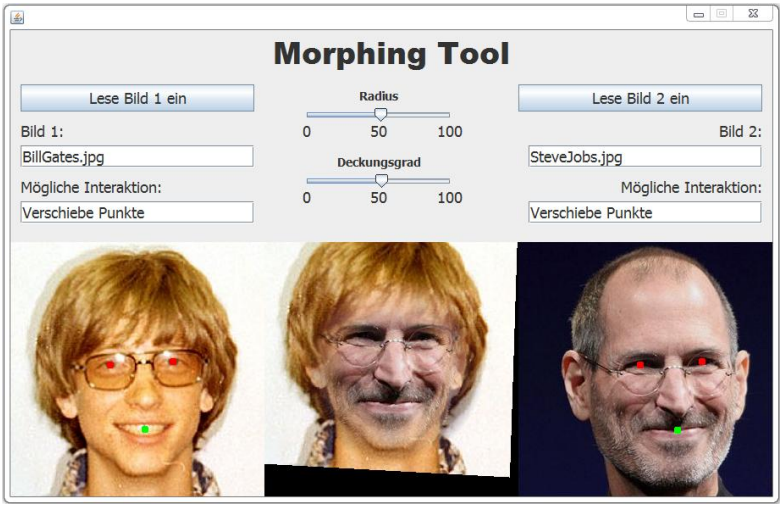
Morphing Puzzle Anleitung[1]



Ziel der Übung

Sie programmieren ein GUI, dass es erlaubt, Gesichter verschmelzen zu lassen (in der Fachsprache wird diese Technik als „Morphing“ bezeichnet). Dazu werden beliebige JPG Bilder von Gesichtern der Grösse 300×300 Pixel eingelesen. Per Mausklick werden die Augen und der Mundmittelpunkte mit farbigen Punkten markiert. Die Software transformiert die beiden Gesichter derart, dass die drei Punkte übereinander zu liegen kommen. Über zwei Schieberegler können die beiden transformierten Gesichter „vereint“ werden. Für eine optimale Übereinstimmung der Gesichter können die Punkte nachträglich per Maus einzeln verschoben werden. Das obenstehende Bild illustriert eine mögliche Programmvariante. Sie können das GUI nach Belieben gestalten.

Unsere Absicht dahinter

In der beruflichen Praxis werden Softwareprojekte in der Regel in Teams realisiert. Dabei wird die Programmierarbeit auf verschiedene Personen verteilt. Zentral sind klare Aufgaben für die Personen und die Kooperation zwischen den Personen sowie das Vereinbaren von Schnittstellen, damit die einzelnen Teile am Schluss problemlos zusammengesetzt werden können. Eine objektorientierte Programmiersprache wie Java unterstützt diese Form der Aufteilung mittels Klassen und Methoden.

Wir sind der Überzeugung, dass Sie im Rahmen des Ergänzungsfachs Informatik zumindest einen Einblick in diese Art der Zusammenarbeit erhalten sollten. Um Ihnen einen guten Einstieg zu ermöglichen, werden wir Ihnen in diesem Projekt die Organisation der Arbeiten zum grössten Teil vorgeben. Konkret wird die Software in drei Teile aufgeteilt. Jeder Teil wird durch ein Team von Experten, die sich gegenseitig unterstützen, umgesetzt. Anschliessend werden neue Teams mit je einem Experten aus jedem Bereich zur Fertigstellung der Software gebildet. Diese Form der Kooperation wird im Unterricht als Puzzle bezeichnet. Vielleicht verfügen Sie bereits schon über Erfahrung damit.

Es ist uns wichtig, dass Sie bei diesem Projekt persönlich Verantwortung übernehmen und positiv zum Gelingen des Projektes beitragen. Wir erwarten von Ihnen, dass Sie Initiative zeigen, vorrausschauend planen und **nicht alles in letzter Minute** erledigen. Wenn Sie gut in und zwischen den Gruppen zusammen arbeiten, werden Sie effizient vorankommen und Erfolg haben. Am Ende des Projektes werden wir Sie im Rahmen einer schriftlichen Einzelprüfung über Ihren persönlichen Beitrag zum Projekt prüfen.

[1] Version 4.0, Ralf Kretzschmar 2012, © no rights reserved. Die Fotos in der Abbildung stammen aus: http://en.wikipedia.org/wiki/File:Bill_Gates_mugshot.png, Time Warner, public domain und http://en.wikipedia.org/wiki/File:Steve_Jobs_Headshot_2010-CROP.jpg, Matt Yohe , cc-by-sa.

Inhaltsverzeichnis

Organisatorischer Ablauf

Doppellektionen	Themen	Bewertung
1	A. Einführung in das Projekt B. Einteilung Experten- und Softwareteams C. Expertenrunde 1: Organisation festlegen D. Softwareteamrunde 1: Organisation festlegen	1 Abgabe
2 – 3	E. Expertenrunde 2: Software Expertenteile erstellen	1 Abgabe
4	F. Softwareteamrunde 2: Software zusammensetzen	1 Abgabe
5	G. Bilder Galerie und Schlussevaluation des Projekts	
6	H. Prüfung	4 Fragen

Überblick über die Software

Für das Projekt sind die vier Klassen in **Abb.1** vorgegeben.

Die Klasse MorphingTool enthält die main Methode und ruft lediglich die Klasse MorphingGUI auf.

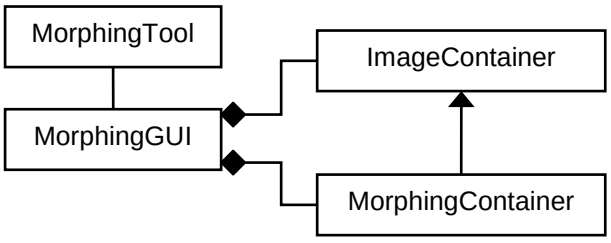


Abb.1: Klassendiagramm

Die Klasse MorphingGUI stellt die Oberfläche der Anwendung (das GUI) dar. Das GUI enthält zwei spezialisierte Container Typen zur Anzeige und Verarbeitung der zu morphenden Bilder: ImageContainer und MorphingContainer.

Die Klasse ImageContainer erweitert bzw. erbt von der Klasse JPanel (JPanel ist nicht in der Abbildung dargestellt). ImageContainer erweitert JPanel um Methoden, welche erlauben, Bilder sowie Augen- und Mundpunkte anzuzeigen.

Die Klasse MorphingContainer erweitert bzw. erbt von der Klasse ImageContainer. MorphingContainer erweitert ImageContainer um Methoden, welche erlauben, zwei Bilder mit Augen- und Mundpunkten einzulesen, diese zu überblenden und anzuzeigen.

Wichtig für das Verständnis der Software sind auch die Objekte und Variablen, welche zwischen den Klassen ausgetauscht werden. Der Austausch wird durch Methodenaufrufe realisiert. **Abb.2** gibt den Fluss der wichtigsten Objekte und Variablen schematisch wieder.

Das GUI Objekt liesst die Bilder als File Objekte ein und übergibt diese dem ImageContainer Objekt. Im ImageContainer Objekt werden die Bilder angezeigt.

