absolute; left: 120px; top: 30px; height: 90px;">
            <label style="font-family: sans-serif;">Schrittweite (Pixel): </label>
            <select id="stepSelect" style="background-color: gold;">
                <option></option><option></option><option></option><option></option><option></option>
                <option></option> <option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option><option></option><option></option>
                <option></option><option></option><option></option>
            </select>
        </div>
        <div style="position: absolute; top: 0px; left: 400px; font-family: sans-serif;
           font-size: 400%">T1IF2 Space Ship</div>
    </div>
    <div id="space" style="position: absolute; left: 0px; top: 90px; right: 0px; bottom: 0px;
         background: -moz-linear-gradient(to bottom right, gold, black);
           background-attachment: fixed;">
        <img id="spaceShip" src="spaceship256x256.png" style="position: absolute; left: 0px;
           top: 0px;" alt="spaceship256x256.png">
    </div>
    <script>
        var space = document.getElementById("space");
        var spaceShip = document.getElementById("spaceShip"), spaceShipWidth = 256,
           spaceShipHeight = 256;
        var stepSelect = document.getElementById("stepSelect");
        var optionsArray = [];

        for (var i = 0; i < 50; i++) {
            optionsArray[i] = Math.floor(Math.random() * 200) + 1;
            if (optionsArray[i] % 5 === 0) optionsArray[i] = optionsArray[i] * 2;
            stepSelect.options[i].text = optionsArray[i];
        }

        function moveLeft() {
            var left = parseInt(spaceShip.style.left);
            var step = stepSelect.options[stepSelect.selectedIndex].text;

            if (left - step < 0) spaceShip.style.left = "0px";
            else spaceShip.style.left = left - step + "px";
        }

        function moveUp() {
            var top = parseInt(spaceShip.style.top);
            var step = optionsArray[stepSelect.selectedIndex];

            if (top - step < 0) spaceShip.style.top = "0px";
            else spaceShip.style.top = top - step + "px";
        }

        function moveRight() {
            var left = parseInt(spaceShip.style.left);
            var step = optionsArray[stepSelect.selectedIndex];
```

```
            if ((left + step + spaceShipWidth) > space.offsetWidth)
                spaceShip.style.left = space.offsetWidth - spaceShipWidth + "px";
            else spaceShip.style.left = left + step + "px";
        }

        function moveDown() {
            var top = parseInt(spaceShip.style.top);
            var step = optionsArray[stepSelect.selectedIndex];

            if (top + step + spaceShipHeight > space.offsetHeight)
                spaceShip.style.top = space.offsetHeight - spaceShipHeight + "px";
            else spaceShip.style.top = top + step + "px";
        }
    </script>
  </body>
</html>
```

Ergebnis

	Kompetenz 1	Integration von JavaScript in CSS-formatiertes HTML	Anwenden von grundlegenden Sprachelementen	Anwenden von elementaren Kontrollstrukturen	Anwenden von vordefinierten Funktionen	Erstellen von einfachen benutzerdefinierten Funktionen ohne Parameter	Benutzen der vordefinierten JavaScript-Objekte	Verschachteln von Kontrollstrukturen	Benutzen und erstellen von eindimensionalen Feldern	Reagieren auf Mausereignisse	Zugreifen und ändern der DOM-Elemente
Schüler 1	90%	1	1	0,5	1	1	1	1	1	1	0,5
Schüler 2	20%	0,5	0,5	0,5	0	0	0	0	0	0,5	0
Schüler 3	45%	0,5	0,5	0	1	1	0	0	0	1	0,5
Schüler 4	95%	1	1	1	1	1	1	1	1	1	0,5
Schüler 5	80%	1	0,5	1	1	1	0,5	1	0,5	1	0,5
Schüler 6	45%	0,5	1	1	0,5	1	0	0	0	0	0,5
Schüler 7	60%	0,5	0,5	1	0,5	1	1	0,5	0,5	0	0,5
Schüler 8	75%	1	1	0,5	1	1	0,5	0	1	1	0,5
Schüler 9	100%	1	1	1	1	1	1	1	1	1	1
Schüler 10	80%	1	0,5	1	1	1	1	0	1	1	0,5
Schüler 11	70%	0,5	1	0,5	1	1	1	0	0,5	1	0,5
Schüler 12	55%	0,5	1	1	1	1	0	0	0	0,5	0,5
Schüler 13	95%	1	1	1	1	1	1	1	1	1	0,5
Schüler 14	90%	1	1	1	1	1	0,5	1	1	1	0,5
Schüler 15	70%	1	1	0,5	1	1	0	0	1	1	0,5
Schüler 16	35%	0,5	0,5	0	1	0,5	0	0	0	0,5	0,5
Schüler 17	95%	1	1	1	1	1	1	1	1	1	0,5
	Durchschnitt	79%	82%	74%	88%	91%	56%	44%	62%	79%	50%
bestanden	11										
nicht bestanden	6										
nicht evaluiert	4										
Total	21										

Kompetenz 1 – Aufgabe 2 28.1.13

Angabe

Erstelle die im Video gezeigte Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst. Arbeite folgende Schritte **der Reihe nach** durch:

1. Aus der Auswahlliste kann zwischen drei Autos gewählt werden. Sobald ein anderes Auto gewählt wird ändert sich das angezeigte Bild.
2. Definiere ein leeres Feld (Array) mit dem Namen **`pointXArray`** sowie ein weiteres Feld

namens **`pointYArray`**, welches mit den Werten in Zeile 24 gefüllt wird.

3. Definiere eine Funktion namens **`fillXArray`**, welche Folgendes tut:
    1. Eine Variable namens **`xOffset`** wird definiert und mit einer zufällig berechneten Ganzzahl aus dem Intervall [1, 400] gefüllt.
    2. **`pointXArray`** wird wie folgt mit Werten gefüllt:
        1. Die Positionen 0-9 erhalten den Wert **`xOffset + pos * 40`**, wobei **`pos`** die Position im Array bezeichnet.
        2. Die Positionen 10-19 erhalten den Wert **`xOffset + 400 - (pos - 10) * 40`**.
4. Beim Klicken des Knopfes ändert sich dessen Beschriftung in "Stop" um und die Funktion **`fillXArray`** wird ausgeführt. Dann durchläuft das Auto die X- und Y-Positionen, die in den beiden Feldern gespeichert sind, mit einem Zeitabstand von jeweils 100 Millisekunden. Wenn das Ende der Felder erreicht wurde werden sie wieder von Anfang an durchlaufen.
5. Beim erneuten Klicken des Knopfes ändert sich dessen Beschriftung wieder in "Loop" um und das Auto stoppt.
6. Beim erneuten Klicken des Knopfes wird Punkt 4 erneut ausgeführt und so weiter.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 Space Circuit</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="position: absolute; left: 0px; top: 0px; width: 100%; height: 100%;
             background: -moz-radial-gradient(rgb(20, 50, 20), rgb(60, 255, 60), rgb(20, 50,20),
               black);
             background-attachment: fixed;">
            <div style="position: absolute; left: 0px; top: 0px; width: 100%; height: 20px;
                      opacity: 0.75;">
                <div style="position: absolute; left: 0px; top: 0px; height: 90px;">
                    <select id="carSelect" style="opacity: 0.75;" onchange="selected();">
                        <option value="camaro256x256.png">Camaro</option>
                        <option value="ferrari256x256.png">Ferrari</option>
                        <option value="cabrio256x256.png">Cabrio</option>
                    </select>
                    <button id="loopButton" onclick="startLoop();">Loop</button>
                </div>
                <div style="position: absolute; top: 0px; left: 400px; color: lightblue;
                      font-family: sans-serif;
                     font-size: 400%">T1IF2 Space Circuit</div>
            </div>
```

```
            <div style="position: absolute; left: 0px; top: 90px; right: 0px; bottom: 0px;">
                <img id="car" src="camaro256x256.png" style="position: absolute; left: 200px;
                      top: 200px;" alt="camaro256x256.png">
            </div>
        </div>
        <script>
            var car = document.getElementById("car"), carWidth = 256, carHeight = 256;
            var carSelect = document.getElementById("carSelect");
            var pointXArray = [];
            var pointYArray = [200, 160, 120,  80,  40,   0,  40,  80, 120, 160, 200, 240, 280, 320,
                               360, 400, 360, 320, 280, 240];
            var idx = 0, arrayPoints = 20;
            var timeout = 100, timerID;

            function fillXArray() {
                var xOffset = Math.floor(Math.random() * 400) + 1;
                for (var i = 0; i < arrayPoints / 2; i++) pointXArray[i] = xOffset + i * 40;
                for (var i = arrayPoints / 2; i < arrayPoints; i++) pointXArray[i] = xOffset + 400 -
                      (i - 10) * 40;
            }

            function selected() {
                car.src = carSelect.options[carSelect.selectedIndex].value;
                car.alt = carSelect.options[carSelect.selectedIndex].value;
            }

            function loop() {
                if (idx >= arrayPoints) idx = 0;
                car.style.left = pointXArray[idx] + "px";
                car.style.top = pointYArray[idx++] + "px";
            }

            function startLoop() {
                fillXArray();
                timerID = setInterval(loop, timeout);
                loopButton.innerHTML = "Stop";
                loopButton.onclick = stopLoop;
            }

            function stopLoop() {
                clearInterval(timerID);
                loopButton.innerHTML = "Loop";
                loopButton.onclick = startLoop;
            }

            selected();
        </script>
    </body>
</html>
```

