HANSER

Leseprobe

zu

„Programmieren trainieren“

von Luigi Lo Iacono et al.

Print-ISBN: 978-3-446-45911-3
E-Book-ISBN: 978-3-446-46057-7
E-Pub-ISBN: 978-3-446-46494-0

Weitere Informationen und Bestellungen unter
http://www.hanser-fachbuch.de/978-3-446-45911-3
sowie im Buchhandel

Inhalt

Vorwort

„Jede hinreichend fortschrittliche Technologie ist von Magie nicht zu unterscheiden."
Arthur C. Clarke

Wir sind Programmierer. Wir sind Magier. Das MIT ist unser Hogwarts, der Google Campus ist unsere Unsichtbare Universität, Cupertino ist unser Narnia. Steve Jobs ist unser David Copperfield, Larry Page und Sergey Brin sind unsere Ehrlich Brothers und Frank Thelen ist immerhin vielleicht noch sowas wie unser Vincent Raven. Wir sind Siegfried und Roy und aus Versehen programmierte Endlosschleifen sind unsere weißen Tig. . . strapazieren wir die Allegorie mal nicht über. Jedenfalls: Wir sind Magier.

Oder wenigstens wirken wir für unser Umfeld so. Der Onkel dritten Grades, der im Atomkraftwerk arbeitet, würde uns selbst dann anrufen, wenn AssetWise mal hakt, weil wir eben Informatiker sind und uns dementsprechend mit allem auskennen, was irgendwie mit Computern zusammenhängt. Dabei ist es auch egal, wie komplex oder unterkomplex die Aufgabe ist. Wir werden angerufen, wenn ein Teilchenbeschleuniger angesteuert werden muss, aber auch, wenn es im Fachgeschäft für Strickzubehör „Woll im Leben" einer Freundin des Freundes deiner Tante väterlicherseits in der alten Fußgängerzone der Kleinstadt, in der du geboren wurdest, nicht mehr ganz so gut läuft und sie jetzt „mal eben" einen Shop braucht, um den Globalisierungseffekt besser für sich zu nutzen und das Wollgeschäft im Sturm zu erobern. Gestrickt wird ja wohl überall und sie ist sogar bereit, dir für die 3 Wochen Arbeit noch 100 € in die Hand zu drücken. Dafür müsstest du dann aber auch die nächsten 3 Jahre zu jeder Tages- und Nachtzeit für Support zur Verfügung stehen.

Doch wir steuern nicht nur die Stromversorgung und das weltweite Woll-Business. Wir halten Banken am Laufen, das Transportwesen und die Kommunikation, ohne uns läuft gar nichts mehr heutzutage. Wir können Welten erschaffen und wir können sie auch zerstören, je nachdem, ob wir Harry Potter oder Lord Voldemort sein wollen.

Welchen Weg du einschlagen willst – und jeder, der schon mal programmiert hat und behauptet, niemals auf die dunkle Seite geschaut zu haben, lügt – entscheidest du selbst und auf dem Weg zu deiner Entscheidung ist dieses Buch dein Hogwarts Express und du musst nicht mal gegen eine Mauer rennen, um hinein zu gelangen. Du hast die erste Seite aufgeschlagen und das Vorwort gelesen und die Richtung, in der du weiterblätterst, ist deine rote und deine blaue Pille.

Schlag es wieder zu – dann endet die Geschichte hier, du wachst auf in deinem Bett und glaubst, was immer du glauben möchtest. Blätter weiter, bleib im Wunderland und das Buch zeigt dir, wie tief der Kaninchenbau geht. Nerd today, boss tomorrow.

Patrick Stenzel (@rock_galore), im Januar 2020

Vorwort zur ersten Auflage

Nerds sind in. Diese liebenswerten Zeitgenossen mit dem vielen Spezialwissen und den kindlichen Vorlieben für Superhelden werden lange nicht mehr nur komisch beäugt. Im Gegenteil. Sie selbst sind nunmehr Stars in vielen Fernsehserien, und ihr modischer Stil ist allgemein akzeptiert. Diese Entwicklung kommt auch der Programmierung zugute. Lange Zeit galt diese Fertigkeit als ein Gebiet, das den Nerds vorbehalten ist. Dem ist nicht so! Es muss nur der Mut aufgebracht werden, sich damit auseinanderzusetzen. Dann wird schnell klar, was mit der Programmierung alles umgesetzt werden kann. Die Bandbreite ist groß und wird durch aktuelle Trends stetig befeuert. Insbesondere durch die Digitalisierung und Vernetzung vieler Alltagsgegenstände finden sich Softwareprogramme vermehrt jenseits gängiger Anwendungsfälle im betrieblichen Kontext von Unternehmen wieder. Also, keine Scheu und ran ans Programmieren!

Mir selbst bereitet das Programmieren viel Freude. Zudem ist es mir eine Herzensangelegenheit, mein Programmier-Knowhow und meine Erfahrung an andere weiterzugeben. Ich weiß aus vielen Schulungen sehr genau, was es für Hürden und Stolpersteine beim Programmieren lernen gibt und wie diesen zu begegnen ist. **Gutem Trainingsmaterial kommt dabei eine zentrale Rolle zu**.

Die Autoren Lo Iacono, Wiefling und Schneider schließen hier eine wichtige Lücke. Sie versorgen dich mit vielen Trainingsaufgaben, die dir helfen werden, die wesentlichen Programmierkonzepte wirklich zu verstehen. Und mehr noch. Du kannst und solltest so lange mit den vielen Aufgaben trainieren, bis der Groschen tatsächlich gefallen ist. Das ist wichtig. Denn erst dann wirst Du in der Lage sein, mit dem erlernten Handwerkszeug auch selbstständig Entwicklungsaufgaben bewältigen und lösen zu können. Genau da sollst du hin. Viele Lehrformate gehen hier nicht weit genug. Die falsche Annahme ist dabei häufig, dass ein Beispiel und eine Übungsaufgabe zum Verständnis ausreichen. Weit gefehlt. Es fängt schon damit an, dass nicht jeder mit dem gegebenen Beispiel oder der gestellten Übungsaufgabe etwas anfangen kann. Hier schafft das vorliegende Buch Abhilfe, und es gehört damit in die „*Einstieg in die Programmierung*“-Ecke deines Bücherregals.

