STAATSINSTITUT FÜR SCHULQUALITÄT
UND BILDUNGSFORSCHUNG
MÜNCHEN

GYMNASIUM

HANDREICHUNG

Informatik am Naturwissenschaftlich-technologischen Gymnasium Jahrgangsstufe10

Erläuterungen und Materialien
für Lehrkräfte

Informatik

Vorwort

Mit der Einführung des achtjährigen Gymnasiums erhielt die Informatik in der Mittelstufe des Naturwissenschaftlich-technologischen Gymnasiums den Status eines Pflichtfachs. Dadurch wurde ein wesentlicher Beitrag zu einer zukunftsorientierten Ausbildung der bayerischen Schülerinnen und Schüler geleistet. Die übergeordneten Ziele dieser Ausbildung sind im Fachprofil Informatik des Lehrplans folgendermaßen formuliert: „Die wesentliche Aufgabe des Informatikunterrichts am Gymnasium ist es […], den Schülern ein systematisches, zeitbeständiges und über bloße Bedienerfertigkeiten hinausgehendes Basiswissen über die Funktionsweise, die innere Struktur sowie die Möglichkeiten und Grenzen informationstechnischer Systeme zu vermitteln. Dadurch wird ihnen deren sinnvolle, kompetente und verantwortungsbewusste Nutzung und Beurteilung ermöglicht. Als zukünftige Entscheidungsträger müssen die Gymnasiasten mit den Denkweisen vertraut gemacht werden, die den Informations- und Kommunikationstechniken zugrunde liegen, um deren prinzipielle Chancen und Risiken richtig einschätzen zu können."

Um die im Fachprofil geforderte Zielsetzung zu realisieren, muss bereits nach Jahrgangsstufe 10 ein erster Abschluss bei den Kenntnissen und Fertigkeiten zur Umsetzung informatischer Grundkonzepte erreicht sein, da Informatik in der Qualifikationsphase der Oberstufe nicht zwangsläufig weitergeführt wird. Nachdem die Jugendlichen aus Jahrgangsstufe 9 mit der funktionalen Modellierung und der statischen Datenmodellierung vertraut sind, ergänzen sie nun mit der Ablaufmodellierung die ihnen zur Verfügung stehenden Techniken und führen diese im Rahmen der objektorientierten Modellierung zusammen. Damit sind die Jugendlichen in der Lage, eigene kleine Systeme zu produzieren und können auch komplexere Aufgabenstellungen bearbeiten.

Die vorliegende Handreichung enthält eine Aufarbeitung von informatischem Fachwissen, ausführliche didaktisch-methodische Überlegungen, wertvolle Hinweise zur Handhabung und zu Einsatzmöglichkeiten von geeigneten Softwarewerkzeugen sowie detaillierte Stundenbilder mit entsprechenden Materialien. Es werden zwei unterschiedliche Unterrichtskonzepte vorgestellt, die verschiedene Möglichkeiten der Lehrplanumsetzung aufzeigen. An den beiden Unterrichtskonzepten wird der Gestaltungsspielraum des Lehrplans deutlich; während ein Konzept das grundlegende Anforderungsniveau des Lehrplans veranschaulicht, zeigt das andere insbesondere Möglichkeiten auf, leistungsfähigeren Schülerinnen und Schülern zusätzliche Anreize zu geben. Insgesamt konkretisieren die vorgestellten Konzepte Intention und Anforderungsniveau des Lehrplans, so dass die Auswahl von Aufgaben und Vertiefungsmöglichkeiten aus dem umfangreichen Angebot der Schulbücher unterstützt wird. Die skizzierten Stundenbilder sind als Beispiele für eine mögliche Umsetzung des Lehrplans zu verstehen. Sie sollen Anregungen für die eigene Unterrichtsgestaltung geben sowie das Ausprobieren und Entwickeln eigener Unterrichtsideen unterstützen.