Ergebnis

	Kompetenz 1	Integration von JavaScript in CSS-formatiertes HTML	Anwenden von grundlegenden Sprachelementen	Anwenden von elementaren Kontrollstrukturen	Anwenden von vordefinierten Funktionen	Erstellen von einfachen benutzerdefinierten Funktionen ohne Parameter	Benutzen der vordefinierten JavaScript-Objekte	Verschachteln von Kontrollstrukturen	Benutzen und erstellen von eindimensionalen Feldern	Reagieren auf Mausereignisse	Zugreifen und ändern der DOM-Elemente
Schüler 1	78%	1	0,5	0,5	1	1	1	nicht evaluiert	0,5	1	0,5
Schüler 2	56%	0,5	0,5	0,5	0,5	1	0,5		0,5	0,5	0,5
Schüler 3	73%	0,5	1	0,5	0,5	1	1		1	0,5	0,5
Schüler 4	78%	1	1	0,5	1	1	1		0,5	0,5	0,5
Schüler 5	78%	0,5	0,5	1	1	1	1		0,5	1	0,5
Schüler 6	62%	1	1	0,5	1	0,5	0		0,5	0,5	0,5
	Durchschnitt	75%	75%	58%	83%	92%	75%		58%	67%	50%
bestanden	4										
nicht bestanden	2										
nicht evaluiert	0										
Total	6										

Kompetenz 1 – Aufgabe 3 29.1.13

Angabe

Erstelle die im Video gezeigte Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst. Arbeite folgende Schritte **der Reihe nach** durch:

1. Definiere ein leeres Feld (Array) mit dem Namen **`yPosArray`** das 100 Ganzzahlen aus dem Intervall [0, 5 * pos] enthält, wobei **`pos`** der Position im Array entspricht.
2. Beim Klicken des Knopfes ändert sich dessen Beschriftung in "Stop" um und die Zielscheibe fängt an, die im Array gespeicherten Positionen zu durchlaufen, wobei alle 20 Millisekunden die vertikale Position geändert wird. Wenn das Ende des Feldes erreicht wurde wird das Feld von hinten nach vorne durchlaufen, dann wieder von vorne nach hinten, etc.
3. Beim erneuten Klicken des Knopfes ändert sich dessen Beschriftung in "Start" um und die Zielscheibe stoppt.
4. Beim erneuten Klicken des Knopfes bewegt sich die Zielscheibe wieder und so weiter.
5. Der Panzer kann mittels der Pfeiltasten nach unten oder nach oben bewegt werden. Er bewegt sich dabei jeweils um die in der Auswahlliste ausgewählte Pixelzahl (10, 25 oder 50), außer wenn er dabei das Feld nach unten oder oben verlassen würde. In dem Fall wird er an den (unteren oder oberen) Rand gesetzt.
6. Für Fortgeschrittene: beim Drücken der Taste F feuert der Panzer. Wenn sich die Kanone dabei gerade auf der Höhe der Zielscheibe befindet erscheint ein Meldungsfenster mit dem Text "Hit!". Nimm an, dass die vertikale Position der Kanone der vertikalen Position des Panzers entspricht.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 CLISS1 Targeting Practice</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <button id="button" onclick="start()">Start</button>
        <select id="select" onchange="selectChange()">
            <option>10</option>
            <option>25</option>
            <option>50</option>
        </select>
        <div style="background: -moz-linear-gradient(to bottom right, white, green);
             position: absolute; top: 35px; left: 0px; right: 0px; bottom: 0px; width: 800px;
               height: 630px; background-attachment: fixed;">
            <img id="tank" src="tank128x66.png" alt="tank128x66.png" width="128" height="66"
                      style="position: absolute; left: 650px; top: 0px;">
            <img id="target" src="target91x91.png" alt="target91x91.png" width="91" height="91"
                      style="position: absolute; left: 0px; top: 0px;">
        </div>
        <script>
            var timerID, timeout = 20, battleFieldHeight = 630;
            var yPosArray = [], arraySize = 100, currIdx = 0, currDirection = 1;
            var target = document.getElementById("target"), targetHeight = 91;
            var tank = document.getElementById("tank"), tankHeight = 66;
            var select = document.getElementById("select");
            var tankStep = parseInt(select.options[select.selectedIndex].text);

            for (var i = 0; i < arraySize; i++) yPosArray[i] = Math.floor(Math.random() * (5 * i +
                1));

            function selectChange() {
                tankStep = parseInt(select.options[select.selectedIndex].text);
                select.blur();
            }

            function moveTarget() {
                target.style.top = yPosArray[currIdx] + "px";
                if ((currIdx >= arraySize - 1) || (currIdx <= 0 && currDirection === -1))
                       currDirection = -currDirection;
                currIdx += currDirection;
            }

            function start() {
                timerID = setInterval("moveTarget()", timeout);
                button.innerHTML = "Stop";
                button.onclick = stop;
            }

            function stop() {
                clearInterval(timerID);
                button.innerHTML = "Start";
                button.onclick = start;
            }

            function keyHandler(event) {
```

```
                var top = parseInt(tank.style.top);
                if (event.keyCode === 38) {
                    if (top - tankStep < 0) tank.style.top = "0px";
                    else tank.style.top = top - tankStep + "px";
                }
                if (event.keyCode === 40) {
                    if (top + tankStep + tankHeight > battleFieldHeight) tank.style.top =
                      battleFieldHeight - tankHeight + "px";
                    else tank.style.top = top + tankStep + "px";
                }
                if (event.keyCode === 70)
                    if ((parseInt(target.style.top) <= top) && (parseInt(target.style.top) +
                      targetHeight >= top)) alert("Hit!");
            }

            onkeydown = keyHandler;
        </script>
    </body>
</html>
```

Kompetenz 1 – Aufgabe 4 31.1.13

Angabe

Erstelle folgende HTML Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst mit diesem Hintergrundbild. Zuerst wird der Benutzer nach seinem Namen gefragt und anschließend mit seinem Namen und dem aktuellen Datum, in der korrekten lokalen Schreibweise, begrüßt. Beim ersten Druck auf die Taste **`S`** wird die Speicherung der aktuellen Mausposition in den Feldern **`mouseXArray`** und **`mouseYArray`** gestartet. Hierzu verwendest du die vorgegebene Funktion **`getMouseXY`**. Diese Funktion zeigt gleichzeitig die aktuellen Mauskoordinaten in den beiden Eingabefeldern an. Beim erneuten Druck auf die Taste **`S`** wird die Speicherung und Anzeige der Mauskoordinaten gestoppt. Beim Druck auf den Knopf durchläuft das Auto die gespeicherten Mauspositionen mit einem Zeitabstand von 10 Millisekunden. Bei einem erneuten Druck auf den Knopf stoppt das Auto, bei erneutem Klick fährt es weiter etc. Achtung: das Auto wird immer so positioniert, dass die gespeicherte Mausposition dem Mittelpunkt des Autos entspricht!

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 CLISS1 Targeting Practice</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <button id="button" onclick="start()">Start</button>
```

```
<select id="select" onchange="selectChange()">
    <option>10</option>
    <option>25</option>
    <option>50</option>
</select>
<div style="background: -moz-linear-gradient(to bottom right, white, green);
     position: absolute; top: 35px; left: 0px; right: 0px; bottom: 0px; width: 800px;
       height: 630px; background-attachment: fixed;">
    <img id="tank" src="tank128x66.png" alt="tank128x66.png" width="128" height="66"
       style="position: absolute; left: 650px; top: 0px;">
    <img id="target" src="target91x91.png" alt="target91x91.png" width="91" height="91"
       style="position: absolute; left: 0px; top: 0px;">
</div>
<script>
    var timerID, timeout = 20, battleFieldHeight = 630;
    var yPosArray = [], arraySize = 100, currIdx = 0, currDirection = 1;
    var target = document.getElementById("target"), targetHeight = 91;
    var tank = document.getElementById("tank"), tankHeight = 66;
    var select = document.getElementById("select");
    var tankStep = parseInt(select.options[select.selectedIndex].text);

    for (var i = 0; i < arraySize; i++) yPosArray[i] = Math.floor(Math.random() * (5 * i +
       1));

    function selectChange() {
        tankStep = parseInt(select.options[select.selectedIndex].text);
        select.blur();
    }

    function moveTarget() {
        target.style.top = yPosArray[currIdx] + "px";
        if ((currIdx >= arraySize - 1) || (currIdx <= 0 && currDirection === -1))
              currDirection = -currDirection;
        currIdx += currDirection;
    }

    function start() {
        timerID = setInterval("moveTarget()", timeout);
        button.innerHTML = "Stop";
        button.onclick = stop;
    }

    function stop() {
        clearInterval(timerID);
        button.innerHTML = "Start";
        button.onclick = start;
    }

    function keyHandler(event) {
        var top = parseInt(tank.style.top);
        if (event.keyCode === 38) {
            if (top - tankStep < 0) tank.style.top = "0px";
            else tank.style.top = top - tankStep + "px";
        }
        if (event.keyCode === 40) {
            if (top + tankStep + tankHeight > battleFieldHeight) tank.style.top =
                      battleFieldHeight - tankHeight + "px";
            else tank.style.top = top + tankStep + "px";
        }
```

```
                if (event.keyCode === 70)
                    if ((parseInt(target.style.top) <= top) && (parseInt(target.style.top) +
                      targetHeight >= top)) alert("Hit!");
            }

            onkeydown = keyHandler;
        </script>
    </body>
</html>
```