Dirk Louis, Januar 2018

1 Einleitung

1.1 Wozu sollte ich programmieren lernen?

Weil Du es kannst und weil die Programmierung **das Werkzeug des 21. Jahrhunderts** ist. Die Bundeskanzlerin Frau Angela Merkel hat in einem Interview mit YouTubern das Programmieren auf eine Stufe mit den Grundfertigkeiten Lesen, Schreiben und Rechnen gestellt (https://youtu.be/Uq2zIzscPgY?t=12m18s). Programmieren ist lange nicht mehr nur etwas für Experten, die das studiert haben. Durch den Einzug des Digitalen in alle Branchen und den Alltag können viele andere als nur Informatiker von der Programmierung profitieren und damit ihre Ideen erproben und verwirklichen. Beispiele kannst du unzählige finden. Lass' uns hier nur einige zur Verdeutlichung kurz anreißen. Dir fallen dann bestimmt selbst viele weitere Beispiele ein.

Angenommen, du bist **Künstler** und hast bisher mit den klassischen Materialien und Techniken deiner Disziplin gearbeitet. Für deine neueste Projektidee möchtest du mit regelmäßigen Formen und Farben experimentieren, wie es z. B. Sol LeWitt in seinem künstlerischen Schaffen getan hat (https://de.wikipedia.org/wiki/Sol_LeWitt). Das erfordert viel Fleiß, Geduld und Präzision. Da du deine Zeit lieber damit verbringen möchtest, an spannenden neuen Konstruktionen und deren Wirkung zu experimentieren, anstatt diese in langwierigen und teils monotonen Arbeitsschritten erst erstellen zu müssen, wünschst du dir einen Automatismus dafür, der das für dich erledigt. Dies kann ein eigens geschriebenes Computerprogramm leisten. Ist ein solches geschrieben, liegen die Vorteile auf der Hand. Veränderungen an den Farben, der Größe sowie Anordnungen der Formen usw. sind umgehend gemacht. Auch das Ausgabeformat kann leicht angepasst werden, um das Kunstwerk in vielfältiger Art und Weise zu drucken oder aus einem Rohling zu fräsen. Pioniere der computergenerierten Kunst sind z. B. Manfred Mohr, Joseph Nechvatal, Olga Kisseleva und John Lansdown.

Als **Veranstaltungstechniker** sieht man sich heute immer stärker mit Anforderungen von Kunden konfrontiert, die nach noch nicht dagewesenen Hinguckern verlangen. Hierfür gibt es naturgemäß keine fertigen Lösungen, die man aus dem Regal ziehen kann. Somit siehst du dich auf der einen Seite immer mit neuen spannenden Entwicklungsaufgaben konfrontiert, musst dafür aber auf der anderen Seite adäquate Lösungen entwickeln. Diese bedingen eigentlich immer auch Software, die es zu programmieren gilt.

Im letzten fiktiven Szenario wollen wir ins **Internet der Dinge** abtauchen. Mit diesem Schlagwort wird der allgemeine Trend bezeichnet, mit dem die Digitalisierung und die Vernetzung

im Gewand des Internets stetig in Gegenstände des alltäglichen Gebrauchs diffundieren. Der smart gewordene Fernseher ist ein Paradebeispiel hierfür. Einige neue Anwendungen findest du toll, willst aber noch nicht in neue Produkte investieren. Die alten tun es ja noch. So findest du es z. B. praktisch, im Supermarkt einen Blick in deinen Kühlschrank werfen zu können, um zu sehen, ob es genügend Frühstückseier fürs Wochenende gibt. Der Kühlschrank ist schnell für diesen Anwendungsfall erweitert. Mit einer batteriebetriebenen Kamera, einem LED-Licht und etwas Programmierung kannst du bald via Smartphone-App in deinen Kühlschrank gucken.

Das soll zeigen, was dir alles an Möglichkeiten offen steht, wenn du die Programmierung als ein Werkzeug verstehst und dich dessen bemächtigst.

1.2 Wie kann mir dieses Buch dabei helfen?

Vor den Erfolg haben die Götter allerdings den Schweiß gesetzt. Diese Tatsache hat der griechische Dichter und Geschichtsschreiber *Hesiod* bereits vor langer Zeit festgestellt und dann so zutreffend formuliert. Dieser Ausspruch trifft unseres Erachtens kaum besser auf etwas zu als die Programmierung. Es gehört eine ordentliche Portion Arbeit dazu, bis der Groschen fällt und man die wesentlichen Programmierkonzepte verstanden hat. Erst dann wird man in der Lage sein, Aufgabenstellungen jeglicher Couleur selbstständig angehen und erfolgreich bewältigen zu können. Wir wollen dir diese notwendigen Mühen nicht verschweigen. Unserer Erfahrung nach scheitert so mancher Einstieg genau an dieser Hürde.

Unser Ansatz ist deshalb, durch **viele spannende Programmieraufgaben** das nötige Material zum Trainieren bereitzustellen. Du wirst in diesem Buch im Wesentlichen Aufgabenstellungen von uns bekommen, an denen du Aufgabe für Aufgabe alle relevanten Programmierkonzepte üben kannst. Damit dir dabei nicht die Laune vergeht, haben wir uns viel Mühe beim Zusammenstellen der Aufgaben gegeben. Es wird deutlich über die meisten „Lehrbuchaufgaben“ hinausgehen und sich, soweit möglich, erheblich näher an praktischen Anwendungsszenarien orientieren. Der klassische Lehrbuchstil hangelt sich meist an Aufgabenstellungen aus der Mathematik entlang. Das ist wichtig, und daher haben auch wir das ab und an mit dir vor. Wir können aber auch verstehen, wenn derartige Aufgaben nicht jedem liegen, um etwas Neues zu lernen. Daher programmierst du eher Anwendungen, die etwas Nützliches tun oder etwas hübsch Anzuschauendes generieren. Als Appetitanreger haben wir im nachfolgenden Bild 1.1 schon mal drei Beispiele aus dem Buch für dich.

Diese drei Bilder zeigen exemplarisch, was du mit unserem Trainingsprogramm programmieren sollst und auch können wirst, wenn du fleißig am Ball bleibst. Es lohnt sich!