München, Juli 2008 Christian Scheungrab

Bemerkungen:

- Der Kürze halber ist im Text meist von Lehrern und Schülern die Rede, gelegentlich auch im Singular. Dass das Kollegium eines Gymnasiums aus Frauen und Männern, die Schülerschaft aus Mädchen und Jungen besteht, wurde überall mit bedacht.
- Auf der Begleit-CD finden sich die Dateien, auf die im Text verwiesen wird. Diese Dateien enthalten Unterrichtsmaterialien, Hinweise zum Umgang mit dem Softwarewerkzeug BlueJ und zur Programmierung in Java.
- Die Dateistruktur auf der Begleit-CD korrespondiert mit den Bezeichnungen des Inhaltsverzeichnisses der Handreichung.
- Die Handreichung sowie die auf der Begleit-CD enthaltenen Materialien stehen in einer Entwurfsfassung auch auf der Homepage des ISB (www.isb.bayern.de → Gymnasium → Fach Informatik → Materialien) zur Verfügung.

Mitglieder des Arbeitskreises waren:

Elke Frey	Friedrich-Alexander-Gymnasium Neustadt a. d. Aisch
Christian Heidrich	Willibald-Gluck-Gymnasium Neumarkt
Christian Scheungrab	ISB
Matthias Spohrer	Gymnasium Fürstenried
Albert Wiedemann	Erasmus-Grasser-Gymnasium München

Inhaltsverzeichnis

Vorwort 3
Inhaltsverzeichnis 5
1 Didaktisch-methodische Vorbemerkungen 7
1.1 Ziele und Inhalte des Informatikunterrichts in Jahrgangsstufe 10 7
1.2 Vorgestellte Unterrichtskonzepte 8
1.3 Sprache und Werkzeug 9
1.4 Ergänzender Einsatz von JavaKarol 10
1.5 Fachdidaktischer Hintergrund 12
1.5.1 Grundlagen der objektorientierten Modellierung und Programmierung 12
1.5.2 Kommunikation zwischen Objekten – Assoziationen 15
1.5.3 Sequenzdiagramme 18
1.5.4 Vererbung und Polymorphie 19
1.5.5 Zustandsmodellierung 22
1.6 Hinweise zum Programmieren 26
1.6.1 Layout und Kommentare 26
1.6.2 Javadoc 27
1.6.3 this-Konstrukt 28
2 Werkzeuge 29
2.1 Die Entwicklungsumgebung BlueJ 29
2.2 JavaKarol 35
2.3 Werkzeuge zum Erstellen von Diagrammen 36
3 Unterrichtskonzepte 39
3.1 Konzept 1 39
3.1.1 Unterrichtskonzept 39
3.1.2 Arbeitsmaterial 93
3.2 Konzept 2 95
3.2.1 Unterrichtskonzept 95
3.2.2 Arbeitsmaterial 142

1 Didaktisch-methodische Vorbemerkungen

1.1 Ziele und Inhalte des Informatikunterrichts in Jahrgangsstufe 10

Im letzten Jahr der Mittelstufe beschäftigen sich die Schüler im Informatikunterricht mit weiteren Modellierungstechniken, die die Fähigkeit der Jugendlichen zu umfassender objektorientierter Modellierung vervollständigen. Nachdem die Schüler aus Jahrgangsstufe 9 mit der funktionalen Modellierung und der statischen Datenmodellierung vertraut sind, wenden sie sich nun auch der Modellierung von Abläufen zu und untersuchen die Kommunikation zwischen Objekten und deren Beziehungen. Dabei lernen sie die Grundprinzipien von Automaten und Algorithmen sowie deren Umsetzung in Programme aus objektorientierter Sicht kennen und erstellen neben den bereits verwendeten Klassendiagrammen auch Zustands- und Interaktionsdiagramme. Ihre Kenntnisse in der objektorientierten Modellierung runden sie mit dem Konzept der Generalisierung und Spezialisierung ab.