Wird ein Augen- oder Mundpunkt über das GUI bestimmt, so werden die gewählten Koordinaten als Point Objekt dem ImageContainer Objekt übergeben. Zusätzlich wird

eine zugehörige Boolean Variable im ImageContainer Objekt auf true gesetzt, was anzeigt, dass der entsprechende Punkt nun definiert ist.

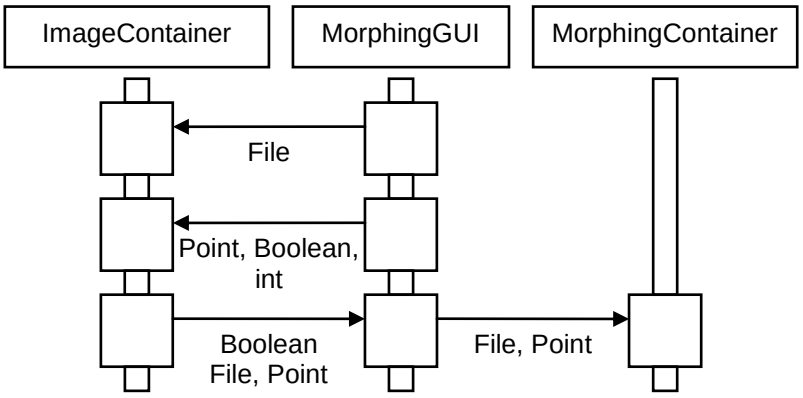


Abb.2: Austausch von Objekten und Variablen zwischen den Klassen.

Sind alle Bilder und Punkte in beiden ImageContainer Objekten definiert, so wird dem GUI Objekt durch die Boolean Variablen angezeigt, dass alles komplett ist. Das GUI Objekt liesst dann die File und Point Objekte aus den ImageContainer Objekten aus und übergibt diese zur weiteren Verarbeitung und Anzeige dem MorphingContainer Objekt.

Wie ein solcher Austausch von Variablen und Objekten in Java realisiert wird, entnehmen Sie dem Kapitel „Anhang: Getter und Setter“ am Ende dieses Dokuments.

Hinweis: Bei manchen Expertenthemen wird anstatt der Klasse MorphingGUI die Klasse MorphingTest verwedet. Der Zweck der Klasse MorphingTest besteht darin, die Funktionalität der zu schreibenden Methoden in den Klassen ImageContainer und MorphingContainer zu testen, ohne dafür ein GUI erstellen zu müssen. Die Klasse MorphingTest ist jeweils so konzipiert, dass der oben abgebildete Fluss der wichtigsten Objekte und Variablen möglichst gewährleistet bleibt, was den Zusammenbau der einzelnen Expertenteile zum fertigen Softwareprojekt erleichtert.

Experten Runden

Expertenrunde C – Organisation und Kommunikation

Legen Sie in jeder Expertengruppe Folgendes fest:

- **Wie** weit Sie während der nachfolgenden Expertenrunde E (Doppellektionen 2 und 3) zusammen arbeiten und kommunizieren wollen (Treffen, Chat, E-Mail, Social Network, etc.). Die Expertenrunde E ist nicht an die Präsenz in den Lektionen gebunden.
- **Wann** Sie während der nachfolgenden Expertenrunde E in und zwischen den Lektionen zusammen arbeiten und kommunizieren wollen. **Rechnen Sie auch ein**, dass es Fragen geben wird und es eine gewisse Zeit braucht, bis Sie eine Antwort bekommen.

  Tipp: Ich empfehle Ihnen dringend, als Vorbereitung für die Expertenrunde E die Source Code Vorlagen zu den Expertenteilen als Hausaufgabe **bis nächste Woche** zu studieren. Nur dann können Sie in der nächsten Woche sinnvoll arbeiten!

Hinweis: Eine Zusammenarbeit (und gestaffelte Aufteilung der Arbeit im Sinne von A programmiert eine Zeit lang und übergibt dann den Source Code weiter an B usw.) ist erlaubt, solange diese Zusammenarbeit koordiniert abläuft und Sie sich bemühen, den von anderen übernommenen Source Code zu verstehen und bei Unklarheiten auch rückzufragen. Ohne dieses Verständnis ist generell Frust vorprogrammiert!

Expertenrunde E – Umsetzung des Expertenteils

Organisation und Abgabe: Organisatorisch können Sie arbeiten wann und wo Sie wollen. In Ihrer Expertengruppe dürfen Sie eine Gruppenabgabe einreichen oder auch individuelle Abgaben Ihrer persönlichen Expertenteilumsetzung tätigen. Wandeln Sie dafür den entsprechenden NetBeans Projektordner in ein ZIP-File um und benennen Sie es so, dass

alle an der Abgabe Beteiligten ersichtlich sind. Geben Sie das ZIP-File spätestens **drei Tage** (bis maximal 23:59) **vor der vierten Doppellektion** in Ihrem Ergänzungsfach Ordner ab. Die Computer werden in den Lektionen für Sie stets reserviert sein. Ich werde ebenfalls anwesend sein und Ihnen bei Bedarf mit Rat und Tat zur Seite stehen.

Vorlagen: Jeder Expertenteil basiert auf einer Vorlage, welche Sie als geziptes NetBeans Projekt in Ihrem Ergänzungsfach Ordner finden. **Entzippen** Sie das Projekt zur Bearbeitung in Ihrem **persönlichen NetBeans Projekt Ordner**. Den Source Code der Vorlage sollen Sie **lesen**, **verstehen** und mit Hilfe der Expertenteil-Anleitungen **erweitern**.

Fragen: Ich beantworte gerne Ihre Fragen. Sie können mir diese während der offiziellen Unterrichtszeit mündlich stellen. Schriftliche Fragen können Sie mir jederzeit über das dafür vorgesehene Forum stellen. Mir ist es wichtig, dass Sie **zuerst selber** versuchen, das Problem zu lösen (d.h. nachdenken, Buch nachschlagen, Internetsuche oder Mitschülerinnen / Mitschüler fragen), bevor Sie mich kontaktieren, dann profitieren Sie mehr.

Wichtig: Rechnen Sie für die Antworten von mir etwas Zeit ein. Ich bin ich nicht permanent online. **Die Expertenteile sind darauf ausgelegt, dass es Fragen geben wird!** Fangen Sie daher **frühzeitig** an, dann kann und werde ich Ihnen gerne helfen! Wenn Sie alles auf die letzte Minute erledigen wollen, werden Sie auf sich allein gestellt sein.

Prüfung 1: Für die Abgabe Ihres Expertenteils können Sie maximal vier Punkte erzielen:

- 4 Punkte: Ihre Lösung ist vollständig lauffähig und gut lesbar (Refactoring!).
- 3 Punkte: Ihre Lösung ist vollständig lauffähig.
- 2 Punkte: Ihre Lösung ist teilweise lauffähig und gut lesbar (Refactoring!).
- 1 Punkt: Ihre Lösung ist teilweise lauffähig.
- 0 Punkte: Abgabe ist nicht lauffähig, zu spät oder gar nicht erfolgt.