1.3 Was muss ich mitbringen?

Nicht viel! Mit diesem Buch können wir es nicht leisten, dir die Basics beizubringen. Das musst du halt selbst tun oder du bekommst es in irgendeiner Form gezeigt. Wir erklären zu

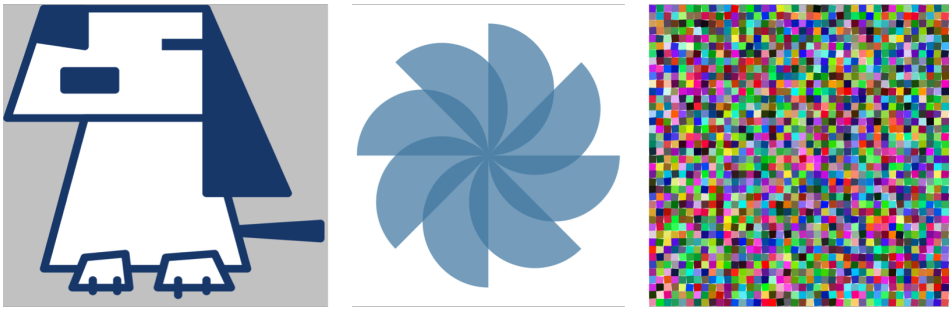


Bild 1.1 Drei Beispielbilder, die du programmieren wirst

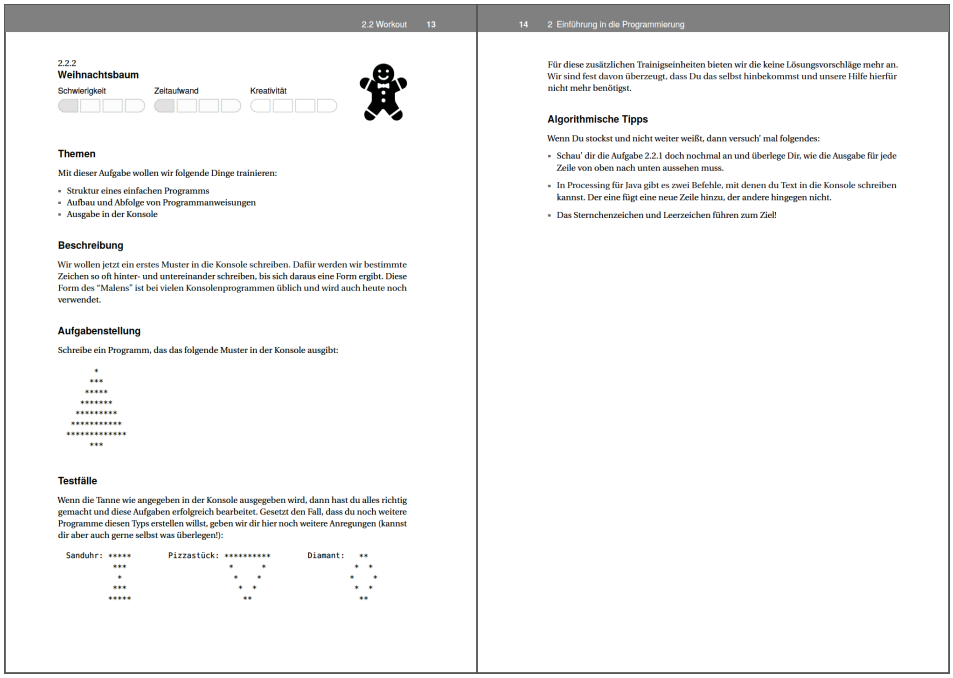
Beginn eines jeden Kapitels nochmal kurz die im Fokus stehenden Übungsschwerpunkte. Das ist mehr eine Gedächtnisstütze und soll als Warm-up dienen, falls du es überhaupt brauchst. Wir gehen dabei nochmals kurz auf die wesentlichen Konzepte ein und erläutern Besonderheiten in den Programmierumgebungen, für die wir Beispiellösungen bereitstellen.

1.4 Wie geht das vonstatten?

Wie schon gesagt, ist das hier ein Trainingsbuch fürs Programmieren. Wir stellen dir über **120 Übungsaufgaben mit Lösungsvorschlägen** zum Training bereit. Die grundlegende Struktur gleicht dabei derer gängiger Ressourcen zur Einführung in die Programmierung. Es geht mit dem Aufbau erster einfacher Programme los und wird durch das Hinzukommen von Programmierkonzepten wie Variablen, Datentypen, Operatoren, Ausdrücke, bedingte Anweisungen, Wiederholungsanweisungen, Funktionen, Arrays, Strings bis hin zur Objektorientierung stetig erweitert. Damit wir uns – ohne unnötiges Störfeuer und Ablenkung – auf das Kernthema des jeweiligen Kapitels konzentrieren können, bestehen die ersten Programme der Kapitel 2, 3 und 4 zunächst aus Anweisungen. Erst in den nachfolgenden Kapiteln 5 bis 9 kommen dann Strukturelemente für den Programmcode in Form von Funktionen sowie Klassen und Objekten hinzu. Wir werden dir Kapitel für Kapitel Trainingsaufgaben stellen, für die du dir ein passendes Programm überlegen und dieses dann in einer Programmiersprache vollständig angeben sollst.

Jede Trainingsaufgabe ist nach einem **festen Schema** aufgebaut (siehe Bild 1.2). Um jede Aufgabe eindeutig identifizieren zu können, haben wir diese mit einem eindeutigen Namen, einer eindeutigen Nummer und einem Piktogramm versehen. Das wird dir insbesondere dabei helfen, dich mit Freunden, Mitschülern, Kommilitonen, Kollegen oder der Community über die Aufgaben auszutauschen. Auch unsere Lösungsvorschläge im Anhang und online wirst du auf diese Weise spielend der jeweiligen Aufgabe zuordnen können. Die Identifizierungsnummer ist dem jeweiligen Buchkapitel zugeordnet. Das einfache Zurückspringen zur Aufgabenstellung im Buch ist damit auch gewährleistet.

Die Aufgaben haben wir unserem Dafürhalten nach in den Kategorien **Schwierigkeit**, **Kreativität** und **Zeitaufwand** bewertet und sortiert. Einfachere Aufgaben, die im Vergleich eher wenig Zeit und kreative Eigenleistung erfordern, findest du eher am Anfang eines jeden Kapitels. Du bist aber völlig frei in deiner Entscheidung, welcher Aufgabe du dich in welcher



2.2 Workout 13

2.2.2
Weihnachtsbaum

Schwierigkeit Zeitaufwand Kreativität

Themen

Mit dieser Aufgabe wollen wir folgende Dinge trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe in der Konsole

Beschreibung

Wir wollen jetzt ein erstes Muster in die Konsole schreiben. Dafür werden wir bestimmte Zeichen so oft hinter- und untereinander schreiben, bis sich daraus eine Form ergibt. Diese Form des "Malens" ist bei vielen Konsolenprogrammen üblich und wird auch heute noch verwendet.

Aufgabenstellung

Schreibe ein Programm, das das folgende Muster in der Konsole ausgibt:

```
    *
   ***
  *****
 *******
*********
   ***
```

Testfälle

Wenn die Tanne wie angegeben in der Konsole ausgegeben wird, dann hast du alles richtig gemacht und diese Aufgaben erfolgreich bearbeitet. Gesetzt den Fall, dass du noch weitere Programme diesen Typs erstellen willst, geben wir dir hier noch weitere Anregungen (kannst dir aber auch gerne selbst was überlegen!):

```
Sanduhr: *****       Pizzastück: **********       Diamant:   **
          ***                    *      *                  *  *
           *                    *    *                   *    *
          ***                    *  *                     *  *
         *****                    **                       **
```

14 2 Einführung in die Programmierung

Für diese zusätzlichen Trainigseinheiten bieten wir die keine Lösungsvorschläge mehr an. Wir sind fest davon überzeugt, dass Du das selbst hinbekommst und unsere Hilfe hierfür nicht mehr benötigst.