Kompetenzen 2 und 3 – Aufgabe 1 30.1.13

Angabe

Erstelle die im Video gezeigte Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst. Arbeite folgende Schritte **der Reihe nach** durch:

Teil 1

1. Definiere ein 2-dimensionales Feld (Array) mit dem Namen **`pixelColorArray`**.
2. Schreibe eine Funktion **`fillColorArray()`**, welche das Feld mit dem RGB-Wert (als String in der Form **`"rgb(Rotanteil, Grünanteil, Blauanteil)"`**) für jedes Pixel eines 100x100 Pixel großen Quadrats füllt. Der Wert für jedes Pixel berechnet sich wie folgt: Rot- und Blauanteil: **`i + j`**, Grünanteil: **`255 - (i + j)`**, wobei **`i`** die horizontale und **`j`** die vertikale Position des Pixels darstellen.
3. Schreibe eine Funktion **`drawColorArray()`**, welche jedes Pixel des Feldes mit der gespeicherten Farbe malt. Hierzu wird die zur Verfügung gestellte Funktion **`draw(x, y, color)`** verwendet.
4. Führe die beiden Funktionen aus und vergewissere dich, dass das Ergebnis so aussieht wie im Video.

Teil 2

1. Erstelle eine Kopie der gegebenen Funktion **`findFirstPos(arr, x)`** mit dem Namen **`fastFindFirstPos(arr, x)`** und optimiere sie so, dass sie möglichst wenig Variablen, Instruktionen und Iterationen benötigt. Zum Überprüfen des Erfolgs sind bereits Beispielinstruktionen vorhanden, die du verwenden und ausbauen kannst.

Teil 3

1. Im Kommentar **`K3 Debugging`** befinden sich 7 fehlerhafte Zeilen JavaScript. Erstelle eine Kopie davon und korrigiere sie so, dass die Uhr wie im Video korrekt und ohne Fehlermeldungen in der Console funktioniert. Alle Variablen müssen lokal deklariert werden.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 Evaluation Kompetenzen 2 und 3 Aufgabe 1</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="position: absolute; left: 0px; top: 0px; width: 100%; height: 100%;
             background: -moz-radial-gradient(rgb(20, 50, 20), rgb(60, 255, 60), rgb(20, 50,20),
               black);
             background-attachment: fixed;">
            <canvas id="canvas" width="150" height="150">Your Browser does not support canvas!
            </canvas>
            <div id="debugMeDiv" style="color: white; font-size: 500%"></div>
        </div>
        <script>
            // K2 komplexe Verschachtelungen und 2-dimensionale Felder
            var canvas = document.getElementById("canvas");
            var context = canvas.getContext("2d");
            var pixelColorArray = [], pixelColorArraySize = 100;
            var numIterations = 0;

            function fillColorArray() {
                for (var i = 0; i < pixelColorArraySize; i++) {
                    pixelColorArray[i] = [];
                    for (var j = 0; j < pixelColorArraySize; j++) {
                        pixelColorArray[i][j] = "rgb(" + (i + j) + "," + (255 - (i + j)) + "," + (i
                              + j) + ")";
                    }
                }
            }

            function draw(x, y, color) {
                context.fillStyle = color;
                context.fillRect(x, y, 1, 1);
            }

            function drawColorArray() {
                for (var i = 0; i < pixelColorArraySize; i++)
                    for (var j = 0; j < pixelColorArraySize; j++) draw(i, j, pixelColorArray[i][j]);
            }

            fillColorArray();
            drawColorArray();

            // K2 Optimierung von Skripten
            function findFirstPos(arr, x) { // arr is an array of numbers
                var pos = -1;
```

```
    numIterations = 0;
    if (typeof arr === 'undefined') return -1;
    if (typeof arr !== 'undefined') {
        for (var i = 0; i < arr.length; i++) {
            if (arr[i] === x && pos === -1) pos = i;
            numIterations++;
        }
    }
    return pos;
}
// Musterlösung
function fastFindFirstPos(arr, x) { // arr is an array of numbers
    numIterations = 0;
    if (typeof arr !== 'undefined')
        for (var i = 0; i < arr.length; i++) {
            numIterations++;
            if (arr[i] === x) return i
        }
    return -1;
}
var arr = [1, 2, 3, 3, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 3];
console.log("Number of iterations before: ", numIterations);
console.log("findFirstPos: ", findFirstPos(arr, 3));
console.log("Number of iterations after: ", numIterations);
numIterations = 0;
console.log("Number of iterations before: ", numIterations);
console.log("fastFindFirstPos: ", fastFindFirstPos(arr, 3));
console.log("Number of iterations after: ", numIterations);

// K3 Debugging
/*
 funtion debugMe1 i if i < 10 i = 0 + i retourn i
 funtion debugMe2 today == Date() h === today.getHour()
 m === today.getMinuttes() s === today.getSeconds
 m === debugMe1 m s === debugMe1 s
 getELementById("debugMeDiv").HTML == h + : + s;
 funtion debugMe3 setInterval debugMe2 1000
 debug
 */
// Musterlösung
function debugMe1(i)
{
    if (i < 10) i = "0" + i;
    return i;
}

function debugMe2() {
    var today = new Date();
    var h = today.getHours();
    var m = today.getMinutes();
    var s = today.getSeconds();
    m = debugMe1(m);
    s = debugMe1(s);
    document.getElementById("debugMeDiv").innerHTML = h + ":" + m + ":" + s;
}

function debugMe3() {
    setInterval("debugMe2()", 1000);
}
```

```
            debugMe3();
        </script>
    </body>
</html>
```

Kompetenzen 2 und 3 – Aufgabe 2 1.2.13

Angabe

Erstelle die im Video gezeigte Seite exakt nach Vorgabe. Verwende dazu folgendes, unvollständige Gerüst. Arbeite folgende Schritte **der Reihe nach** durch:

Teil 1

1. Definiere ein 2-dimensionales Feld (Array) mit dem Namen **`pixelColorArray`**.
2. Schreibe eine Funktion **`fillColorArray()`**, welche das Feld mit dem RGB-Wert (als String in der Form **`"rgb(Rotanteil, Grünanteil, Blauanteil)"`**) für jedes Pixel eines 100x100 Pixel großen Quadrats füllt. Der Wert für jedes Pixel berechnet sich wie folgt: Rot- und Blauanteil: **`i + j`**, Grünanteil: **`255 - (i + j)`**, wobei **`i`** die horizontale und **`j`** die vertikale Position des Pixels darstellen. Wenn **`i`** jedoch durch 10 teilbar ist, dann ist die Pixelfarbe schwarz.
3. Schreibe eine Funktion **`drawColorArray()`**, welche jedes Pixel des Feldes mit der gespeicherten Farbe malt. Hierzu wird die zur Verfügung gestellte Funktion **`draw(x, y, color)`** verwendet.
4. Führe die beiden Funktionen aus und vergewissere dich, dass das Ergebnis so aussieht wie im Video.

Teil 2

1. Erstelle eine Kopie der gegebenen Funktion **`sumFromTo(arr, startPos, endPos)`** mit dem Namen **`sumFromToOptimal(arr, startPos, endPos)`** und optimiere sie so, dass sie möglichst wenig Variablen, Instruktionen und Iterationen benötigt. Alle Variablen müssen lokal deklariert sein. Zum Überprüfen des Erfolgs sind bereits Beispielinstruktionen vorhanden die du verwenden und ausbauen kannst.

Teil 3

1. Im Kommentar **`K3 Debugging`** befinden sich 4 fehlerhafte Zeilen JavaScript. Erstelle eine Kopie davon und korrigiere sie so, dass die Anzeige wie im Video korrekt und ohne Fehlermeldungen in der Console funktioniert. Alle Variablen müssen lokal deklariert werden.

Musterlösung

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF2 Space Circuit</title>
        <meta charset="UTF-8">
    </head>
    <body>
        <div style="position: absolute; left: 0px; top: 0px; width: 100%; height: 100%;
             background: -moz-radial-gradient(rgb(200, 50, 20), rgb(255, 255, 60), rgb(255, 50,20),
               black);
             background-attachment: fixed;">
            <canvas id="canvas" width="150" height="150">Your Browser does not support canvas!
            </canvas>
            <div id="debugMeDiv" style="text-align: center; color: black; font-size: 500%"></div>
        </div>
        <script>
            // K2 komplexe Verschachtelungen und 2-dimensionale Felder
            var canvas = document.getElementById("canvas");
            var context = canvas.getContext("2d");
            var pixelColorArray = [], pixelColorArraySize = 100;
            var numIterations = 0;

            function fillColorArray() {
                for (var i = 0; i < pixelColorArraySize; i++) {
                    pixelColorArray[i] = [];
                    for (var j = 0; j < pixelColorArraySize; j++) {
                        pixelColorArray[i][j] = "rgb(" + (i + j) + "," + (255 - (i + j)) + "," + (i
                              + j) + ")";
                        if (i % 10 === 0)
                            pixelColorArray[i][j] = "black";
                    }
                }
            }

            function draw(x, y, color) {
                context.fillStyle = color;
                context.fillRect(x, y, 1, 1);
            }

            function drawColorArray() {
                for (var i = 0; i < pixelColorArraySize; i++)
                    for (var j = 0; j < pixelColorArraySize; j++)
                        draw(i, j, pixelColorArray[i][j]);
            }

            fillColorArray();
            drawColorArray();
```

```
// K2 Optimierung von Skripten
function sumFromTo(arr, startPos, endPos) {
    var sum = 0;
    numIterations = 0;

    if ((typeof arr === 'undefined') && startPos < 0)
        return -1;
    if ((typeof arr === 'undefined') && endPos > arr.length)
        return -1;
    if ((typeof arr !== 'undefined') && startPos < 0)
        return -1;
    if ((typeof arr !== 'undefined') && endPos > arr.length)
        return -1;
    for (var i = 0; i < arr.length; i++) {
        if (i >= startPos && i <= endPos)
            sum += arr[i];
        numIterations++;
    }
    return sum;
}

function sumFromToOptimal(arr, startPos, endPos) {
    var sum = 0;
    numIterations = 0;
    if ((typeof arr === 'undefined') || startPos < 0 || endPos > arr.length)
        return -1;
    for (var i = startPos; i <= endPos; i++) {
        sum += arr[i];
        numIterations++;
    }
    return sum;
}

var arr = [1, 2, 3, 3, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 3];
console.log("Number of iterations before: ", numIterations);
console.log("sumFromTo: ", sumFromTo(arr, 3, 7));
console.log("Number of iterations after: ", numIterations);
numIterations = 0;
console.log("Number of iterations before: ", numIterations);
console.log("sumFromToOptimal: ", sumFromToOptimal(arr, 3, 7));
console.log("Number of iterations after: ", numIterations);

// K3 Debugging
/*
 funtin debugMe1 n x = 0for i===0 i < n i++ if i % 5 !== 0 x += i retourn x
 funton debugMe2 var today === Date() getELementById("debugMeDiv").HTML ==
     today.toLocaleString);
 function debugMe3() setInterval debugMe2 1000
 debug
 */
// Musterlösung
function debugMe1(n)
{
    var x = 0;
    for (var i = 0; i < n; i++)
        if (i % 5 !== 0)
            x += i;
    return x;
}
```

```
            function debugMe2() {
                var today = new Date();
                document.getElementById("debugMeDiv").innerHTML = today.toLocaleString() + " " +
                      debugMe1(Math.floor(Math.random() * 100));
            }

            function debugMe3() {
                setInterval("debugMe2()", 1000);
            }
            debugMe3();
        </script>
    </body>
</html>
```