Zwei Tage vor der vierten Doppellektion werde ich Musterlösungen zu jedem Expertenteil in den Ergänzungsfach Ordner stellen. Für die nachfolgende Softwareteamrunde F und für die Prüfung steht es Ihnen frei, ob Sie Ihre eigene Abgabe oder die Musterlösung als Grundlage nutzen wollen. Sie werden an der Prüfung beide Varianten zur Verfügung haben. Bedenken Sie für den Zusammenbau der Software, dass die Musterlösung nicht automatisch zu den Expertenteilen Ihrer Softwareteammitglieder kompatibel sein wird.

Prüfung 2: An der Prüfung wird Ihnen zur Theorie und Implementierung Ihres Expertenteils eine Frage gestellt werden. In dieser Frage können Sie maximal 4 Punkte erzielen.

Softwareteam Runden

Softwareteam Runde D – Planung und Kommunikation

Wenn Sie mit dieser Runde beginnen, kennen Sie Ihre Expertenrollen und haben Ihre Aufträge gelesen. Damit in der abschliessenden *Softwareteamrunde F* das Erstellen der Software effizient und reibungslos ablaufen kann ist es nötig, dass Sie vorausschauend planen und sich absprechen. Legen Sie dafür in jedem Softwareteam kurz Folgendes fest:

- **Was** Sie kommunizieren wollen. Es wird z.B. von Vorteil sein, Variablen, Objekte und Methoden, welche in jedem Expertenteil vorkommen, gleich zu benennen. Wie wollen Sie sich hierbei während des Projektes absprechen?
- **Wie** und ob Sie auch während der nachfolgenden *Expertenrunde E* als Softwareteam zusammen kommunizieren wollen (Treffen, Chat, E-Mail, Social Network, etc.).
- **Wann** und ob Sie während der nachfolgenden *Expertenrunde E* als Softwareteam zusammen kommunizieren wollen. Berücksichtigen Sie, dass einige Punkte, bei welchen eine Absprache sinnvoll wäre, erst während des Arbeitens erkannt werden.
- **Wie** Sie in der *Softwarerunde F* vorgehen und das Morphing Tool zusammen bauen wollen. In welcher Reihenfolge fügen Sie die Expertenteile zusammen? Wer arbeitet wann an der Software (speziell in der Zeit nach der dafür vorgesehenen Doppellektion)? Wie und warum setzen Sie das Prinzip „iterative Development“ ein?

- **Wo** Sie die grössten Schwierigkeiten bei den vorangehenden Punkten erwarten.

Abgabe: Halten Sie Ihre Vereinbarungen in einem Dokument fest. Legen Sie Ihr Dokument am Ende dieser Doppellektion mit **Ihren Namen** im Ergänzungsfach Ordner ab.

Tipp: Ergänzen Sie Ihr Dokument im Verlaufe des Projekt, damit es Ihnen als Hilfe für die nachfolgenden Runden dienen kann.

Prüfung 1: Sie erhalten 2 Punkte, falls in Ihrem Dokument alle fünf Punkte diskutiert werden und die Abgabe bis **heute 23:59** erfolgt ist. Falls nicht, erhalten Sie 0 Punkte.

Prüfung 2: Während der Prüfung teile ich Ihnen Ihr Dokument in der abgegebenen Version aus und stelle Ihnen eine Frage dazu, in welcher Sie 2 Punkte erzielen können.

Softwareteam Runde F – Software zusammensetzen

In dieser Runde setzten Sie Ihre Softwareteile aus den Expertenrunden gemäss Ihrer aktualisierten Vorgehensvereinbarung aus der *Softwareteamrunde D* zusammen.

Tipp: Es kann sein, dass im Scource Code beim zusammen setzen an der einen oder anderen Stelle noch ein „repaint();“ fehlt.

Abgabe: Sie geben Ihren NetBeans Projektordner als Zip-File mit Ihrem Softwaregruppennamen im Ergänzungsfach Ordern bis **23:59** Uhr **einen Tag vor der fünften Doppellektion** ab. Ich werde am Morgen der fünften Doppellektion eine Musterlösungung für das MorphingTool in den Ergänzungsfach Order stellen.

Prüfung 1: Für die Abgabe Ihrer Software können Sie maximal zwei Punkte erzielen:

- 2 Punkte: Ihre Lösung ist vollständig lauffähig.
- 1 Punkt: Ihre Lösung ist teilweise lauffähig.
- 0 Punkte: Abgabe ist nicht lauffähig, zu spät oder gar nicht erfolgt.

Prüfung 2: In der Prüfung stelle ich Ihnen zwei Fragen à 2 Punkten zum Thema Projektplanung. Eine Frage wird den Sinn und Nutzen einer Softwareprojektplanung betreffen. Die andere Frage wird Sie zu einem Vergleich auffordern zwischen der Planung dieses Projekts und der Planung Ihres Langzeitprojekts, die Matura zu erlangen.

Plenum

Plenum G – Galerie und Schlussevaluation

Sie erhalten Zeit, um ein paar schöne Überblendungen von Gesichtern zu erzeugen, welche Sie danach im Plenum präsentieren. Als Gesichter können Sie eigene Photos, Bilder aus dem Internet, etc. nutzen. Sie können Ihr eigenes Morphing Tool oder die Musterlösung dafür verwenden. Den Abschluss bildet ein Austausch im Plenum über das Projekt.

Prüfung

In der Prüfung können Sie maximal 18 Punkte erreichen, wovon 8 Punkte durch Abgaben im Vorfeld der abschliessenden Prüfung erzielt werden. Die Abgaben der beiden Softwareteamrunden ergeben je maximal 2 Punkte. Die Abgabe der *Expertenrunde E* ergibt maximal 4 Punkte. Zur *Expertenrunde E* und zu jeder der beiden Softwareteamrunden werden Ihnen an der abschliessenden Prüfung insgesamt vier Fragen gestellt, in welchen Sie zusammen maximal 10 Punkte erzielen können.

Die Noten werden gemäss nachfolgender Tabelle verteilt.

Punkte	Note	Punkte	Note	Punkte	Note
18	6.00	11	4.50	4	2.75
17	6.00	10	4.25	3	2.50
16	5.75	9	4.00	2	2.25
15	5.50	8	3.75	1	2.00
14	5.25	7	3.50	0	1.50
13	5.00	6	3.25	Nichterbrachte Leistung	1.00
12	4.75	5	3.00		

Expertenthema – GUI und Steuerung

Ziel: Sie gestalten das GUI visuell nach Ihrem eigenen Gutdünken. Dazu erzeugen Sie die Klasse MorphingGUI mit Hilfe des GUI Builders neu.

Theorie: Eines der grundlegendsten Konzepte objektorientierter Programmiersprachen besteht darin, Klassen mit klar definierten Zuständigkeiten zu erzeugen und von anderen Klassen abzugrenzen (Encapsulation). Dieses Konzept besagt für GUI Anwendungen, dass die graphische Oberfläche (View) vom Auswerten der Events (Control) und von den damit verknüpften Rechnungen / Verarbeitungen (Model) getrennt werden soll. Dieser Spezialfall der Encapsulation wird als Model-View-Control Konzept bezeichnet.