Algorithmische Tipps

Wenn Du stockst und nicht weiter weißt, dann versuch' mal folgendes:

- Schau' dir die Aufgabe 2.2.1 doch nochmal an und überlege Dir, wie die Ausgabe für jede Zeile von oben nach unten aussehen muss.
- In Processing für Java gibt es zwei Befehle, mit denen du Text in die Konsole schreiben kannst. Der eine fügt eine neue Zeile hinzu, der andere hingegen nicht.
- Das Sternchenzeichen und Leerzeichen führen zum Ziel!

Bild 1.2 Exemplarische Darstellung des Aufgaben-Templates

Reihenfolge widmen möchtest. Du musst auch längst nicht alle Aufgaben durchackern. Wenn der Groschen in Bezug auf die in einem Kapitel fokussierten Programmierkonzepte gefallen ist, kann es ans nächste Kapitel gehen.

Wir haben schon eingangs gesagt, dass wir mit dir und den Aufgaben eines jeden Kapitels bestimmte Programmierkonzepte trainieren möchten. Auf welche **Themen** sich eine jeweilige Aufgabe dabei im Besonderen fokussiert, geben wir dir kurz stichwortartig an. Dies soll ein zusätzliches Kriterium sein, wonach du entscheiden kannst, ob du eine bestimmte Aufgabe bearbeiten möchtest.

Worum es sich in einer Aufgabe dreht, erläutert eine kurze **Beschreibung**. Hier wird der Kontext gesetzt und ggf. auch ein Praxisbezug hergestellt, damit du weißt, wo sich in der Praxis etwas Derartiges finden lässt. Was du dann tatsächlich tun musst, geben wir dir im anschließenden Abschnitt mit dem passenden Namen **Aufgabenstellung** an. Du solltest beide Abschnitte aufmerksam und eventuell mehrmals lesen, um sicherzustellen, dass du deinen Auftrag richtig verstanden hast.

Zur Überprüfung, ob dein Programm tatsächlich funktioniert, geben wir dir **Testfälle** mit entsprechenden Testdaten an die Hand. Zudem haben wir zu jeder Aufgabe eine ausführlich kommentierte Lösung in den Programmiersprachen Java und Python beigefügt. Bevor du dir die aber anguckst, solltest du wirklich lange Zeit selbst an der Erarbeitung einer Lösung werkeln.

Wenn du völlig auf dem Schlauch stehst und gar keinen Zugang findest, sind am Ende einer jeden Aufgabe **algorithmische Tipps** aufgelistet, die dich einem möglichen Lösungsansatz näherbringen sollen. Bitte nutze diese Tipps und versuche unbedingt, selbst eine Lösung

herzustellen, bevor du dir unsere Lösungsvorschläge im Anhang des Buchs anschaust. Nur auf diese Weise kommst du genügend ins Schwitzen, um nachher wirklich behaupten zu können, das Programmieren auch selbstständig zu beherrschen.

1.5 Was muss ich sonst noch wissen?

Damit du möglichst effektiv und fokussiert trainieren kannst, haben wir die Trainingsumgebung für dich von unnötigem Ballast entschlackt. Du sollst nicht schon bei der Installation, Konfiguration und Verwendung der Programmierumgebung die Lust am Programmieren bzw. die Sicht auf das Wesentliche verlieren. Wir stützen uns daher auf ein einziges Werkzeug, mit dem du in **Java und Python** das Programmieren trainieren kannst. Es handelt sich dabei um das frei und kostenlos verfügbare Tool mit dem Namen **Processing**, das du unter der Web-Adresse https://processing.org/ abrufen kannst. Hier findest du auch viele weitere Informationen und Dokumentationen rund um Processing. Im Anhang C haben wir Installationsanleitungen für die gängigen Betriebssysteme Windows, macOS und Linux beigefügt.

Dass wir auf Processing abstellen, soll aber nicht heißen, dass du mit dem Erlernten in der Praxis nicht viel anfangen kannst. Ganz im Gegenteil! Die Programmiersprachen Java und Python gehören zu den am weitesten verbreiteten Sprachen, und schließlich kommt es im Wesentlichen auf die Programmierkonzepte an. Wenn du diese intensiv trainiert und dadurch verinnerlicht hast, dann bist du bereit, alle möglichen Aufgabenstellungen programmatisch zu lösen. Dann haben wir unser gemeinsames Ziel erreicht. Die Verwendung professionellerer Programmierumgebungen wie z. B. Eclipse, IntelliJ oder PyCharm ist dann ein Klacks. Darüber müssen wir dann nicht mehr viele Worte verlieren. Wenn Du so gar keine Lust auf Processing haben solltest und unsere Aufgaben lieber mit anderen Programmierumgebungen für Java oder Python bearbeiten möchtest, dann findest Du im Anhang D Anleitungen für den Aus- und Umstieg.

Die Quelltexte zum Buch – unsere Lösungsvorschläge – haben wir auf dem Onlinedienst **GitHub** für dich bereitgelegt. Du findest sie unter der Adresse https://github.com/protrain. Wie du damit umgehst, erklären wir für alle gängigen Desktop-Betriebssysteme im Anhang A.1 (für Java) bzw. im Anhang B.1 (für Python).

Mit dem Kauf des Buches soll aber noch nicht Schluss sein. Wir würden uns sehr freuen, von dir zu hören. Über GitHub kannst du uns z. B. auf Fehler im Buch oder in den Lösungen aufmerksam machen. Wir tragen das dann in die Errata-Liste ein bzw. korrigieren die Programme. Außerdem kannst du uns auch deine Lösung(en) bereitstellen. Wenn diese einen eigenen Lösungsweg zeigen, nehmen wir sie mit in das Repositorie auf. Selbiges gilt für Lösungen in anderen Programmiersprachen. Achte aber bitte hierbei darauf, dass es für die Sprache eine ähnliche einfache und umfangreiche Programmierumgebung gibt, wie es Processing für Java und Python ist. Wenn du sonstige Anregungen zur Verbesserung hast oder Ideen für weitere Aufgaben beisteuern möchtest, freuen wir uns von dir zu hören.

Hoffentlich konnten wir dein Interesse wecken und dir unseren Ansatz zum Programmieren lernen schmackhaft machen. Jedenfalls würden wir uns sehr freuen, gemeinsam mit dir das Programmieren zu trainieren.