9.2.2 CLISS2

9.2.2.1 *Formative Evaluation*

T1IF Invaders

index.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>T1IF Invaders v1.0</title>
        <meta charset="UTF-8">
        <style>
            body {background-color: black; color: gold;}
            #splash {display:block; width: 200px; margin: auto;}
            #splash > ul {list-style-type: none;}
            #highScoreSection, #keysSection, #board {display: none;}
            table {border-spacing: 0;}
            table caption {font-size: 2em; font-weight: bold;}
            table thead {text-align: left;}
            table thead th {background-color: #CC2222;}
            th, td {padding: 5px;}
            tr:nth-of-type(odd) {background-color: #222222;}
            tr:nth-of-type(even) {background-color: #444444;}
        </style>
    </head>
    <body>
        <section id="splash">
            <h2>T1IF Invaders v1.0</h2>
            <ul>
                <li><button id="newGame" onclick="newGame();">New Game</button></li>
                <li><button id="highScores" onclick="displayHighScores();">High Scores</button></li>
                <li><button id="keys" onclick="displayKeys();">Keys</button></li>
            </ul>
        </section>
        <section id="highScoreSection">
            <table id="highScoreListTable">
```

```
                <caption>Hall of Fame</caption>
                <thead>
                    <tr>
                        <th>Rank</th>
                        <th>Player</th>
                        <th>Score</th>
                        <th>Level</th>
                    </tr>
                </thead>
                <tr>
                    <td>1</td>
                    <td>Gilles Everling</td>
                    <td>2300</td>
                    <td>56</td>
                </tr>
                <tr>
                    <td>2</td>
                    <td>Gilles Everling</td>
                    <td>2290</td>
                    <td>56</td>
                </tr>
                <tr>
                    <td>3</td>
                    <td>Gilles Everling</td>
                    <td>2280</td>
                    <td>56</td>
                </tr>
            </table>
            <button onclick="hideHighScores();">OK</button>
        </section>
        <section id="keysSection">
            <p>Use left and right cursor keys to move your spaceship and the space bar to fire.
                The game can be paused at any time by pressing <code>P</code>. Pressing
                <code>P</code> again will resume.
                You can exit the game early using <code>Esc</code>.</p>
            <button onclick="hideKeys();">OK</button>
        </section>
        <section id="board">
            <p>
                Score: <span id="score"></span> 
                Level: <span id="level"></span>
            </p>
            <canvas id="canvas" width="600" height="400">This browser does not run this game (canvas
               support missing).</canvas>
            <section>
                <img id="life1" src="playerspaceship36x46.png" alt="playerspaceship36x46.png">
                <img id="life2" src="playerspaceship36x46.png" alt="playerspaceship36x46.png">
            </section>
        </section>
        <script src="index.js"></script>
    </body>
</html>
```

index.js

```
// Declaration of global variables
var canvas = document.getElementById('canvas');
var context = canvas.getContext('2d');
```

```
var imagePlayer = new Image(36, 46);
var imageShot = new Image(12, 23);
var imageAlien = new Image(42, 27);
var imageAlienBomb = new Image(10, 10);
imagePlayer.src = 'playerspaceship36x46.png';
imageShot.src = 'shot12x23.png';
imageAlien.src = 'alien42x27.png';
imageAlienBomb.src = 'alienbomb10x10.png';
var currPlayerX = (canvas.width - imagePlayer.width) / 2;
var currPlayerY = canvas.height - imagePlayer.height - 5;
var playerShotsX = [], playerShotsY = [], numAliens = 40, numAliensPerRow = 10, minFireThreshold =
 300, initialPlayerSpeed = 20, DBServerURL = location.protocol + "//" + location.host +
 "/COURS/cliss/T1IFInvaders/index.php?callback=", initialNumOfLives = 2;
var aliensX = [], aliensY = [], alienBombsX = [], alienBombsY = [], highScoresReady;
var alienDirection, fireButton = false, moveLeft = false, moveRight = false, timeOfLastShot = 0;
var alienXSpeed, alienYSpeed, loseLife, numOfLives, alienBombSpeed;
var score = ~0, highScoreList, currLevel = 1, currPlayerSpeed, lastName = "", numShots, pauseButton,
        lastAnimationTime = 0, adjustmentFactor = 0.2;

// from http://www.sitepoint.com/simple-animations-using-requestanimationframe
var requestAnimationFrame = function(win, t) {
    return win["webkitR" + t] || win["r" + t] || win["mozR" + t]
            || win["msR" + t] || function(fn) {
        setTimeout(fn, 1000 / 60)
    }
}(window, "equestAnimationFrame");

/* Called by newGame.
 * Draws aliens and adjusts global variables. */
function init() {
    var x = 1, y = 1;
    for (var i = 0; i < numAliens; i++) {
        aliensX[i] = x;
        aliensY[i] = y;
        context.drawImage(imageAlien, aliensX[i], aliensY[i]);
        x += imageAlien.width + 10;
        if (i % numAliensPerRow == 9) { // 10 aliens per row
            x = 1;
            y += imageAlien.height + 7;
        }
    }
    document.getElementById("score").innerHTML = ~score;
    document.getElementById("level").innerHTML = currLevel;
    alienXSpeed = (5 + currLevel) * adjustmentFactor;
    alienYSpeed = currLevel;
    alienBombSpeed = Math.max(1, Math.floor(alienYSpeed / 3));
    alienDirection = 1; // aliens start moving to the right
    moveLeft = false; // no keys pressed
    moveRight = false;
    playerShotsX = []; // delete shots
    playerShotsY = [];
    alienBombsX = []; // delete bombs
    alienBombsY = [];
}

// Called by gameLoop.
function moveAliens() {
    var leftBorderTouched = false;
    for (var i = 0; i < aliensX.length; i++) {
```

```
        // Determine whether there is enough room to continue in the current direction.
        // If there isn't enough room, we need to change direction.
        if (aliensX[i] <= alienXSpeed && alienDirection === -1 || aliensX[i] + imageAlien.width >
               canvas.width && alienDirection === 1) {
            alienDirection = -alienDirection;
            // If we touched the left border, we need to move the aliens down.
            if (alienDirection === 1) leftBorderTouched = true;
            break; // Once we have changed direction, no need to continue checking the other aliens.
        }
    }
    if (leftBorderTouched) {
        alienYSpeed += 2 * adjustmentFactor;
        for (i = 0; i < aliensY.length; i++) aliensY[i] += alienYSpeed;
    }
    // Move aliens horizontally.
    for (i = 0; i < aliensX.length; i++) {
        aliensX[i] += alienDirection * alienXSpeed;
        context.drawImage(imageAlien, aliensX[i], aliensY[i]);
    }
}