In Java wird dieses Konzept nicht konsequent durchgezogen. Die View und die Control Komponenten werden üblicherweise in einer Klasse, der GUI-Klasse, zusammengefasst. Der Grund dafür liegt in der häufigen Interaktion zwischen den View und Control Komponenten. View und Control in eine Klasse zusammenzufassen erleichtert die Programmierarbeit. In Ihrem Softwareprojekt stellt die Klasse MorphingGUI die View-Control Komponente dar, die Klassen ImageContainer und MorphingContainer das Model (siehe auch Abschnitt „Überblick über die Software“).

Wie im Abschnitt „Überblick über die Software“ ersichtlich ist, steuert das GUI den Austausch aller Variablen und Objekte. Daher ist es **besonders wichtig**, dass Sie verstehen, wie man die Übertragung von Daten zwischen Klassen programmiert. Eine Anleitung dazu finden Sie im Kapitel „**Anhang: Getter und Setter**“. Der Anhang gehört zur Theorie Ihres Expertenteils und somit für Sie zum **Prüfungsstoff**.

Überblick: Das GUI soll mindestens folgende funktionalen Elemente beinhalten.

- Zwei ImageContainer der Grösse 300×300 Pixel, welche die Originalbilder anzeigen. In einer ersten Runde werden durch Anklicken nacheinander die Augen- und Mundpunkte gewählt. In einer zweiten Runde können die Punkte verschoben werden.
- Einen MorphingContainer der Grösse 300×300 Pixel, welcher die Überblendung der Bilder anzeigt, nachdem in beiden ImageContainern alle drei Punkte gewählt sind.
- Zwei Regler oder etwas Ähnliches welche die Werte 0 bis 100 annehmen können. Sie dienen dazu das Morphing zu steuern.
- Zwei Buttons über welche Bilder eingelesen werden können.
- Textfelder, welche die Filenamen anzeigen und dem Benutzer mitteilen, was als Nächstes zu tun ist (z.B. „Linkes Auge markieren“ oder „Punkte verschieben“).

Zu tun: In den folgenden Schritten wird das GUI erstellt, vernetzt und getestet.

Wichtig: Lesen Sie zuerst den Source Code der Vorlage **gründlich** durch, bevor Sie sich auf das Programmieren stürzen. Es ist für das Programmieren zentral, dass Sie die Vorlage **verstanden** haben! Sie müssen dabei die unbekannten Methoden nur soweit verstehen, dass Sie erfassen, was das vorliegende Programm macht.

Hinweis: Die Vorlage Ihres Expertenteils ist alleine (noch) nicht lauffähig. Das wird die Vorlage erst, wenn Sie gemäss des ersten Punkts die GUI Klasse MorphingGUI dem Programm hinzugefügt haben. Dann können Sie die Software über die Klasse MorphingTool in der Vorlage aufstarten, welche die main Methode enthält.

- Leeres GUI mit den Namen „MorphingGUI“ erzeugen: Öffnen Sie die Projektvorlage im NetBeans und lassen Sie sich in der Ansicht „Projects“ die Verzeichnisse „MorphingToolGUIVorlage > Source Packages > morphingtool >“ anzeigen. Dann Rechtsklick auf „morphingtool“ und „New > JFrame Form…“ (**wichtig: JFrame** und nicht JPanel!) und dort den „Class Name“ „MorphingGUI“ wählen.
- **Wichtig:** Wechseln Sie in die Source Code Ansicht des GUI. Sollte dort eine main Methode erzeugt worden sein, entfernen Sie diese. Jetzt ist Ihre Vorlage lauffähig.
- Eigene Elemente wie der ImageContainer und MorphingContainer können Sie ganz normal über den GUI Builder mittels drag and drop in Ihr GUI einfügen. Dazu ziehen Sie das entsprechende File aus dem Projects Fenster direkt in das GUI.

- Um ein File zu öffnen, bietet sich die „FileChooser Swing Window Component“ des GUI Builders an. Diese muss im Inspector Fenster unter „Other Components“ verstaut werden (Rechtsklick auf „Other Components“ und „Add from Palette“ wählen, es genügt ein einziger FileChooser). Damiat der FileChooser über einen Button gestartet werden kann, muss in der Button Action Methode folgender Programmcode (aus diesem Dokument im Ergänzungsfach Ordner) kopiert **und angepasst** werden:

```
// Öffne den FileChooser Dialog
int returnVal = jFileChooser1.showOpenDialog(this);
if (returnVal == javax.swing.JFileChooser.APPROVE_OPTION) {
  // Lese das Bild in das File Objekt "file" ein
  java.io.File file = jFileChooser1.getSelectedFile();
  // Zeige den Filenamen im Textfeld jTextField1 an
  jTextField1.setText(file.getName());
    // Beim Einlesen von Files können Fehler auftreten. Der try catch
    // Block prüft auf Fehler und beendet allenfalls das Programm
  try {
    /* Ersetzen Sie diesen Kommentar mit dem Aufruf eines Setters,
     * welcher das File Objekt in den ImageContainer überträgt */
  } catch (IOException ex) {
    // Gebe den Fehler im NetBeans Output aus, beende Programm
    System.out.println(ex);
    System.exit(1);
  }
}
```

  Der oben fehlende Setter zum Übertragen eines File Objekts wird im nächsten Punkt beschrieben. Mit dem Setter kann das GUI dann bereits Bilder einlesen und anzeigen.

  Hinweis: Das Startverzeichnis des FileChooser lässt sich im Properties Fenster unter „currentDirectory“ einstellen. Über den „Default Editor“ lässt sich bequem ein Verzeichnis bestimmen. Im GUI Projektordner finden Sie einige Beispielbilder ☺.

- Nach dem Einfügen z.B. eines ImageContainers in Ihr GUI können Sie die Methoden der Klasse ImageContainer über das entsprechende Objekt im GUI verwenden. Der Aufruf jTextField1.setText(imageContainer1.ladeBildInImageContainer(file));“ ruft die Methode public String ladeBildInImageContainer(File file) der Klasse ImageContainer auf. Dazu wird das File Objekt file als Methodenparameter übergeben. Als Rückgabewert gibt die Methode einen Text als String Objekt zurück, welches im jTextField1 Textfeld Objekt angezeigt wird. Die Methode pubilc String ladeBildInImageContainer(File file) ist bereits für Sie implementiert. Ersetzen Sie den Kommentar aus dem kopierten Source Code und passen Sie die Objektnamen an.
- Die Augen- und Mundpunkte werden mit Mausklicks auf die Bilder in den ImageContainer bestimmt. Fügen Sie den ImageContainer den Mouse Event mouseClicked zu. Der GUI Builder erzeugt dabei eine Methode mit dem MouseEvent Objekt evt als Parameter, welches die genaue Mausposition enthält. Die Methode public Point getPoint() der Klasse MouseEvent gibt die Mausposition als Point Objekt zurück. Dieses Point Objekt können Sie direkt als Point Objekt dem ImageContainer übergeben.
- Übertragen Sie die Augen- und Mundpunkte mit geeigneten Settern in die ImageContainer. Ob Ihre Übertragung geklappt hat können Sie überprüfen, indem Sie in der Klasse ImageContainer die Koordinaten der übertragenen Punkte im Output Fenster des NetBeans ausgeben, z.B. mit „System.out.println(„Linkes Auge: „ + positionLinkesAuge);“. Bemerkung: positionLinkesAuge stellt ein Objekt der Klasse Point dar.
- Der Benutzer des GUI soll durch einen ersten Klick die Position des linken Auges auswählen. Das GUI gibt dem Benutzer im Gegenzug eine Anweisung der Art „Wähle rechtes Auge“ zurück. Mit einem zweiten Klick wird das rechte Auge bestimmt,