2 Einführung in die Programmierung

2.1 Warmup

Dein Training beginnt in diesem Kapitel mit ersten einfachen Programmen. Dazu musst du wissen, wie der grundlegende Aufbau eines Programms sowie der Aufbau der Anweisungen in einer bestimmten Programmiersprache ist. Letzteres gehört zur sogenannten **Syntax** einer Programmiersprache. So wie z. B. die Syntax einer natürlichen Sprache Prinzipien und Regeln des Wort- und Satzbaus festlegt, so legt die Syntax einer Programmiersprache das Vokabular und den Aufbau von Anweisungen fest.

Für die allerersten Programme, die du entwickeln sollst, genügt zunächst die allereinfachste Struktur überhaupt. Hierbei werden Programme als eine lineare Abfolge von Anweisungen angegeben. Anweisungen verfügen immer über einen Namen und eine Liste von Parametern, die die Anweisung verarbeiten soll. Um den Anweisungsnamen von der Parameterliste unterscheiden zu können, werden die Parameter häufig eingeklammert und dem Anweisungsnamen nachgestellt.

```
nameAnweisung(parameter);
```

Verfügt die Parameterliste über mehrere Einträge, so werden diese mit Komma (`,`) voneinander getrennt.

```
nameAnweisung(parameter1, parameter2);
```

Parameterlose Anweisungen sind durch ein leeres Klammernpaar gekennzeichnet.

```
nameAnweisung();
```

Um mehrere Anweisungen voneinander unterscheiden zu können, wird dafür ein Trennzeichen in der Syntax einer Programmiersprache festgelegt. In Java ist das das Semikolon (`;`). Das folgende Beispiel zeigt ein abstraktes Programm, das sich aus sieben Anweisungen zusammensetzt, die in der angegebenen Reihenfolge ausgeführt werden. Die lineare Programmabfolge führt die programmierten Anweisungen zeilenweise von links nach rechts beginnend mit der obersten Zeile aus.

```
Anweisung1(); Anweisung2(); Anweisung3(); Anweisung4(); Anweisung5();
Anweisung6(); Anweisung7();
```

Durch diese Syntaxregel können die einzelnen Anweisungen separiert werden, unabhängig davon, wie du diese in die Quelltextdatei schreibst. Zur besseren Lesbarkeit empfehlen wir dir aber, dich auf eine Anweisung pro Zeile zu beschränken und die Anweisungen untereinander zu schreiben.

```
Anweisung1();
Anweisung2();
Anweisung3();
Anweisung4();
Anweisung5();
Anweisung6();
Anweisung7();
```

Die Programmiersprache Python legt in ihrer Syntax als Trennzeichen von Anweisungen den Zeilenumbruch fest. Ein Zeilenumbruch kann je nach Betriebssystem aus einem oder zwei Zeichen bestehen (`'\n'`, `'\r'` oder `'\r\n'`).

In der Programmierliteratur hat sich das `"Hello World!"`-Programm als einführendes Beispiel zur Darstellung der grundlegenden Syntax eines einfachen Programms in einer bestimmten Programmiersprache etabliert. Das Hello-World-Programm gibt in der Konsole einen einfachen Text aus, nämlich `Hello World!`. Wir wollen es zur Konkretisierung der einführenden Erläuterungen verwenden.

Java:

```
print("Hallo Welt!");
```

Python:

```
print("Hallo Welt!")
```

Die `print()`-Anweisung bekommt einen Parameter übergeben. Dieser enthält den Text, den die Anweisung in der Konsole ausgeben soll. Um den Text eingrenzen zu können, wird dieser von doppelten Anführungszeichen (`"`) eingerahmt.

Die Aufgaben dieses Kapitels drehen sich um derartige Programme. Deine Aufgabe wird es sein, die zur Lösung der Aufgabenstellung benötigten Anweisungen zu identifizieren und diese dann in einer geeigneten Abfolge zu platzieren. Welche Anweisungen eine Programmiersprache im Standardumfang bereitstellt, sind in der Referenzdokumentation aufgeführt. Die Referenz der von Processing bereitgestellten Anweisungen kann im Internet eingesehen werden:

- https://processing.org/reference/ (Java)
- http://py.processing.org/reference/ (Python)

Referenzen sind sehr umfangreich. Dies gilt auch für die von Processing. Es kann daher etwas dauern, bist du dich darin zurechtfindest. Für die in diesem Kapitel bereitgestellten Trainingsaufgaben sind insbesondere Funktionen zur Ausgabe von Texten in der Konsole und Funktionen zur Ausgabe elementarer geometrischer Formen im grafischen Ausgabefenster wichtig. Um dir das Auffinden dieser Anweisungen zu erleichtern, führen wir dir in der nachfolgenden Auflistung die relevanten auf.

- Konsolenausgabe
  - https://processing.org/reference/print_.html (Java)
  - http://py.processing.org/reference/print.html (Python)
- Linie
  - https://processing.org/reference/line_.html (Java)
  - http://py.processing.org/reference/line.html (Python)

- Dreieck
  - https://processing.org/reference/triangle_.html (Java)
  - http://py.processing.org/reference/triangle.html (Python)
- Rechteck
  - https://processing.org/reference/rect_.html (Java)
  - http://py.processing.org/reference/rect.html (Python)
- Viereck
  - https://processing.org/reference/quad_.html (Java)
  - http://py.processing.org/reference/quad.html (Python)
- Ellipse
  - https://processing.org/reference/ellipse_.html (Java)
  - http://py.processing.org/reference/ellipse.html (Python)
- Kreisausschnitt
  - https://processing.org/reference/arc_.html (Java)
  - http://py.processing.org/reference/arc.html (Python)

Um sich mit der Funktionsweise der Anweisungen vertraut zu machen, empfehlen wir dir, die Beschreibung in der Referenz aufmerksam zu lesen. Dies ist eine wichtige Grundfertigkeit, die zum Programmieren dazugehört.

Verwenden werden wir in diesem Buch die Entwicklungsumgebung Processing. Hiermit können wir Programme sowohl in Java als auch in Python schreiben. Processing bietet nicht nur den Vorteil der einfachen Installation auf nahezu allen Betriebssystemen. Wir können damit auch sehr einfach (grafische) Programme auf Basis von Anweisungen schreiben. Aber auch höherwertige Konzepte, wie wir sie in den späteren Kapiteln umsetzen werden, sind in Processing möglich. Perfekte Voraussetzungen also zum Trainieren deiner Programmiertechniken mit diesem Buch.

Alle Installationsschritte von Processing findest du in Anhang C.1. Wie du an die digitalen Quelltexte unserer Lösungsvorschläge zu einzelnen Aufgaben kommst und wie du sie in Processing öffnest, steht im Anhang A.1.1 für Java und im Anhang B.1.1 für Python.