// Called by gameLoop
function moveShotsAndBombs() {
    for (var i = 0; i < playerShotsY.length; i++) {
        playerShotsY[i] -= imageShot.height * adjustmentFactor;
        if (playerShotsY[i] < 0) { // delete bullet as it has left space
            playerShotsX.splice(i, 1);
            playerShotsY.splice(i, 1);
            i--;
        }
        else context.drawImage(imageShot, playerShotsX[i], playerShotsY[i]);
    }
    for (i = 0; i < alienBombsY.length; i++) {
        alienBombsY[i] += alienBombSpeed;
        if (alienBombsY[i] > canvas.height) { // delete bomb as it has left space
            alienBombsX.splice(i, 1);
            alienBombsY.splice(i, 1);
            i--;
        }
        else context.drawImage(imageAlienBomb, alienBombsX[i], alienBombsY[i]);
    }
}

// Called by gameLoop.
function checkCollisions() {
    // check whether a bullet has hit an alien or an alien or alien bomb touches the player
    var shotLeft, shotRight, shotTop, shotBottom, alienLeft, alienRight, alienTop, alienBottom,
        playerRight, playerBottom, alienIndex = 0, alienBombLeft, alienBombRight, alienBombTop,
        alienBombBottom, alienBombIndex = 0;

    // first check whether player has been touched by an alien or a bullet has touched an alien
    playerRight = currPlayerX + imagePlayer.width;
    playerBottom = currPlayerY + imagePlayer.height;

    while (alienIndex < aliensX.length) {
        alienLeft = aliensX[alienIndex];
        alienRight = alienLeft + imageAlien.width;
        alienTop = aliensY[alienIndex];
        alienBottom = alienTop + imageAlien.height;
```

```
        // if an alien manages to leave via the bottom of the frame, we lose a life
        if (alienRight >= currPlayerX && playerRight >= alienLeft && alienBottom >= currPlayerY ||
               alienBottom > canvas.height) {
            loseLife = true;
            return;
        }
        shotIndex = 0;
        while (shotIndex < playerShotsY.length) {
           // for each bullet check whether it touches the alien
            shotLeft = playerShotsX[shotIndex];
            shotRight = shotLeft + imageShot.width;
            shotTop = playerShotsY[shotIndex];
            shotBottom = shotTop + imageShot.height;
            if (alienRight >= shotLeft && shotRight >= alienLeft && alienBottom >= shotTop &&
               shotBottom >= alienTop) {
                playerShotsX.splice(shotIndex, 1);
                playerShotsY.splice(shotIndex, 1);
                aliensX.splice(alienIndex, 1);
                aliensY.splice(alienIndex, 1);
                alienIndex--;
                score = ~(~score + 1);
                document.getElementById("score").innerHTML = ~score;
                break;
            }
            shotIndex++;
        }
        alienIndex++;
    }

    while (alienBombIndex < alienBombsX.length) { // check whether player has been touched by a bomb
        alienBombLeft = alienBombsX[alienBombIndex];
        alienBombRight = alienBombLeft + imageAlienBomb.width;
        alienBombTop = alienBombsY[alienBombIndex];
        alienBombBottom = alienBombTop + imageAlienBomb.height;
        if (alienBombRight >= currPlayerX && playerRight >= alienBombLeft && alienBombBottom >=
               currPlayerY && alienBombTop <= playerBottom) {
            loseLife = true;
            return;
        }
        alienBombIndex++;
    }
}

function handleKeyDown(event) {
    if (event.keyCode === 80) {
        pauseButton = !pauseButton; // P pressed
        lastAnimationTime = new Date().getTime();
    }
    else if (event.keyCode === 27) {// ESC
        loseLife = true;
        numOfLives = 0;
    }
    else if (event.keyCode === 32) {// fire shot
        fireButton = true;
    }
    else if (event.keyCode === 37) { // move left
        if (moveLeft) currPlayerSpeed += 2;
        moveLeft = true;
        moveRight = false;
```

```
    }
    else if (event.keyCode === 39) { // move right
        if (moveRight) currPlayerSpeed += 2;
        moveRight = true;
        moveLeft = false;
    }
}

function handleKeyUp(event) {
    if (event.keyCode === 32) {// fire shot
        fireButton = false;
    }
    else if (event.keyCode === 37) { // move left
        moveLeft = false;
        currPlayerSpeed = initialPlayerSpeed;
    }
    else if (event.keyCode === 39) { // move right
        moveRight = false;
        currPlayerSpeed = initialPlayerSpeed;
    }
}

function restartLevel() {
    init();
    lastAnimationTime = 0;
    gameLoop(new Date().getTime());
}

// This is the function that controls the game. Called by newGame and requestAnimationFrame.
function gameLoop(currTime) {
    if (pauseButton) requestAnimationFrame(gameLoop);  // if paused, do nothing
    else {
        var timeElapsed = currTime - lastAnimationTime;
        if (lastAnimationTime === 0) adjustmentFactor = 0.2;
        else adjustmentFactor = timeElapsed / 100;
        context.clearRect(0, 0, canvas.width, canvas.height);
        // If space key pressed and enough time since the last shot has elapsed, we shoot again.
        if (fireButton && ((currTime - timeOfLastShot) > minFireThreshold)) {
            timeOfLastShot = currTime;
            if (numShots === 3) {
                playerShotsX.push(currPlayerX);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX, currPlayerY);
                playerShotsX.push(currPlayerX + (imagePlayer.width - imageShot.width) / 2);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + (imagePlayer.width - imageShot.width) /
                      2, currPlayerY);
                playerShotsX.push(currPlayerX + imagePlayer.width - imageShot.width);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + imagePlayer.width - imageShot.width,
                      currPlayerY);
            }
            else if (numShots === 2) {
                playerShotsX.push(currPlayerX);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX, currPlayerY);
                playerShotsX.push(currPlayerX + imagePlayer.width - imageShot.width);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + imagePlayer.width - imageShot.width,
```

```
                currPlayerY);
            }
            else {
                playerShotsX.push(currPlayerX + (imagePlayer.width - imageShot.width) / 2);
                playerShotsY.push(currPlayerY);
                context.drawImage(imageShot, currPlayerX + (imagePlayer.width - imageShot.width) /
                      2, currPlayerY);
            }
        }
        if (moveLeft) { // If left arrow key pressed, move spaceship to the left.
            if (currPlayerX > currPlayerSpeed) currPlayerX -= currPlayerSpeed * adjustmentFactor;
            else currPlayerX = 1;
        }
        if (moveRight) { // If right arrow key pressed, move spaceship to the right.
            if ((currPlayerX + imagePlayer.width) < (canvas.width - currPlayerSpeed))
                currPlayerX += currPlayerSpeed * adjustmentFactor;
            else currPlayerX = canvas.width - imagePlayer.width - 1;
        }
        for (var i = 0; i < aliensX.length; i++) { // generate bombs
            if (Math.random() > (1 - (currLevel * adjustmentFactor / 3000))) {
                alienBombsX.push(aliensX[i] + (imageAlien.width - imageAlienBomb.width) / 2);
                alienBombsY.push(aliensY[i] + imageAlien.height);
                context.drawImage(imageAlienBomb, aliensX[i] + (imageAlien.width -
                      imageAlienBomb.width) / 2, aliensY[i] + imageAlien.height);
            }
        }
        checkCollisions();
        if (loseLife) { // If we have been hit or touched...
            if (numOfLives >= 1) document.getElementById("life" + numOfLives).style.display =
               "none";
            numOfLives--;
            loseLife = false;
            if (numOfLives < 0) gameOver();
            else restartLevel();
        }
        else if (aliensX.length === 0) nextLevel();
        else { // Move everything and continue the fun.
            alienXSpeed = (5 + currLevel) * adjustmentFactor;
            context.drawImage(imagePlayer, currPlayerX, currPlayerY);
            moveShotsAndBombs();
            moveAliens();
            lastAnimationTime = new Date().getTime();
            requestAnimationFrame(gameLoop); //setTimeout("gameLoop()", timeOut);
        }
    }
}

function newGame() {
    document.getElementById("splash").style.display = "none";
    document.getElementById("board").style.display = "block";
    score = ~0;
    currLevel = 1;
    currPlayerSpeed = initialPlayerSpeed;
    // Display spare spaceships.
    for (var i = 1; i <= initialNumOfLives; i++) document.getElementById("life" + i).style.display =
       "inline"; //src="playerspaceship.png";
    numOfLives = initialNumOfLives;
    loseLife = false;
    fireButton = false;
```