mit einem dritten der Mundmittelpunkt. Realisieren Sie diesen Ablauf mittels if-then Abfragen und Boolean Variablen (Repetition Boolean Variablen: siehe Unterrichtsunterlagen oder Internet). Eine mögliche Umsetzung wäre: Wenn das linke Auge noch nicht ausgewählt ist, trägt der Boolean des linken Auges den Wert false. Sobald das linke Auge ausgewählt ist, wird der Boolean auf true gestellt. Wenn nun irgendein Programmteil wissen möchte, ob schon ein linkes Auge definiert wurde, so braucht das Programmteil nur zu überprüfen ob der zugehörige Boolean true oder false ist. Definieren Sie alle Boolean Variablen als Instanzvariablen der Klasse ImageContainer.

- Soblad alle Augen- und Mundmittelpunkte in beiden ImageContainer durch den Anwender des Tools vollständig bestimmt sind, muss das GUI gemäss dem Kapitel „Überblick über die Software“ die folgenden Objekte in den MorphingContainer übertragen: beide Bilder, alle Augen- und Mundmittelpunkte sowie die Werte der beiden Schieberegler.

  Hinweis 1: Im ImageContainer ist das Bild als BufferedImage Objekt mit dem Namen „imageBuffer“ gespeichert.

  Hinweis 2: Fragen Sie bei jedem mouseClicked Event aus dem GUI heraus ab, ob alle Boolean Variablen beider ImageContainer Objekte den Wert true haben.

- Kontrollieren Sie nun ob die Übertragung aller Variablen und Objekte in den MorphingContainer geklappt hat. Geben Sie dazu in der Klasse MorphingContainer die Punktkoordinaten und Schiebereglerwerte in den NetBeans Output aus. Lassen Sie auch eines der Bilder im MorphingContainer Fenster anzeigen. Ein übertragenes BufferedImage Objekt, z.B. „bild1“, können Sie anzeigen lassen, indem Sie dieses der Instanzvariablen imageBuffer zuweisen und die Anzeige mit „repaint();“ aktualisieren (konkret: „imageBuffer = bild1;“ und „repaint();“). Da die Klasse MorphingContainer von der Klasse ImageContainer erbt, ist die Instanzvariable imageBuffer sichtbar.
- Zum Schluss noch Folgendes: Je nach Wert des ersten Schiebereglers soll entweder das eine oder das andere Bild direkt im MorphingContainer angezeigt werden.
- **Wichtig:** Obwohl die nachfolgenden Schritte freiwillig sind, erstellen Sie dort Methoden, welche Sie für das finale Morphing Tool brauchen. Sollten Sie die nachfolgenden Schritte nicht durchführen, so können Sie die Methoden welche erstellt werden jedoch problemlos der Musterlösung entnehmen.
- **Freiwillig für Schnelle:** Löschen Sie alle Punkte, wenn ein neues Bild eingelesen wird (d.h. setzten Sie alle Boolean Variablen auf false). Geben Sie im GUI eine entsprechende Meldung aus, wenn ein Bild mit einem anderen Format als 300×300 eingelesen wird. In diesem Fall sollte wieder das blaue Startbild angezeigt werden. Die Breite und Höhe eines BufferedImage Objektes erhalten Sie über die Methoden public int getWidth() und public int getHeight() der Klasse BufferedImage.
- **Freiwillig für ganz Schnelle:** Lassen Sie zu, dass man gewählte Punkte nachträglich per Maus verschieben kann. Das Verschieben der Punkte realisieren Sie über die Mouse Events mousePressed (d.h. die Maustaste wird gedrückt), mouseDragged (die Maus wird bei gedrückter Taste verschoben) und mouseReleased (die Maustaste wird losgelassen). Definieren Sie alle drei Events für jeden ImageContainer und leiten Sie aus jedem Event die akutellen Mauspositionen an die ImageContainer weiter.

  Ersetzen Sie im ImageContainer jeweils denjenigen „alten“ Punkt, der am nächsten zum neuen Punkt liegt. Die Methode public double distance(Point point) der Klasse Point berechnet die Distanzen für Sie.

  Hinweis: Es kann gut sein, dass einige der System.out.println() Anweisungen mehrfach ausgeführt werden. Der Grund liegt darin, dass bei einem Mausklick nicht nur ein mouseClicked Event erkannt wird, sondern gleichzeitig auch ein mousePressed und ein mouseReleased Event. Diese Unsauberkeit nehmen wir aber in Kauf, eine entsprechende saubere Abfrage wäre nur mit grossem Aufwand zu realisieren.

Expertenthema – Bilder erzeugen und darstellen

Ziel: Ihre Aufgabe besteht darin, die Augen- und Mundmittelpunkte bei den Bildern visuell darzustellen und das Überlagern der Bilder gemäss den Schieberegler Einstellungen des GUI zu realisieren.

Theorie: Ein Bild das in Java erzeugt wird, existiert zuerst nur im Computerspeicher. Den Prozess das Bild auf den Bildschirm darzustellen, nennt man Rendern.

In unserer Anwendung möchten wir ein Foto, das auf der Harddisk gespeichert ist, mehrfach bearbeiten. Damit dafür nicht jedes Mal auf die langsame Harddisk zugegriffen werden muss, wird zuerst das Foto als schlankes BufferedImage Objekt im Computerspeicher abgelegt. Um zu verhindern, dass wir das Originalbild im BufferedImage Objekt nicht bei jeder Bearbeitung zerstören, wird zusätzlich im Computerspeicher eine Arbeitskopie erstellt. Die Arbeitskopie wird jedoch als aufwändigeres Graphics2D Objekt realisiert, da die Klasse Graphics2D O diverse nützliche Methoden mit sich bringt und direkt von Java als Grundlage für das Rendern gebraucht wird.

Ein GUI wird als Desktopfenster dargestellt, welches von anderen Fenstern zeitweise überdeckt werden kann. Daher muss Java automatisch in der Lage sein, bei Bedarf die gewünschten Graphics2D Objekte erneut zu rendern. Dafür ruft Java automatisch die Methode public void paintComponent(Graphics grafik) der Klasse JComponent auf. Gerendert wird bei diesem Aufruf dann alles, auf was die Referenzvariable grafik verweist. Daher macht es Sinn, von Anfang an alles was auf den Bildschirm dargestellt werden soll, innerhalb der Methode paintComponent() dem Graphics Objekt grafik zu zuweisen.