Dateien mit Quelltext können wir in Processing mit Klick auf *Datei → Öffnen …* laden. In Bild 2.1 haben wir zum Beispiel eine solche Datei geöffnet. Dort können wir gut die grafische Benutzeroberfläche von Processing erkennen:

- Mit dem Start- und Stopp-Button *(1)* kannst du deinen Java- bzw. Python-Code ausführen.
- Um vom Java- auf den Python-Modus zu wechseln, kannst du den Modus-Auswahlreiter *(2)* verwenden. Wie du den Python-Modus in Processing installierst, steht in Anhang C.5. Links neben diesem Button ist der integrierte Debugger, den du zur Analyse von Java-Code verwenden kannst. Mehr dazu findest du in Anhang A.1.3.
- In der Mitte der Benutzeroberfläche *(3)* steht der eigentliche Quelltext. In diesen Bereich kannst du deinen Java- bzw. Python-Code hineinschreiben.
- Entsprechende Ausgaben in der Konsole findest du im darunterliegenden Bereich *(4)*. Hier werden auch auftretende Fehler im Code angezeigt, sofern es welche gibt.

Nach der Einrichtung von Processing und dem Lesen der Einführung solltest du für dieses Kapitel ausgerüstet sein. In dem Sinne: Viel Spaß bei den ersten Aufgaben!

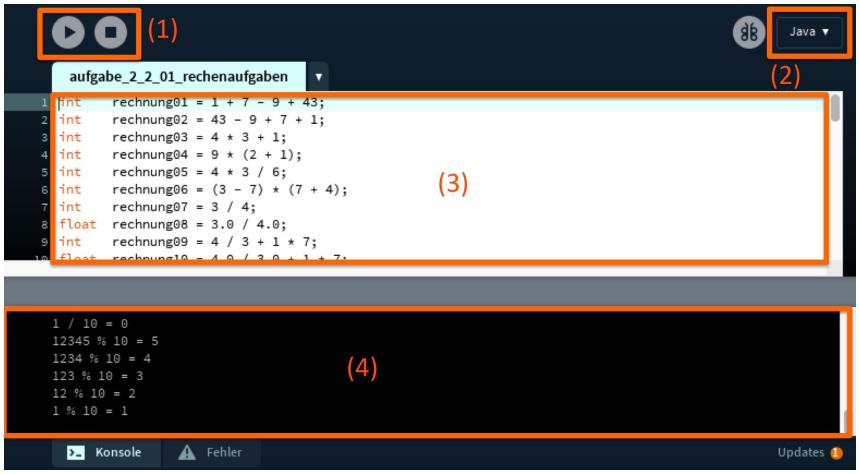
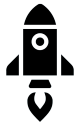


Bild 2.1 So sieht die grafische Benutzeroberfläche von Processing aus.

2.2 Workout

W.2.1 Three-Two-One – Mein erstes Programm



Schwierigkeit Zeitaufwand Kreativität

Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau von Programmanweisungen
- Ausgabe in der Konsole

Beschreibung

Wir wollen ein erstes Programm schreiben. Der Klassiker hierfür ist die Ausgabe eines Texts – meist der Text `Hello World` – in der Konsole. Dazu braucht es in der Regel nur eine einzige Anweisung. An dieser kannst du aber bereits den Aufbau von Anweisungen und einfachen Programmen nachvollziehen und trainieren. Los geht's!

Aufgabenstellung

Schreibe ein Programm, das den Text `Three-Two-One - Takeoff!` in der Konsole ausgibt. Wenn dein Programm funktioniert, solltest du den angegebenen Text in der Konsole lesen können, so wie nachfolgend exemplarisch zu sehen ist:

```
Three-Two-One - Takeoff!
```

Wenn das geklappt hat, dann mach' doch einfach weiter und modifiziere dein erstes Programm nach deinen Wünschen. Ändere z. B. den Text oder füge weitere Anweisungen zur Textausgabe hinzu. Reflektiere dabei, wie dein Programm auf die Änderungen reagiert. Wenn du das Resultat hast kommen sehen und es ist nichts Unerwartetes bei der Ausführung deines Programms passiert, hast du es im Griff und verstanden, wie Anweisungen und einfache Programme aufgebaut sind.

Testfälle

Zum Testen deines Programms brauchst du in diesem Fall noch keine Testdaten. Starte dein Programm und prüfe, ob die geforderte Ausgabe in der Konsole ausgegeben wird.

(Algorithmische) Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Gib' nicht auf. Du solltest es so lange probieren, bis es klappt. Das nennt man *Trial and Error* (Versuch und Irrtum). Versuch es weiter! Vermutlich bist du schon nah dran an der Lösung, denn der Fehler liegt sehr häufig im Detail.
- Wir benötigen eine passende Anweisung, die uns die Programmiersprache zur Ausgabe von Daten in der Konsole bereitstellt. Wie lautet diese?
- Anweisungen folgen einem festgelegten Aufbau. Hier schleichen sich schon mal Tippfehler ein. Was sagen denn die Fehlermeldungen, wenn du versuchst, dein Programm zu starten?

W.2.2 Weihnachtsbaum

Schwierigkeit Zeitaufwand Kreativität

Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe in der Konsole

Beschreibung

Wir wollen jetzt ein erstes Muster in die Konsole schreiben. Dafür werden wir bestimmte Zeichen so oft hinter- und untereinander schreiben, bis sich daraus eine Form ergibt. Diese Form des „Malens“ ist bei vielen Konsolenprogrammen üblich und wird auch heute noch verwendet.

Aufgabenstellung

Schreibe ein Programm, das das folgende Muster in der Konsole ausgibt:

```
      *
     ***
    *****
   *******
  *********
 ***********
*************
     ***
```

Testfälle

Wenn die Tanne wie angegeben in der Konsole ausgegeben wird, dann hast du alles richtig gemacht und diese Aufgaben erfolgreich bearbeitet. Gesetzt den Fall, dass du noch weitere Programme dieses Typs erstellen willst, geben wir dir hier noch weitere Anregungen (du kannst dir aber auch gerne selbst was überlegen!):

```
Sanduhr: *****        Pizzastück: **********        Diamant:   **
          ***                     *      *                   *  *
           *                       *    *                   *    *
          ***                       *  *                     *  *
         *****                        **                       **
```

Für diese zusätzlichen Trainingseinheiten bieten wir dir keine Lösungsvorschläge mehr an. Wir sind fest davon überzeugt, dass du das selbst hinbekommst und unsere Hilfe hierfür nicht mehr benötigst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Schau' dir die Aufgabe 2.2.1 doch noch einmal an und überlege dir, wie die Ausgabe für jede Zeile von oben nach unten aussehen muss.
- In Processing für Java gibt es zwei Befehle, mit denen du Text in die Konsole schreiben kannst. Der eine fügt eine neue Zeile hinzu, der andere hingegen nicht.
- Das Sternchen- und das Leerzeichen führen zum Ziel!