Aus den vorhergehenden Jahrgangsstufen sind die Schüler bereits mit grundlegenden Begriffen der objektorientierten Modellierung vertraut. Sie kennen die Klasse als Konstruktionsplan für Objekte und die Beschreibung von Objekten mittels Attributen und Methoden. Diese Begriffe werden wiederholt und präzisiert; die Modellierung wird graphisch in Form von Objekt- und Klassendiagrammen dargestellt („Inf 10.1.1 Zusammenfassung und Festigung der bisher erlernten objektorientierten Konzepte“). Hier ist es möglich, wiederholend auf bereits bekannte Werkzeuge zurückzugreifen wie z. B. EOS oder ObjectDraw von M. Pabst. Zur Wiederholung algorithmischer Strukturen eignet sich Karol von U. Freiberger; für die Einführung von Programmierkonzepten des Informatikunterrichts der Jahrgangsstufe 10 ist die Weiterentwicklung JavaKarol sehr gut geeignet (siehe Kapitel 1.4).

Den größten Teil im Lehrplan nimmt der Themenbereich „Inf 10.1.2 Zustände von Objekten und algorithmische Beschreibung von Abläufen“ ein. Spätestens ab hier lässt sich der Lehrplan nicht mehr sequentiell abarbeiten. Mögliche Vorgehensweisen werden in dieser Handreichung erläutert.

Algorithmen sind den Schülern bereits aus Jahrgangsstufe 7 bekannt, in der sie mithilfe eines Roboterwerkzeugs einfache Informatiksysteme unter Verwendung der Kontrollstrukturen Sequenz, bedingte Anweisung und Wiederholung programmierten. Im Unterschied zu den algorithmischen Modellen in der Unterstufe spielt jetzt das neu hinzukommende Variablenkonzept eine wesentliche Rolle. Zu dessen Beschreibung sind Zustandsmodelle hilfreich. Der Zustand eines Objektes wird durch dessen Attributwerte (Belegung seiner Variablen) eindeutig beschrieben. Eine Wertzuweisung löst unmittelbar einen Zustandsübergang aus. So lassen sich die Veränderungen eines Objektes mithilfe von Zuständen und Übergängen darstellen. Die Zustandsdiagramme einfacher Automaten werden schließlich in objektorientierte Programme umgesetzt.

Zudem werden die Beziehungen und die Kommunikation zwischen Objekten untersucht und Möglichkeiten zu ihrer Implementierung erarbeitet („Inf 10.1.3 Beziehungen zwischen Objekten“). In diesem Zusammenhang lernen die Schüler, Interaktionsdiagramme zu verwenden, und verstehen, warum das Prinzip der Datenkapselung sinnvoll ist. Eine wichtige Beziehung zwischen Objekten ist die Aggregation (Enthält-Beziehung). Die Schüler werden schließlich mit dem Prinzip der Vererbung zur Umsetzung hierarchischer Klassenstrukturen (Generalisierung bzw. Spezialisierung) und dessen Anwendungsmöglichkeiten vertraut gemacht („Inf 10.2 Generalisierung und Spezialisierung“).

Wie in Jahrgangsstufe 9 festigt auch in Jahrgangsstufe 10 ein größeres Projekt zum Schuljahresende die Lerninhalte und motiviert die Schüler dazu, eine komplexere Aufgabenstellung im Team zu meistern („Inf 10.3 Komplexeres Anwendungsbeispiel“).

1.2 Vorgestellte Unterrichtskonzepte

In dieser Handreichung werden zwei Konzepte für eine mögliche Umsetzung des Lehrplans vorgestellt. In Konzept 1 werden mehrere kleinere (Bei-)Spiele entwickelt, während in Konzept 2 themenzentriert ein größeres Projekt (Simulation der Kassensituation im Supermarkt) im Vordergrund steht, das über das ganze Schuljahr hinweg entwickelt und erweitert wird. Insbesondere aufeinander aufbauende Aufgabenbeispiele erfordern es, dass die Lehrkraft z. B. längerfristig erkrankten Schülern notwendige Lösungsteile und Zwischenschritte zur Verfügung stellen kann.