Bemerkung: Unsere Arbeitskopie ist jedoch nicht als Graphics sondern als Graphics2D Objekte abgelegt. Das ist jedoch kein Problem. Da die Klasse Graphics2D von der Klasse Graphics erbt, kann die Graphics Referenzvariable grafik auf beides, Graphics und Graphics2D Objekte, verweisen (Polymorphismus). Der Vorteil Graphics2D Objekte zu verwenden liegt darin, dass aufgrund der Vererbung nicht nur die Methoden der Klasse Graphics sondern auch die Methoden der neueren Klasse Graphics2D sichtbar sind.

Um ein Foto anzeigen zu können, muss dieses aus der Methode paintComponent() heraus aufgerufen werden. Die Methode paintComponent() in der Klasse JComponent direkt zu verändern wäre mühsam. Eleganter ist es, eine Klasse zu schreiben, welche direkt von der Klasse JComponent oder von einer Ihrer Kindklassen (z.B. von JPanel) erbt. In der neuen Klasse wird die originale Methode paintComponent() mit einer eigenen Version der paintComponent() Methode überschrieben. Ruft Java dann automatisch die Methode paintComponent() zum Rendern auf, so wird die eigene Version verwendet.

In der Vorlage zu diesem Expertenteil erbt die Klasse ImageContainer von der Klasse JPanel. Die originale paintComponent() Methode wird somit von der paintComponent() Methode in der Klasse ImageContainer überschrieben. In der ImageContainer Variante der Methode paintComponent() mit wird das Originalbild zuerst aus dem BufferedImage Objekt imageBuffer zur weiteren Bearbeitung in das Graphics2D Objekt angezeigteGrafik kopiert. Das geschieht über den Aufruf „angezeigteGrafik.drawImage(imageBuffer, 0, 0, null);“. Danach kann die Arbeitskopie des Bildes in der Methode paintComponent() über die Methoden der Klasse Graphics2D bearbeitet werden.

Hinweis: Mit dem Aufruf „repaint();“ veranlasst man Java dazu, die Methode paintComponent() sofort aufzurufen. Somit lässt sich die Methode paintComponent() indirekt auch jederzeit im Programm aufrufen, um die Bildschirmanzeige gezielt zu aktualisieren.

Zu tun: Sie sollen die folgenden funktionalen Elemente den Klassen ImageContainer und MorphingContainer hinzufügen. Um Ihre Methoden zu testen werden Sie kein eigentliches GUI benutzen, sondern die Hilfsklasse MorhpingTest verwenden, welche das eigentliche GUI ersetzt.

- Starten Sie die Vorlage. Zwei Bilder werden in zwei verschiedenen Fenstern angezeigt (ImageContainer Objekte) , ein drittes Fenster ist leer (MorphingContainer Objekt).

- In den nächsten Schritten stellen Sie in der Klasse ImageContainer die Augenpunkte rot und den Mundmittelpunkt grün dar.

  Wichtig: Lesen Sie zuerst den Source Code der Vorlage **gründlich** durch, bevor Sie sich auf das Programmieren stürzen. Es ist für das Programmieren zentral, dass Sie die Vorlage **verstanden** haben! Sie müssen dabei die unbekannten Methoden nur soweit verstehen, dass Sie erfassen, was das vorliegende Programm macht. Die folgenden Hinweise werden Ihnen das Lesen des Source Codes erleichtern.

  - Die Variablen, welche das Aussehen der Punkte definieren, sind in der Klasse ImageContainer bereits als Instanzvariablen deklariert. Die Positionen der Augen- und Mundmittelpunkte werden von der Klasse MorphingTest (Morphing-Tool Ersatz) dem „zweiten" Konstruktor der Klasse ImageContainer übergeben.
  - Für die Steuerung des fertigen Softwareprojekts ist es wichtig, dass in der Klasse ImageContainer festgehalten wird, welche der Punkte bereits gesetzt sind. Dazu werden in der Klasse MorphingTest entsprechende Boolean Variablen definiert (Repetition Boolean Variablen: siehe Unterrichtsunterlagen oder Internet) und ebenfalls per Konstruktor in die ImageContainer Objekte übertragen. Wenn das linke Auge noch nicht ausgewählt ist, trägt der Boolean des linken Auges den Wert false. Sobald die Position des linken Auges bestimmt ist, wird der Boolean auf true gestellt. Welche Boolean Variablen wann auf true oder false gesetzt werden, wird im Expertenteil „GUI und Steuerung" festgelegt (siehe Abschnitt „Überblick über die Software"). Am besten legen Sie daher gleich zusammen mit Ihrem GUI Experten sinnvollere Namen für diese Boolean Variablen fest, die im Source Code enthaltenen Namen entsprechen nicht einem guten Programmierstiel. Um zu testen, ob alle Punkte angezeigt werden, sind alle Boolean Varialben in der Klasse MorphingTest auf den Wert true gesetzt.
- Stellen Sie nun die Augen- und Mundmittelpunkte graphisch dar. Zeichnen Sie einen Punkt nur dann, wenn der entsprechende Boolean auf true steht. Einen Punkt zeichnen Sie z.B mit der Methode public void fillOval(int x, int y, int width, int height) der Java Klasse Graphics, wobei x und y die Koordinaten der linken oberen Ecke des Ovals bezeichnen (nicht den Mittelpunkt!) und width die Breite und height die Höhe des Ovals bestimmen. Da die Klasse Graphics2D von der Klasse Graphics erbt, können Sie die Methode fillOval() auch über ein Graphics2D Objekt aufrufen.
- Übertragen Sie beide Bilder mit allen Punkten aus den ImageContainer Objekten und auch die Werte der Schieberegler aus der MorphingTest Klasse in den MorphingContainer. . Realisieren Sie die Übertragung mittels Gettern und Settern (siehe „Anhang: Getter und Setter"). Im fertigen Morphing Tool werden die Daten über das GUI übertragen (siehe Kapitel „Überblick über die Software"). In Ihrem Expertenteil übernimmt die Klasse MorphingTest diese Rolle. Für Testzwecke sind in der Klasse MorphingTest die beiden Schieberegler auf die Werte „100" und „0" voreingestellt.
- **Achtung:** Bei der Übertragung mit Getter und Setter Methoden übertragen Sie nicht die eigentlichen Objekte sondern nur Referenzen auf diese Objekte. D.h. wenn Sie im MorphingContainer ein Bild verändern, so verändern Sie über die übertragene Referenzvariable auch gleich das im ImageContainer angezeigte Bild mit. Um das zu verhindern, müssen Sie neue, unabhängige BufferedImage und Point Objekte erstellen und den Inhalt der zu kopierenden Objekte in diese übertragen. Diese Art des Kopierens nennt sich „klonen". Sie sind für das Klonen der Bilder zuständig, das Klonen der Point Objekte realisiert Ihr Transformationsexperte. Gehen Sie wie folgt vor:
  - Weisen Sie den Instanzvariablen imageBuffer1 und imageBuffer2 in der Klasse MorphingContainer neue BufferedImage Objekte über den new Operator zu. Wie das geht können Sie in der Klasse ImageContainer nachrecherchieren. Mit den public Methoden int getWidth(), int getHeight() und int getType() der Klasse BufferedImage können Sie die Breite, Höhe und den Typ der übertragenen BufferedImage Objekte auslesen und direkt den Konstruktoren der neuen BufferedImage Objekte übergeben.