```
    currPlayerX = (canvas.width - imagePlayer.width) / 2;
    currPlayerY = canvas.height - imagePlayer.height - 5;
    onkeydown = handleKeyDown;
    onkeyup = handleKeyUp;
    numShots = 1;
    minFireThreshold = 300;
    pauseButton = false;
    init();
    lastAnimationTime = 0;
    gameLoop(new Date().getTime());
}

function nextLevel() {
    currLevel++;
    if (currLevel >= 25) numShots = 3;
    else if (currLevel >= 15) numShots = 2;
    if (currLevel >= 20) minFireThreshold = 100;
    else if (currLevel >= 10) minFireThreshold = 200;
    document.getElementById("level").innerHTML = currLevel;
    lastAnimationTime = 0;
    init();
    gameLoop(new Date().getTime());
}

function gameOver() {
    onkeydown = null;
    onkeyup = null;
    alert("Game over!");
    updateHighScores();
    displayHighScores();
}

function displayHighScores() {
    if (highScoresReady) {
        document.getElementById("board").style.display = "none";
        document.getElementById("splash").style.display = "none";
        document.getElementById("highScoreSection").style.display = "block";
    }
    else setTimeout("displayHighScores()", 100);
}

function hideHighScores() {
    showSplash();
}

function readHighScores(scores) { // callback for PHP
    highScoreList = scores;
    var HL = document.getElementById("highScoreListTable");
    if (HL) {
        var childNodes = HL.tBodies;
        var x = childNodes.length;
        while (x > 0) {
            HL.removeChild(childNodes[0]);
            x = childNodes.length;
        }
    }
    var body = document.createElement("tbody");
    for (var i = /*Math.min(9,*/ highScoreList.length - 1/*)*/; i >= 0; i--) {
        var row = body.insertRow(0);
```

```
        var cell1 = row.insertCell(0);
        var cell2 = row.insertCell(1);
        var cell3 = row.insertCell(2);
        var cell4 = row.insertCell(3);
        cell1.innerHTML = (i + 1);
        cell2.innerHTML = highScoreList[i].Player.slice(0, Math.min(30,
              highScoreList[i].Player.length));
        cell3.innerHTML = highScoreList[i].HighScore;
        cell4.innerHTML = highScoreList[i].Level;
        cell1.style.cssText = "text-align: right";
        cell3.style.cssText = "text-align: right";
        cell4.style.cssText = "text-align: right";
    }
    HL.appendChild(body);
    highScoresReady = true;
}

function loadSaveHighScores(action, player) { // action = "insert" or nothing
    var newScriptElement = document.createElement("script");
    highScoresReady = false;
    if (action && player) newScriptElement.setAttribute("src", DBServerURL +
             "readHighScores&action=insert&player=" + player + "&score=" + ~score + "&level=" +
             currLevel);
    else newScriptElement.setAttribute("src", DBServerURL + "readHighScores");
    newScriptElement.setAttribute("id", "jsonp");
    var oldScriptElement = document.getElementById("jsonp");
    var head = document.getElementsByTagName("head")[0];
    if (oldScriptElement === null) head.appendChild(newScriptElement);
    else head.replaceChild(newScriptElement, oldScriptElement);
}

function updateHighScores() {
    loadSaveHighScores();
    if (~score > 0) {
        lastName = prompt("Enter your name", lastName);
        loadSaveHighScores("insert", lastName);
    }
    if (lastName === null) lastName = "";
}

function showSplash() {
    document.getElementById("splash").style.display = "block";
    document.getElementById("highScoreSection").style.display = "none";
    document.getElementById("keysSection").style.display = "none";
    document.getElementById("board").style.display = "none";
}

function displayKeys() {
    document.getElementById("splash").style.display = "none";
    document.getElementById("highScoreSection").style.display = "none";
    document.getElementById("keysSection").style.display = "block";
    document.getElementById("board").style.display = "none";
}