An den beiden exemplarischen Unterrichtskonzepten wird der Gestaltungsspielraum des Lehrplans deutlich, der Schwerpunktsetzungen gemäß dem didaktisch-methodischen Ermessen der Lehrkraft und gemäß der Unterrichtssituation vor Ort erlaubt. Beide Konzepte enthalten Vertiefungsmöglichkeiten mit optionalen Teilen, die je nach Leistungsfähigkeit der unterrichteten Gruppe umgesetzt werden können. Während Konzept 1 Schüler aller Leistungsstufen gleichermaßen anspricht, zeigt Konzept 2 Möglichkeiten auf, auch leistungsfähigeren Schülern zusätzliche Anreize zu geben. Insbesondere aufgrund dieser Differenzierungsmöglichkeiten ist Konzept 2 eher für leistungsfähigere Klassen geeignet, die sich durch hohe Motivation und Interesse auch über den Lehrplan hinaus auszeichnen.

Beide Konzepte unterscheiden sich nicht nur in ihrem prinzipiellen Aufbau, sondern auch in der Schreibweise bezüglich Attribut-, Methoden- und Klassenbezeichner. Konzept 1 orientiert sich bei der Notation an der Java-Konvention: Methoden werden klein geschrieben und analog zum im Englischen häufig verwendeten „*set*...“, „*get*...“ usw. wird der Imperativ verwendet, z. B. `setzeFarbe()`[1], `zeichneKreis()`,… Diese Formulierungsweise betont den Aspekt des Methodenaufrufs durch Botschaften, der bei der Kommunikation zwischen Objekten auftritt.

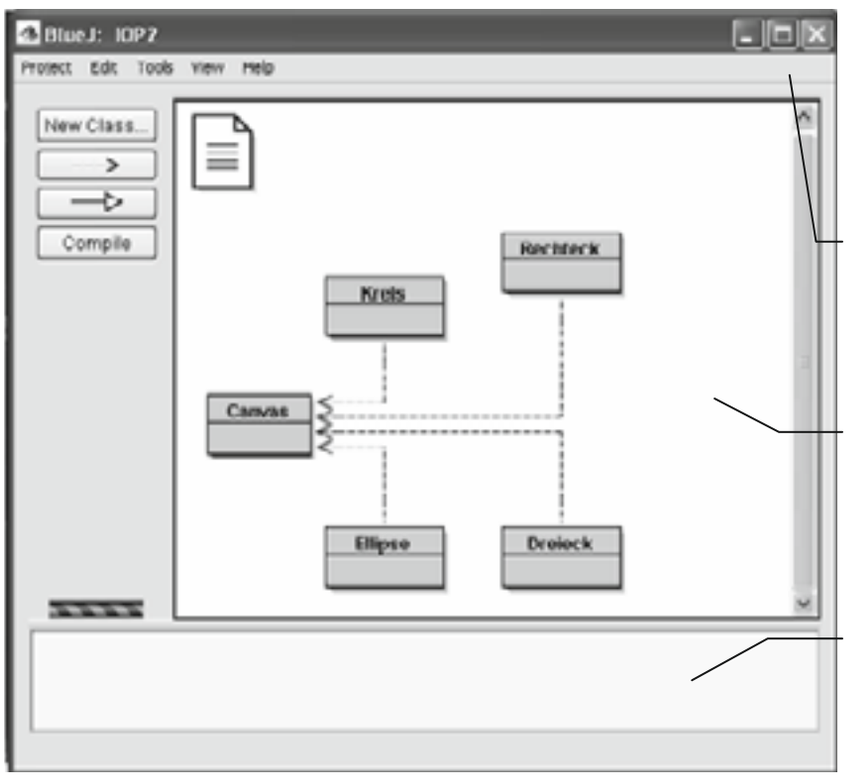
In Konzept 2 wird die aus der Unterstufe schon bekannte Notation mit Infinitiv verwendet, z. B. `FarbeSetzen()` oder `KreisZeichnen()`. Hier beginnen entgegen der Java-Konvention Methoden mit einem Großbuchstaben; durch die Verwendung großer Anfangsbuchstaben lassen sich selbst definierte Methoden leichter von bereits vorhandenen Methoden unterscheiden. Die Verwendung des Infinitivs betont, dass offensichtlich das Objekt selbst die Methode ausführt und Methoden Dienste sind, die das Objekt anbietet, die also zum Objekt gehören.