  - Mit der Methode Raster getData() der Klasse BufferedImage lesen Sie die Daten aus einem BufferedImage Objekt als Raster Objekt aus. Mit der Methode void setData(Raster r) der Klasse BufferedImage lesen Sie die Daten, welche in einem Raster Objekt enthalten sind in das BufferedImage Objekt ein. „Klonen" Sie mit diesen Methoden die Daten der übertragenen BufferedImage Objekte in die neuen BufferedImage Objekt.

- Im MorphingContainer ist bereits die Methode ueberblendeDieBilder(…) teilweise für Sie ausprogrammiert. Dort überblenden Sie die beiden Bilder in den imageBuffer1 und imageBuffer2 Objekten in das BufferedImage Objekt imageBuffer. Das imageBuffer Objekt ist in der Klasse ImageContainer definiert und somit auch in der Kindklasse MorphingContainer sicht- und aufrufbar. Was immer Sie in das imageBuffer Objekt hinein schreiben, es wird im MorphingContainer Fenster angezeigt. Gehen Sie für das Überblenden punktweise vor, d.h. berechnen Sie für jede Koordinate (x,y) in imageBuffer1 und imageBuffer2 den überblendeten Punkt an der Koordinate (x,y) im imageBuffer. Halten Sie sich an nachfolgendes Vorgehen und ziehen Sie die deutsche oder englische API betreffend der Klassen BufferedImage und Color zu Rate. (Siehe http://www.dpunkt.de/java/Index/index.html , http://java.sun.com/javase/6/docs/api/)
  - Lesen Sie die RGB Farbwerte aus den imageBuffer1 und imageBuffer2 Koordinaten aus und übertragen Sie diese via Color Konstruktor in Color Objekte.
  - Lesen Sie die Farbkomponenten Rot aus diesen Color Objekten heraus und berechnen Sie daraus mit den Einstellungen der Schieberegler die neue, kombinierte Rot-Komponente. Die Funktionsweise der Schieberegler ist im Source Code erklärt. **Achtung:** Stellen Sie dabei sicher, dass der Wert „255" nicht überschritten wird. Berechnen Sie analog die Grün- und Blau-Komponenten.
  - Erzeugen Sie ein neues Color Objekt, welchem Sie die vorher berechneten Farbkomponenten (via Color Konstruktor) übergeben.
  - Lesen Sie den RGB Farbwert aus dem neuen Color Objekt aus und übertragen Sie diesen an die entsprechende Koordinate im imageBuffer Objekt.

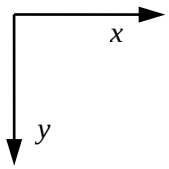
- **Wichtig:** Obwohl die nachfolgenden Schritte freiwillig sind, erstellen Sie dort Methoden, welche Sie für das finale Morphing Tool brauchen. Sollten Sie die nachfolgenden Schritte nicht durchführen, so können Sie die Methoden welche erstellt werden jedoch problemlos der Musterlösung entnehmen.

- **Freiwillig für Schnelle:** Als letztes müssen Sie noch alles dafür vorbereiten, damit die Transformationen der Bilder, welche Ihr Experte erstellt, mit eingebaut werden können. Sie sollen dies im Folgenden mit einer Testtransformation überprüfen. Das Vorgehen dafür ist knifflig. Daher ist in der Vorlage die entsprechende Methode wendeTransformationenAn() in der Klasse MorphingContainer fast vollständig ausprogrammiert. Markieren Sie die Methode im NetBeans und kommentieren Sie diese über „Source > Toggle Comment" aus. Führen Sie die folgenden Schritte durch:
  - Zum Schluss der Methode steht der Kommentar „Klonen kann sich lohnen". Ersetzen Sie diese Zeile durch einen Programmcode, der die Bilder der ArbeitsImageBuffer Objekte in die BufferedImage Objekte hinein klont.
  - Weisen Sie den beiden AffineTransform Instanzvariablen in der Klasse MorphingContainer zwei Testtransformationen über die statische public Methode static AffineTransform getScaleInstance(double Sx, double Sy) zu und wählen Sie für die Parameter Sx und Sy jeweils den Wert „1.5".
  - Die Methode wendeTransformationenAn() soll aufgerufen werden, bevor Sie die beiden Bilder überblenden. Die Testtransformation bewirkt eine Vergrösserung der überblendeten Bilder, so dass Sie nur je einen Teil der Bilder vergrössert sehen.

Expertenthema – Affine Transformation

Ziel: Ihre Aufgabe besteht darin, die zwei Gesichter derart zu verschieben, dehnen und verzerren, dass die Augen und die Mundmittelpunkte exakt aufeinander zu liegen kommen und sich beide Gesichter genau in der Bildmitte befinden. Dazu werden Sie in diesem Teil die entsprechende „affine Transformation“ programmieren.

Theorie: Der Ursprung des Koordinatensystems in einem Container ist oben links, die x-Achse zeigt gegen rechts und die y-Achse gegen unten.

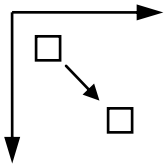


In einer affinen Transformation ändern Sie die „alten“ Koordinaten *(x,y)* in die „neuen“ Koordinaten *(x',y')* um. Die Formeln der affinen Transformation sind in Matrizenschreibweise dargestellt. Die Allgemeine Formel der affinen Transformation lautet

$$(x', y') = (x, y)\begin{pmatrix} a & b \\ c & d \end{pmatrix} + (e,\ f) = (ax + cy + e, bx + dy + f).$$

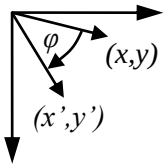
Die Bedeutung der Parameter *a, b, c, d, e* und *f* wird klar, wenn man die folgenden vier Spezialfälle dieser Formel genauer ansieht.

Translation: Die Koordinate *(x,y)* wird um den Betrag *Tx* in x-Richtung und den Betrag *Ty* in y-Richtung verschoben. Die Formel dafür lautet



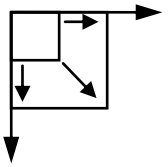
$$(x', y') = (x, y)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + (Tx, Ty) = (x + Tx, y + Ty).$$

Rotation: Die Rotation des Punktes *(x,y)* mit dem Winkel φ um den Ursprung (Koordinate *(0,0)*) wird in diesem Koordinatensystem im Uhrzeigersinn ausgeführt und mit der folgenden Formel beschrieben



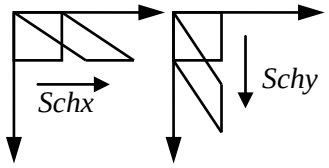
$$(x', y') = (x, y)\begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} = (x\cos\varphi - y\sin\varphi, x\sin\varphi + y\cos\varphi).$$

Skalierung: Die Koordinate *(x,y)* wird mit dem Faktor *Sx* in x-Richtung und dem Faktor *Sy* in y-Richtung multipliziert. Dies hat eine Vergrösserung bzw. eine Verkleinerung entlang der entsprechenden Achse zur Folge. Die Formel dafür lautet



$$(x', y') = (x, y)\begin{pmatrix} Sx & 0 \\ 0 & Sy \end{pmatrix} = (x \cdot Sx, y \cdot Sy).$$

Scherung: Die Scherung bewirkt eine Verzerrung in x- bzw. in y-Richtung, welche mit steigender y- bzw. x-Koordinate zunimmt. Die Formel dafür lautet



$$(x', y') = (x, y)\begin{pmatrix} 1 & Schy \\ Schx & 1 \end{pmatrix} = (x + y \cdot Schx, x \cdot Schy + y).$$

Um eine Transformation eines Objektes, z.B. eines Dreiecks oder Vierecks mit diesen Formeln durchzuführen, wird üblicherweise zuerst eine Ecke oder der Mittelpunkt des Objekts durch den Ursprung verschoben. Danach werden nacheinander in der korrekten Reihenfolge Rotation, Skalierung und Scherung durchgeführt und schliesslich das transformierte Objekt wieder an die gewünschte Stelle zurückverschoben. Die Reihenfolge der einzelnen Schritte ist ebenso entscheidend, wie die Wahl der entsprechenden Parameter.