function hideKeys() {
    showSplash();
}

loadSaveHighScores();
```

9.2.2.2 *Summative Evaluation*

Die abgegebenen Problemlösungen befinden sich auf der beigelegten CD und sind hier aus Platzgründen (über hundert Seiten Quellcode) nicht abgedruckt.

9.3 *Schülerumfrage*

9.3.1 CLISS1

		Gilles Everling						Alle außer Gilles Everling						Insgesamt					
		Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung	Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung	Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung
1	Die Problemstellungen haben mich motiviert	1	2,6	3,0	3,0	5	1,2	1	3,2	3,0	3,0	5	1,1	1	3,0	3,0	3,0	5	1,1
2	Die Übungsaufgaben haben mich motiviert	1	2,6	2,0	2,0	5	1,3	1	3,2	3,0	4,0	5	1,1	1	3,0	3,0	4,0	5	1,2
3	Die automatische Rückmeldung bei den Übungsaufgaben hat meinen Lernprozess gefördert	1	2,8	3,0	2,0	5	1,1	1	3,0	3,0	3,0	5	1,0	1	2,9	3,0	3,0	5	1,1
4	Ich habe mindestens soviel für diesen Kurs wie für einen herkömmlichen Kurs gearbeitet	1	2,8	3,0	1,0	5	1,5	1	3,0	3,0	4,0	5	1,4	1	2,9	3,0	1,0	5	1,4
5	Ich fühlte mich überfordert	1	2,7	3,0	1,0	5	1,4	1	2,6	3,0	3,0	5	1,2	1	2,6	3,0	1,0	5	1,3
6	Ich habe ein klares Verständnis der grundlegenden Konzepte von JavaScript	1	3,2	3,0	3,0	5	1,3	1	3,5	4,0	4,0	5	1,2	1	3,4	4,0	4,0	5	1,2
7	Ich kann dieses Verständnis zur Lösung noch nicht behandelter Probleme einsetzen	1	3,0	3,0	4,0	5	1,2	1	3,2	3,0	3,0	5	1,2	1	3,1	3,0	3,0	5	1,2
8	Meine Fähigkeit, Probleme zu analysieren, hat sich verbessert	1	2,9	3,0	3,0	5	1,1	1	3,3	4,0	4,0	5	1,2	1	3,2	3,0	3,0	5	1,2
9	Ich kann komplexe Skripte erstellen und optimieren	1	2,6	3,0	3,0	5	1,1	1	3,1	3,0	2,0	5	1,4	1	2,9	3,0	3,0	5	1,3
10	Ich kann fehlerhafte Programme mittels Debuggertools analysieren	1	3,0	3,0	3,0	5	1,0	1	3,6	4,0	4,0	5	1,3	1	3,4	3,0	3,0	5	1,3
11	Meine Fähigkeit, Informationen selbstständig in Referenzen nachzuschlagen und einzusetzen, hat sich erhöht	1	2,7	3,0	3,0	5	1,2	1	3,2	3,0	4,0	5	1,0	1	3,0	3,0	3,0	5	1,1
12	Mein Verständnis meines eigenen Lernprozesses hat sich verbessert	1	2,5	3,0	3,0	5	1,3	1	3,0	3,0	2,0	5	1,1	1	2,8	3,0	3,0	5	1,1
13	Ich gehe systematischer an die Lösung eines Problems heran	1	2,9	3,0	3,0	5	1,3	1	3,4	4,0	4,0	5	1,2	1	3,2	3,0	4,0	5	1,3
14	Ich arbeite gerne in einer Gruppe	1	3,9	4,0	5,0	5	1,4	1	3,7	4,0	5,0	5	1,3	1	3,8	4,0	5,0	5	1,3
15	Ich arbeite am liebsten alleine	1	2,4	2,0	1,0	5	1,3	1	3,1	3,0	5,0	5	1,5	1	2,8	3,0	1,0	5	1,4
16	Ich hätte die Gruppenarbeit am liebsten fortgesetzt	1	3,3	3,0	5,0	5	1,4	1	3,9	4,0	5,0	5	1,2	1	3,6	4,0	5,0	5	1,3
17	Das Gespräch mit anderen hat mir geholfen, Probleme besser zu verstehen	1	3,1	3,0	3,0	5	1,4	1	3,8	4,0	4,0	5	1,0	1	3,5	4,0	4,0	5	1,2
18	Das Gespräch mit anderen hat mir geholfen, Probleme besser zu lösen	1	3,1	3,0	3,0	5	1,4	1	3,8	4,0	4,0	5	1,0	1	3,5	4,0	4,0	5	1,2
19	Ich kann die Lösung eines Problems effektiv planen	1	3,1	3,0	3,0	5	1,3	1	3,2	3,0	3,0	5	1,1	1	3,2	3,0	3,0	5	1,2
20	Ich kann meinen Lösungsansatz erklären und begründen	2	3,6	3,5	3,0	5	1,1	1	3,5	4,0	4,0	5	1,1	1	3,6	4,0	3,0	5	1,1
21	Ich kann meine eigenen Lernziele bestimmen und formulieren	1	3,1	3,0	3,0	5	1,2	1	3,5	4,0	3,0	5	1,0	1	3,4	3,0	3,0	5	1,1
22	Ich habe mein Wissen und meine Fähigkeiten voll eingesetzt	1	2,9	3,0	3,0	5	1,1	1	3,5	3,0	3,0	5	1,1	1	3,3	3,0	3,0	5	1,1
23	Ich habe ein klares Verständnis meiner Stärken und Schwächen bezüglich des Lehrplans entwickelt	1	3,4	3,5	3,0	5	1,2	1	3,5	4,0	4,0	5	1,0	1	3,5	4,0	4,0	5	1,1
24	Ich bin zufrieden mit dem was ich erreicht habe	1	3,1	4,0	4,0	5	1,6	1	2,8	2,0	1,0	5	1,6	1	2,9	3,0	1,0	5	1,6
25	Ich bevorzuge problembasierten gegenüber herkömmlichem Unterricht	1	2,7	3,0	3,0	5	1,4	1	3,2	3,0	3,0	5	1,3	1	3,0	3,0	3,0	5	1,3
			3,0	3,0	2,9		1,3		3,3	3,4	3,5		1,2		3,2	3,3	3,1		1,2

9.3.2 CLISS2

		Gilles Everling						Alle außer Gilles Everling						Insgesamt					
		Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung	Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung	Minimum	Durch-schnitt	Median	Häufigster Wert	Maximum	Standard-abweichung
1	Ich kann Probleme in Teilprobleme zerlegen und Funktionen erstellen, welche diese lösen	1	3,7	4,0	5,0	5	1,3	1	3,2	3,0	3,0	5	1,2	1	3,3	3,0	3,0	5	1,2
2	Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt	1	2,9	3,0	1,0	5	1,5	1	3,3	3,5	4,0	5	1,3	1	3,2	3,0	4,0	5	1,3
3	Ich beherrsche das Erstellen von interaktiven Webseiten	1	3,5	4,0	4,0	5	1,4	1	3,4	3,0	3,0	5	1,3	1	3,4	3,0	3,0	5	1,3
4	Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt	1	3,0	3,0	1,0	5	1,5	1	3,2	3,0	4,0	5	1,3	1	3,1	3,0	4,0	5	1,4
5	Ich kann vorgefertigte Codelösungen an meine eigene Webseite anpassen und einbinden	1	4,2	4,0	4,0	5	1,0	1	3,6	4,0	4,0	5	1,2	1	3,7	4,0	4,0	5	1,2
6	Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt	1	3,0	3,0	4,0	5	1,4	1	3,1	3,0	3,0	5	1,2	1	3,1	3,0	3,0	5	1,3
7	Ich arbeite selbstständig und motiviert	1	3,5	3,0	3,0	5	1,3	1	3,4	3,0	3,0	5	1,2	1	3,4	3,0	3,0	5	1,2
8	Meine Kompetenz bzgl. des vorherigen Punktes hat sich dieses Semester deutlich weiter entwickelt	1	2,9	3,0	3,0	5	1,4	1	3,1	3,0	3,0	5	1,3	1	3,1	3,0	3,0	5	1,3
9	Meine Fähigkeit konstruktiv in einer Gruppe zu arbeiten hat sich erhöht	1	2,7	3,0	3,0	5	1,3	1	3,4	4,0	5,0	5	1,5	1	3,2	3,0	5,0	5	1,4
10	Ich habe meinen eigenen Stil entwickelt um Probleme zu analysieren und zu lösen und bin mir dessen bewusster geworden	1	3,1	3,0	4,0	5	1,5	1	3,4	4,0	4,0	5	1,3	1	3,3	4,0	4,0	5	1,3
11	Meine Fähigkeit zu erkennen, welche Informationen mir fehlen und diese gezielt und effizient zu finden hat sich deutlich verbessert	1	2,9	3,0	1,0	5	1,4	1	3,3	3,0	3,0	5	1,1	1	3,2	3,0	3,0	5	1,2
12	Ich habe ein besseres Verständnis für meinen eigenen Lernstil entwickelt	1	2,6	3,0	1,0	5	1,4	1	3,1	3,0	4,0	5	1,3	1	3,0	3,0	4,0	5	1,4
13	Die behandelten Problemstellungen haben mich motiviert	1	3,3	3,0	5,0	5	1,4	1	2,7	3,0	3,0	5	1,2	1	2,9	3,0	3,0	5	1,3
14	Mein Selbstvertrauen auch anspruchsvolle Problemstellungen lösen zu können hat sich gesteigert	1	2,8	3,0	1,0	5	1,5	1	3,3	3,5	4,0	5	1,3	1	3,1	3,0	4,0	5	1,4
15	Der Lehrer war hauptsächlich als Instruktor tätig und hat mittels Vorlesungen Wissen vermittelt	2	3,8	4,0	3,0	5	0,9	1	3,2	3,0	3,0	5	1,3	1	3,3	3,0	3,0	5	1,2
16	Der Lehrer war hauptsächlich als Tutor tätig und hat mich dabei unterstützt, meine eigenen Lernprozesse besser zu verstehen und zu entwickeln	2	3,8	4,0	5,0	5	1,1	1	3,5	4,0	3,0	5	1,2	1	3,6	4,0	3,0	5	1,2
17	Ich habe hauptsächlich mittels Recherche im Internet und in Büchern die mir fehlenden Informationen gefunden	1	3,4	4,0	4,0	5	1,4	1	3,5	3,0	3,0	5	1,2	1	3,4	4,0	4,0	5	1,3
18	Ich habe hauptsächlich durch das Gespräch mit meinen Klassenkollegen die mir fehlenden Informationen gefunden	1	3,4	4,0	4,0	5	1,3	1	3,4	3,0	4,0	5	1,2	1	3,4	3,0	4,0	5	1,2
19	Durch die Diskussionen in der Gruppe habe ich andere Perspektiven kennen gelernt, die mich voran gebracht haben	1	2,8	3,0	3,0	5	1,2	1	3,1	3,0	3,0	5	1,4	1	3,0	3,0	3,0	5	1,3
20	CLISS2 hat mir viel Spaß gemacht	1	3,4	3,0	3,0	5	1,4	1	3,0	3,0	4,0	5	1,4	1	3,1	3,0	4,0	5	1,4
21	Ich habe hart gearbeitet	1	2,8	3,0	3,0	5	1,4	1	3,3	3,0	3,0	5	1,2	1	3,2	3,0	3,0	5	1,3
22	Ich bin stolz auf das was ich erreicht habe	1	3,0	3,0	3,0	5	1,4	1	3,0	3,0	3,0	5	1,4	1	3,0	3,0	3,0	5	1,4
23	Ich habe mir dieses Jahr die Grundlagen der Programmierung erarbeitet, auf denen ich in den nächsten Jahren aufbauen werde	1	2,8	3,0	1,0	5	1,5	1	3,3	3,5	4,0	5	1,3	1	3,2	3,0	4,0	5	1,4
			3,2	3,3	3,0		1,3		3,2	3,2	3,5		1,3		3,2	3,2	3,5		1,3

9.4 *Lehrpläne und Evaluierungsraster*

9.4.1 CLISS1

Lehrplan:

Der Aufbau des Kurses basiert auf dem **obligatorischen** Kursbuch: „Herdt JavaScript 1.8 Grundlagen".

K:	Aufgaben/ Lernsituationen:	Hinweise: Inhalte/Methoden/Material	St.
SA1, SZE1	Einführung in die Programmierung	• Aktivität „Programmiersprachen" von http://csunplugged.org/programming-languages • Aktivität „LightBot" von http://www.lightbot.lu *Hinweise:* • Schülerdemo • Lehrerdemo	2
SA1, SA3, SZE1	Grundlegende Sprachelemente erlernen anhand von Eingabe- und Ausgabe-Fenster	• Kapitel 2,3, 8.4, 8.5 • Wertausgabe (*document.write()*, Meldungsfenster) • Werteingabe (Eingabefenster) • Variablen und Datentypen • Operatoren *Hinweise:* • Skripte werden direkt im Body, ohne Funktionen zu benutzen, erstellt und aufgerufen • Kommentare als Debugginghilfe • Firebug einsetzen: Breakpoints und Watches	10
SA1, SA3, SZE1	Bestätigungsfenster und Alternativ-Struktur einsetzen	• Kapitel: 3.8, 4.1-4.3, 8.6 • Alternativ-Struktur (mit else) • Mehrstufige if-else Anweisung • Bestätigungsfenster *Hinweise:* • Keine Verschachtelung • Debugging mittels Firebug • Kein switch case • Strikte Gleichheit (===) beim Vergleich benutzen • Strikte Ungleichheit (!==) beim Vergleich benutzen	8