[1] In der Regel wird im Fließtext dieser Handreichung zur Erhöhung der Lesbarkeit auf die Angabe der Parameter von Methoden verzichtet.

2 Werkzeuge

2.1 Die Entwicklungsumgebung BlueJ

BlueJ ist eine Entwicklungsumgebung für Java, die speziell für den Schulunterricht entwickelt wurde. BlueJ gibt es für Windows-, MacOS- und andere Systeme. Diese Entwicklungsumgebung ist für den privaten Gebrauch kostenlos und kann unter www.bluej.org heruntergeladen werden (aktuelle Version 2.2.1, Stand Mai 2008). Voraussetzung ist eine installierte Java-Umgebung (Java SE Development Kit (JDK) 1.5 oder höher, kostenlos erhältlich unter java.sun.com/javase).



Menüleiste: Enthält alle wichtigen Anweisungen.

Klassenfenster: Zeigt das Klassendiagramm an.

Objektfenster (Objektleiste): Zeigt alle bereits erzeugten Objekte an.

Mit dem Button „New Class“ kann man eine neue Klasse definieren.

Mit den Pfeil-Buttons lassen sich Benutzungs-Abhängigkeiten zwischen Klassen darstellen. Der gestrichelte Pfeil steht hierbei für eine „uses“-Verbindung (im Beispiel benutzen Objekte der Klassen Kreis, Rechteck, Ellipse und Dreieck ein Objekt der Klasse Canvas (Leinwand) zum Zeichnen).

Mithilfe des durchgezogenen Pfeils mit der nicht ausgefüllten Dreiecksspitze werden Vererbungen dargestellt. Er steht für eine „is a“-Verbindung.

Vor der Erzeugung eines neuen Objektes muss der Quelltext compiliert werden („Compile“). Beim Klick auf die Schaltfläche wird das gesamte Projekt compiliert.

Bei Verwendung von BlueJ arbeitet man mit **Projekten**. BlueJ Projekte sind Verzeichnisse – wie normale Java-Packages auch. Alle Dateien, die zu dem Projekt gehören, sind in diesem Verzeichnis enthalten. Es sind dies die Java-Klassen (*.java und *.class) des Projekts und Einstellungsdateien.
Das Öffnen eines Projektes/Pakets erfolgt durch Wahl von „Open Project…“ im Menü Project.

Menüaufbau

Die Befehle im Menü **Projekt** (Project):

Befehl	**Erläuterung**
Neues Projekt… New Project…	erstellt ein neues Projekt.
Projekt öffnen… Open Project…	öffnet ein vorhandenes BlueJ-Projekt.
Letzte Projekte… Open Recent…	zeigt eine Liste mit den zuletzt bearbeiteten Projekten zum direkten Öffnen an.
Fremdprojekt öffnen… Open NonBlueJ…	öffnet ein Projekt, das nicht mit BlueJ erstellt wurde.
Schließen Close	schließt das aktuelle Projekt.
Speichern Save	speichert das aktuelle Projekt.
Speichern unter… Save As…	speichert das aktuelle Projekt unter einem neuen Namen.
(Importieren… Import…)	importiert vorhandene Klassen in das aktuelle Projekt.
Als jar-Archiv speichern… Create Jar File…	erstellt eine Anwendung, die sich auch ohne BlueJ öffnen lässt.
Seite einrichten… Page Setup…	richtet die Seite für einen Ausdruck ein.
Drucken… Print…	druckt Klassendiagramm oder Quelltext aus.
Beenden Quit	beendet BlueJ.