W.2.3 Perlenkette

Schwierigkeit Zeitaufwand Kreativität

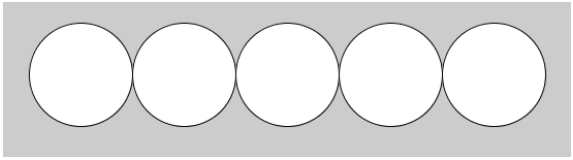
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir die unten dargestellte Perlenkette programmieren:



Die Kette besteht aus fünf Perlen, die als Kreise mit schwarzer Linie und weißer Füllung dargestellt sind.

Aufgabenstellung

Programmiere das angegebene Bild mithilfe der grafischen Grundelemente von Processing.

Testfälle

Wenn die geforderten Grundformen in Art, Größe, Farbe und Lage wie in der Aufgabenstellung gefordert gezeichnet werden, dann hast du eine Lösung gefunden und die Aufgabenstellung richtig gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Alle Processing-Befehle kannst du auf der offiziellen Homepage nachlesen (Java: https://processing.org/reference/, Python: http://py.processing.org/reference/). Hier kannst du nachschauen, um die entsprechenden Befehle für das Programm zu finden.
- In Processing gibt es keine Funktion zum Zeichnen von Kreisen. Aber es gibt eine Funktion zum Malen von Ellipsen. Ab wann wird eine Ellipse zum Kreis?
- Wenn du die Ausmaße des Bildschirmfensters weißt, wo wird wohl die Mitte des Bildschirmfensters liegen?

W.2.4 Die erste Zeichnung

Schwierigkeit

Zeitaufwand

Kreativität

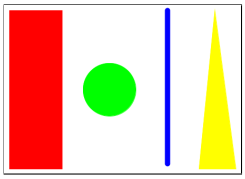
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir die Processing-Grundelemente besser kennenlernen. Dazu wollen wir folgende Grafik programmieren:



Die Grafik hat unter anderem folgende Eigenschaften:

- Fenstergröße: 450 Pixel breit und 320 Pixel hoch
- Rechteck:
  - x-Position: 10
  - y-Position: 10
  - Größe: 100 Pixel breit und 300 Pixel hoch
  - Farbe: rot
- Kreis:
  - x-Position: 200
  - y-Position: Mitte des Bildschirmfensters
  - Radius: 100 Pixel
  - Farbe: grün
- Linie:
  - Breite: 10
  - Start: 310 (x), 10 (y)
  - Ziel: 310 (x), 300 (y)
  - Farbe: blau
- Dreieck:
  - Eckpunkte:
    - 400 (x), 10 (y)
    - 370 (x), 310 (y)
    - 440 (x), 310 (y)
    - Farbe: gelb

Aufgabenstellung

Programmiere das angegebene Bild mithilfe der grafischen Grundelemente von Processing.

Testfälle

Wenn die geforderten Grundformen in Art, Größe, Farbe und Lage wie in der Aufgabenstellung gefordert gezeichnet werden, dann hast du die Lösung gefunden und umgesetzt.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Alle Processing-Befehle kannst du auf der offiziellen Homepage nachlesen (Java: https://processing.org/reference/, Python: http://py.processing.org/reference/). Hier kannst du nachschauen, um die entsprechenden Befehle für das Programm zu finden.
- In Processing gibt es keine Funktion zum Zeichnen von Kreisen. Aber es gibt eine Funktion zum Malen von Ellipsen. Ab wann wird eine Ellipse zum Kreis?
- Wenn du die Ausmaße des Bildschirmfensters weißt, wo wird wohl die Mitte des Bildschirmfensters liegen?

W.2.5 Raupe Allzeitappetit

Schwierigkeit

Zeitaufwand

Kreativität

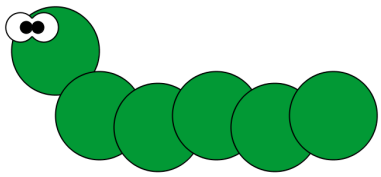
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir eine Raupe zeichnen:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Wenn deine Raupe grundsätzlich mit der abgebildeten Raupe übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Überlege dir zunächst, welche Grundelemente dieses Bild beinhaltet und wo diese platziert sind. Achte dabei auch auf eventuell „unsichtbare“ Grundelemente.
- Die Augen der Raupe bestehen entweder aus fünf (!) Kreisen oder zwei ganzen und zwei halben Kreisen. Beides ist möglich.
- Bei Ellipsen/Kreisen wird immer der Mittelpunkt angegeben und nicht die linke obere Ecke.

W.2.6 Klötzchen-Kunst

Schwierigkeit

Zeitaufwand

Kreativität

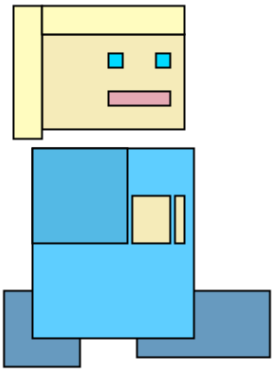
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir einen Menschen aus Rechtecken programmieren:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Wenn dein Klötzchen-Männeken grundsätzlich mit dem abgebildeten Menschen übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Schaue dir die vorherigen Zeichenaufgaben noch einmal an.
- Bei der Höhe und Breite des Rechtecks kannst du auch negative Werte angeben, um das Rechteck in die umgekehrte Richtung zu zeichnen.

W.2.7 Nachteule

Schwierigkeit Zeitaufwand Kreativität

Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

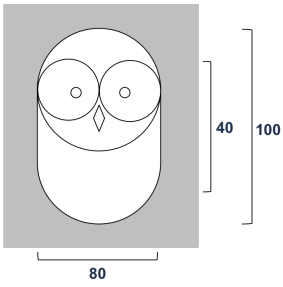
- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

In dieser Aufgabe wollen wir eine Eule nach dem folgenden Vorbild zeichnen:

Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente. Die folgenden Konstruktionsüberlegungen sollen dir dabei eine Hilfestellung bieten:



Testfälle

Wenn deine Eule grundsätzlich mit der abgebildeten Eule übereinstimmt, dann hast du die Lösung gefunden und die Aufgabe gelöst.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Bei Ellipsen und dem Spezialfall der Kreise wird immer der Mittelpunkt angegeben und nicht die linke obere Ecke.
- Achte auf die Reihenfolge!