SA1, SA3, SZE1	Eingabefelder, Schaltfächen einsetzen	• Kapitel: 5.1, 8.8 (nur *getElementById*), 9.5, 9.8, 10.1, 10.2 • Funktionen ohne Parameter • Eingabefelder und *getElementById* • Auslesen und ausgeben von CSS-Eigenschaften eines DOM-Objektes • Schaltfächen einsetzen *Hinweise:* • Funktionen nur zur Eventauswertung • Nur onClick-Event mit parameterlosen Funktionen im HTML-Kopf • Wiederholung der Alternativ-Struktur • Grundbegriffe der Objektorientierung (Objekte, Eigenschaften, Methoden)	8

SA1, SA3, SZE1	Schleifen einsetzen	• Kapitel: 4.4, 9.7
		• Benutzen der Timer-Methoden
		
		• Aufgabenbeispiele:
		  ○ Bewegen von DOM-Objekten mit Hindernissen
		  ○ Positionieren eines DOM-Objektes anhand einer Auswahlliste
			16

		○ Werte der ausgewählten Einträge in einer Auswahlliste anzeigen ○ … *Hinweise:* • Schleifen im Zusammenhang mit Auswahllisten erklären • Zugriff auf eindimensionale Felder erklären (keine Felder erstellen!) • Zähler- und kopfgesteuerte Schleifen (*for, while*) • Schleifen mit Objektpositionierung einsetzen • Verschachteln von Schleifen mit Alternativen • Schleifen zum Rechnen einsetzen	
SA1, SA2, SA3, SZE1	Eindimensionale Felder erstellen und einsetzen	• Kapitel: 7.5 • Abspeichern der (aktuellen/Weg-) Positionen eines beweglichen DOM-Objektes • Benutzen der sort-Methode zum Sortieren einer Auswahlliste • Erstellen einer dynamisch gefüllten Auswahlliste • Programmieren eines animierten Objektes welches einen zuvor mit der Maus festgelegten Weg zurücklegt (Eine Funktion zum Ermitteln der Mausposition wird zur Verfügung gestellt) → SA2 *Hinweise:* • Erstellen der Felder nur mit: var name = [3,5,77,34]; • Das Benutzen der Methoden *push*, *pop* und *shift* ist Teil der Kompetenz SA2 „Komplexe Skripte erstellen und optimieren"	12
SA1, SA2, SA3, SZE1	Übungsaufgaben zu diversen Themen	• Kapitel 7, 10.2 • Vordefinierte Objekte • Tastaturereignis: z.B. onKeyUp • Ändern von CSS-Eigenschaften • Zugreifen auf DOM-Elemente • Vereinfachte Version des „Leap frog game" • …	8
		TOTAL:	64

Allgemeine Anmerkungen

Die restlichen 8 Stunden dienen der Evaluation.

Kompetenzen:

- SA2, SZE1 ergeben sich aus den zu lösenden Aufgaben

Obligatorische Lehrbücher

- Herdt JavaScript 1.8 Grundlagen, JAVS18, Januar 2010 oder neuer
- Offizieller Kursablauf (Dokument oder Website)

Begleitende Lehrbücher

- https://developer.mozilla.org/en/JavaScript/Guide/
- http://www.w3schools.com/js/

Evaluierungsraster :				
Typ	**Kat**	**Kompetenzen**	**Indikatoren**	**Standards**
O	SA1	**Die grundlegenden Konzepte von JavaScript in der Entwicklung von dynamischen Webseiten einsetzen**	Integrieren von JavaScript-Code beim Erstellen von CSS-formatierten HTML-Seiten Anwenden von grundlegenden Sprachelementen (Variablen, Konstanten, Operatoren,...) Anwenden von elementaren Kontrollstrukturen (Schleifen, Alternativen, ...) Anwenden von vordefinierten Funktionen (Meldungsfenster, Eingabefenster, Number, String, parseFloat, ...) Erstellen von einfachen benutzerdefinierten Funktionen ohne Parameter Verschachteln von Kontrollstrukturen: maximale Verschachtelungstiefe 1 Benutzen der vordefinierten JavaScript-Objekte: String, Math, Number und Date Benutzen und erstellen von eindimensionalen Feldern (Array-Objekt) Reagieren auf Maus-Ereignisse (onClick) Reagieren auf Tastatur-Ereignisse (z.B. onKeyUp) Zugreifen und ändern der DOM-Elemente: Document, Input, Button, Bilder, ...	70% der festgelegten Schwerpunkte in der Aufgabenstellung sind gelöst
S	SA2	**Komplexere Skripte erstellen und optimieren**	Einsetzen von komplexen Verschachtelungen: minimale Verschachtelungstiefe 2 Optimieren von selbst erstellten und gegebenen Skripten Einsetzen von nicht im Kurs behandelten JavaScript Objekten Einsetzen von 2 dimensionalen Feldern	50% der Schwerpunkte in der Aufgabenstellung sind gelöst Mehr als 85% der Schwerpunkte der Kompetenz „Die grundlegenden Konzepte von JavaScript in der Entwicklung von dynamischen Webseiten einsetzen“ werden erreicht
S	SA3	**Debuggertools bei der Fehleranalyse im Programmierprozess einsetzen**	Analysieren von fehlerhaften Skripten (vordefinierte oder eigene Skripte) Anzeigen von Hilfsinformationen in der Webseite (alert und document.write) Einsetzen eines Debuggertools wie z.B. Firebug, (Breakpoint, Watches, ...) Analysieren von logischen Fehlern Analysieren von Fehlermeldungen	Vordefinierte Skripte: 80% der Fehler sind behoben Eigene Skripte: Die Vorgehensweise der Fehleranalyse und Fehlerbeseitigung ist erläutert
S	SZE1	**Informationen selbstständig in Referenzen nachschlagen und einsetzen**	Recherchieren von Lösungen im Internet und in Büchern (Die Referenzen sollten vorgegeben werden) Integrieren des Lösungsansatzes	Das Problem ist weitestgehend gelöst Die Quellen sind angegeben
Anzahl der zu evaluierenden selektiven Kompetenzen : 2				

9.4.2 CLISS2

Lehrplan:

Der Aufbau des Kurses basiert auf dem **obligatorischen** Kursbuch: „Herdt JavaScript 1.8 Grundlagen". Dieses Semester dient vor allem der Vertiefung der im ersten Semester erlernten Konzepte.

K:	Aufgaben/ Lernsituationen:	Hinweise: Inhalte/Methoden/Material	St.
SA1, SA3	Funktionen erstellen und anwenden	• Funktionen mit und ohne Parameter anwenden • Funktionen mit und ohne Parameter erstellen • Ausgelagerten JavaScript Code einbetten • Bestehenden Quellcode mittels Funktionen modularisieren (Probleme aufteilen und auslagern) • Bestehende Lösungen können vom Lehrer vorgegeben werden oder im Internet gesucht werden • Einführung in die Lizenzbestimmungen (GPL, ...) in Bezug auf fremden Quellcode Hinweise: • Die im ersten Semester erlernten Konzepte sollen wiederholt und vertieft werden • Beim Erstellen der Funktionen kann es sich um neue Aufgaben sowie bestehende, nicht modularisierte Aufgaben handeln. • Diese Lernsituation wird auch in den folgenden Lernsituationen wieder zu finden sein	20
SA1, SA2, SA3	Interaktive Webseiten erstellen	• Formularvalidierung durchführen (z.B.: Text oder Zahl, Zeichenkettenlänge, Emailadresse, Datum, obligatorische Felder, ...) • Simulieren eines rechnungsbasierten Captcha's (Bsp.: 2+3*4=?) • Dynamische Elemente im DOM hinzufügen (z.B.: Bilder, Inputfelder, ...) • Benutzen der Timer-Methoden	12
SA3	jQuery in Webseiten einsetzen	• Vor- und Nachteile erläutern • Anpassen bestehender Aufgaben Hinweise: • Online Editor JSFIDDLE.net benutzen	8
		TOTAL:	40

Allgemeine Anmerkungen

Die restlichen 8 Stunden dienen der Evaluation.

Kompetenzen:

- SZE1 ist in allen Lernsituationen wieder zu finden.

Obligatorische Lehrbücher

- Herdt JavaScript 1.8 Grundlagen, JAVS18, Januar 2010 oder neuer
- Offizieller Kursablauf (Dokument oder Website)

Begleitende Lehrbücher

- https://developer.mozilla.org/en/JavaScript/Guide/
- http://www.w3schools.com/js/
- http://jquery.com/

LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Education nationale
et de la Formation professionnelle

Evaluierungsraster :				
Typ	**Kat**	**Kompetenzen**	**Indikatoren**	**Standards**
O	SA1	**Modularen Quellcode erstellen und in Webseiten einbinden**	Modularisieren von bestehendem Quellcode Auslagern von JavaScript Quellcode in eine externe Datei Zerlegen von Problemen in Teilprobleme Erstellen von Funktionen welche die Teilprobleme lösen Einbinden von Funktionen in die Webseite	70% der zu erstellenden Funktionen sind vorhanden und können erfolgreich eingesetzt werden
O	SA2	**Erstellen von interaktiven Webseiten**	Validieren der Dateneingabe in Formularen Durchführen von DOM-Modifikationen (erstellen und ändern von DOM-Objekten) Verändern von CSS-Eigenschaften mittels JavaScript Erarbeiten eines Gesamtkonzeptes zum Aussehen und Verhalten der Website Anwenden der Gestaltungsregeln von HTML und CSS Erstellen von komplexerem JavaScript-Code	70% der festgelegten Schwerpunkte in der Aufgabenstellung sind gelöst
S	SA3	**Einbinden von vorgefertigten Codelösungen in der eigenen Webseite**	Gezieltes Suchen nach Quellcode-Lösungen Anpassen einer vorgefertigten Lösung an das vorgegebene Problem Beachten der Lizenzen zur Benutzung des Quellcodes von Drittpersonen	Fremder und eigener Quellcode können zusammen arbeiten Die Quellen werden angegeben
S	SZE1	**Selbstständig und motiviert arbeiten**	Eigenständig Probleme lösen Gegebene Aufgaben erweitern Recherchen durchführen Weiterführende Fragen stellen	Die Aufgaben- und Problemstellungen sind von den Schüler intensiv und eigenständig erarbeitet
Anzahl der zu evaluierenden selektiven Kompetenzen : 1				

Typ Obligatorisch oder Selektiv
Kat Sachkompetenz oder Sozial- bwz. Selbstkompetenz
SA Sachkompetenz
SZE Sozial- bwz. Selbstkompetenz
SAx Sachkompetenz Nummer x aus der Modulbeschreibung
SZEx Sozial- bwz. Selbstkompetenz Nummer x aus der Modulbeschreibung