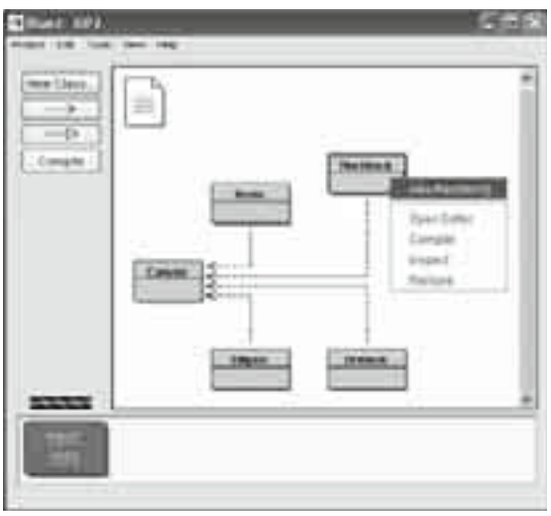
Die Befehle im Menü **Bearbeiten** (Edit):

Befehl	**Erläuterung**
Neue Klasse… New Class…	erstellt eine neue Klasse innerhalb des aktuellen Projektes, entspricht der Schaltfläche „New Class ...“ („Neue Klasse ...“).
Neues Paket… New Package…	erstellt ein neues Package.
Klasse aus Datei hinzufügen… Add Class from File...	fügt eine Klasse aus einer anderen Datei dem aktuellen Projekt hinzu.
Entfernen Remove	löscht die gerade ausgewählte Klasse oder Beziehung.

Neuer Verwendungspfeil New Uses Arrow	fügt einen neuen Verwendungspfeil ein, entspricht der zweiten Schaltfläche im Klassenfenster.
Neuer Vererbungspfeil New Inheritance Arrow	fügt einen neuen Vererbungspfeil ein, entspricht der dritten Schaltfläche im Klassenfenster.

Die wichtigsten Befehle im Menü **Werkzeuge** (Tools):

Befehl	**Erläuterung**
Übersetzen Compile	kompiliert das gesamte Projekt, entspricht der untersten Schaltfläche.
Auswahl übersetzen Compile Selected	kompiliert die ausgewählten Klassen.
Paket neu übersetzen Rebuild	Ganzes Paket neu übersetzen: Es wird jede Klasse neu compiliert, auch wenn sie seit der letzen Übersetzung nicht geändert wurde.
Dokumentation erzeugen Project Documentation	erzeugt die Dokumentation für die verwendeten Klassen (JavaDoc).
Einstellungen… Preferences…	ändert die Einstellungen der Entwicklungsumgebung.



Erzeugen neuer Objekte

Ein neues Objekt wird erzeugt, indem mit der rechten Maustaste die gewünschte Klasse ausgewählt und der Konstruktor über „`new …()`“ aufgerufen wird. Dem neuen Objekt muss hierbei ein eindeutiger Bezeichner zugewiesen werden. Erzeugte Objekte werden im Objektfenster angezeigt.

Aufruf von Methoden

Erzeugte Objekte können mittels Methodenaufrufen direkt manipuliert werden. Nachdem ein Objekt erstellt wurde, kann man seine öffentlichen Methoden ausführen. Dazu klickt man mit der rechten Maustaste auf das Objekt und ein Popupmenü mit Methodennamen (Methodensignaturen) öffnet sich. Das Menü zeigt sämtliche Methoden, die für das gewählte Objekt verfügbar sind, sowie zwei weitere Menübefehle, die durch die BlueJ-Umgebung hinzugefügt werden: *Inspect* ruft den Objektinspektor auf (siehe nächsten Punkt), *Remove* entfernt das Objekt.
Von Oberklassen geerbte Methoden lassen sich über *inherited from…* anzeigen und aufrufen.