W.2.8 Ghettoblaster

Schwierigkeit Zeitaufwand Kreativität

Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

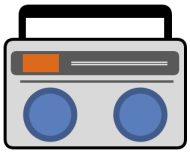
Beschreibung

Der sogenannte Ghettoblaster gilt quasi als der Vorgänger der Bluetooth-Box. Er bestand aus zwei Lautsprechern und meistens auch einem UKW-Radio, mit welchem man unterwegs Musik hören konnte.

Einen solchen Ghettoblaster wollen wir in dieser Aufgabe als Grafik realisieren.

Aufgabenstellung

Programmiere in Processing die Zeichnung eines Ghettoblasters. Er soll in dieser Form gestaltet werden:



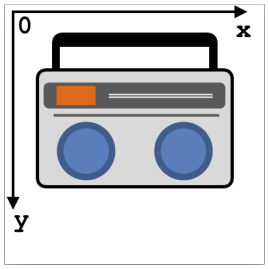
Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Bevor du dich an die Programmierung setzt, solltest du dir die Umsetzung überlegen. Am besten skizzierst du dir das Bild auf ein kariertes Blatt Papier. Danach zeichnest du das Koordinatensystem des Ausgabefensters ein. Wichtig hierbei ist, dass die y-Achse des Koordinatensystems von oben nach unten geht und der Nullpunkt in der linken oberen Ecke liegt:



Auf dem Blatt Papier kannst du anschließend bei allen Elementen die Höhe und Breite der einzelnen Elemente einzeichnen. Ebenso kannst du planen, wo die Koordinatenposition liegen wird.

- Nach der Planung kannst du mit der Programmierung beginnen. Hierbei wird es sehr helfen, wenn du die einzelnen Elemente deines Bildes mit entsprechenden Kommentaren versiehst. So behältst du immer den Überblick, an welcher Stelle welches Element gezeichnet wird. Das könnte in Java zum Beispiel so aussehen:

```
// Blaue Lautsprecherbox unten links
... (hier steht dann der entsprechende Code)
```

- Sollten Elemente nicht an der vermuteten Stelle gezeichnet werden: Prüfe die entsprechende Stelle im Code und schaue nach, ob sich nicht ein Gedanken- oder Tippfehler eingeschlichen hat. Probiere auch gerne verschiedene Werte in den Zeichenfunktionen aus. Das wird dir beim Verstehen der Funktionen sicher weiterhelfen.

W.2.9 Hallo Bello!

Schwierigkeit Zeitaufwand Kreativität

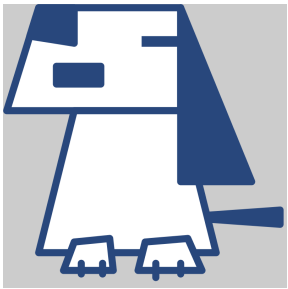
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

Auch einen Hund können wir mit einfachen geometrischen Objekten selber programmieren. Folgendes Bild haben wir als Vorgabe bekommen:



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- In diesem Bild haben wir viele Elemente, die sich nur über die Eckpunkte beschreiben lassen (Trapez, Linien, Dreiecke). Deshalb solltest du hier besonders vor dem Programmieren die genauen Positionen der Elemente planen. Nimm dir dazu ein (kariertes) Blatt Papier und zeichne die Elemente und deren Position ein. Ist dann alles genau geplant, läuft die Programmierung wesentlich einfacher.
- Achte darauf, welches Element über welches andere Element gelegt werden soll. Dies kannst du über die Reihenfolge festlegen, mit der du die Elemente in das Ausgabefenster zeichnest.
- Es ist empfehlenswert, wenn du zunächst alle Formen einzeichnest. Stimmt die Zeichnung dann mit dem Ergebnis einigermaßen überein, kannst du die Eigenschaften der Elemente noch hinzufügen (Farbe, Liniendicke, Linienart etc.).

W.2.10 Haus

Schwierigkeit Zeitaufwand Kreativität

Themen

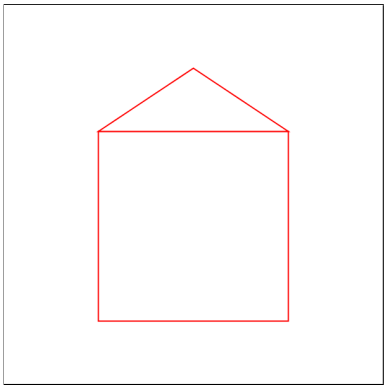
Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

Wir wollen uns ein virtuelles Haus bauen. Dafür haben wir ein 600×600 Pixel großes Fenster zur Verfügung gestellt bekommen.

Das Dach ist 300 Pixel breit und 100 Pixel hoch. Der Grundbau darunter ist 300 Pixel breit und 300 Pixel hoch.



Aufgabenstellung

Programmiere das angegebene Bild mithilfe der Processing-Grundelemente.

Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Schau' dir die Aufgabe zu den Grundelementen an.
- Das Bild hat ein Rechteck und ein Dreieck.
- Die Füllfarbe ist weiß und die Strichfarbe ist rot.

W.2.11 Daumen

Schwierigkeit Zeitaufwand Kreativität

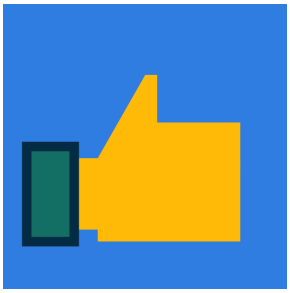
Themen

Mit dieser Aufgabe wollen wir Folgendes trainieren:

- Struktur eines einfachen Programms
- Aufbau und Abfolge von Programmanweisungen
- Ausgabe im grafischen Fenster

Beschreibung

Wir wollen einen Daumen zeichnen. Das folgende Bild ist keine exakte Vorgabe, aber soll zeigen, wie er aussehen könnte:



Hierfür haben wir folgende Farben benutzt:

- Hintergrund: 47 (Rot), 125 (Grün), 225 (Blau)
- Daumen: 255 (Rot), 186 (Grün), 8 (Blau)
- Hemdkragen:
  - Außen: 3 (Rot), 43 (Grün), 67 (Blau)
  - Innen: 19 (Rot), 111 (Grün), 99 (Blau)

Aufgabenstellung

Programmiere einen Daumen wie in der oben stehenden Darstellung. Verwende zum Zeichnen die Processing-Grundelemente Rechteck und Dreieck.

Testfälle

Siehe Aufgabenstellung.

Algorithmische Tipps

Wenn du stockst und nicht weiter weißt, dann versuch mal Folgendes:

- Die Bestimmung der Pixelpositionen ist am einfachsten, wenn du auf kariertem Papier die Grafik einzeichnest.
- Den Hemdkragen kannst du mit nur einem Rechteck zeichnen.
- Den Daumen kannst du mit vier Rechtecken und einem Dreieck zeichnen.