Prüfung: In der Prüfung werden Sie auf Papier ein geometrisches Objekt von einem Anfangszustand in einen Endzustand mittels affiner Transformation überführen. Dazu skizzieren und formulieren Sie die einzelnen Schritte und bestimmen die einzelnen Parameter. An der Prüfung kennen Sie die nötigen Formeln dazu **auswendig**.

Zu tun: In Ihrem Expertenteil werden Sie nicht direkt Gesichter transformieren sondern Sie erstellen die gewünschte Transformation anhand von Testdreiecken.

Wichtig: Lesen Sie zuerst den Source Code der Vorlage **gründlich** durch, bevor Sie sich auf das Programmieren stürzen. Es ist für das Programmieren zentral, dass Sie die Vorlage **verstanden** haben! Sie müssen dabei die unbekannten Methoden nur soweit verstehen, dass Sie erfassen, was das vorliegende Programm macht.

- Starten Sie als erstes die Vorlage. Sie sehen zwei Fenster mit je einem grünen und einem roten Dreieck. Jedes rote Dreieck repräsentiert die beiden Augen und den Mundmittelpunkt eines Gesichts in den ImageContainer Objekten (siehe Abschnitt: „Überblick über die Software“). Das grüne Dreieck stellt das Ergebnis der gelungenen Transformation dar, d.h. diejenigen Augen- und Mundpunktkoordinaten mit welchen die Gesichter MorphingContainer Objekte dargestellt werden. In beiden Fenstern weisen die grünen Dreiecke daher die gleiche Grösse und Lage auf. Wenn Sie nun die beiden roten Dreiecke richtig transformieren, werden die beiden roten Dreiecke nach der Transformation identisch sein. Damit Sie dieses überprüfen können sind die grünen Zieldreiecke in beide Fenster eingezeichnet. Wenn alles geklappt hat, sollten die transformierten roten Dreiecke exakt auf den grünen Zieldreiecken zu liegen kommen.

  Wichtig: Zu Beginn der Transformation kennen Sie nur die beiden Gesichter in den ImageContainer Objekten, welche den roten Dreiecken entsprechen. Wie die transformierten Gesichter aussehen wissen Sie vor der Transformation noch nicht. D.h. Ihre Transformation darf nur die Koordinaten der beiden roten Dreiecken für die Berechnung der Transformationen verwenden, die Koordinaten der grünen Dreiecke dürfen Sie **NICHT** für die Berechnung der Transformationen verwenden! Sie dienen in Ihrem Expertenteil rein der Überprüfung.
- Die Gesichter bzw. roten Dreiecke sollen derart transformiert werden, dass
  - die Eckpunkte beider Dreiecke exakt aufeinander zu liegen kommen. Wenn die x- oder y-Koordinate zweier sich entsprechender Ecken einen Unterschied aufweist, so stellen Sie die neuen Koordinaten der Ecken auf den Mittelwert ihrer ursprünglichen Koordinaten ein (d.h. legen Sie den neuen Punkt genau zwischen die alten Punkte).
  - die beiden Augen bzw. oberen Eckpunkte der Dreiecke dieselbe y-Koordinate aufweisen (d.h. waagerecht zu liegen kommen).
  - beide Dreiecke in der „Mitte“ des Bildes zu liegen kommen. D.h. der Abstand vom linken Auge zum linken Rand ist gleich gross wie der Abstand vom rechten Auge zum rechten Rand. Der Abstand der Augen zum oberen Rand ist gleich gross wie der Abstand vom Mundpunkt zum unteren Rand.
  - **Wichtig:** Noch einmal: Sie dürfen für die Berechnungen **nur die Koordinaten der roten Dreiecke verwenden**! Die Koordinaten der grünen Dreiecke sind tabu.
- Lösen Sie **zuerst** die Aufgabe **AUF PAPIER** indem Sie für jeden Transformationsschritt eine neue Skizze anfertigen und bestimmen Sie so die Reihenfolge der einzelnen Transformationen und die Berechnungen der einzelnen Parameter.
- In Java steht Ihnen die Klasse AffineTransform zur Verfügung, welche unter anderem die nachfolgenden public Methoden enthält. Weitere Informationen siehe: Java API (http://www.dpunkt.de/java/Index/index.html, http://java.sun.com/javase/6/docs/api/).
  - Transformationen: void translate(double Tx, double Ty), void rotate(double φ), void scale(double Sx, double Sy), void shear(double Schx, double Schy). Über diese Methoden werden die entsprechenden Parameter in der Matrize des AffineTransform Objekts bestimmt.

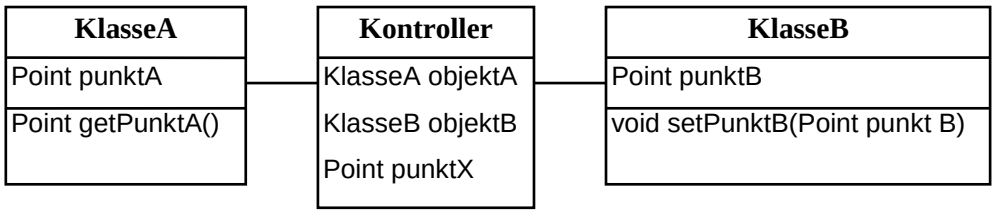
    Achtung! Wenn Sie mehrere Methoden auf einem AffineTransform Objekt aufrufen, dann werden die Transformationen in **umgekehrter** Reihenfolge ausgeführt!
  - Mehrere AffineTransform Objekte zu einem einzigen AffineTransform Objekt zusammenfügen: void preConcatenate(AffineTransform aT). Die Transformationen werden in der Reihenfolge des Einfügens eingetragen.

  - Transformation auf mehrere Punkte anwenden: void transform(double[] srcPts, int srcOff, double[] dstPts, int dstOff, int numPts). Der double Array srcPts enthält Punkt für Punkt die zu transformierenden Koordinaten in der Reihenfolge (x0, y0, x1, y1, …), wobei (x0, y0) den ersten Punkt bezeichnen, (x1, y1) den zweiten, etc. Wählt man als Parameter z.B. „(punkte, 0, punkte, 0, 3)“ so werden die ersten drei Punkte des Arrays „punkte“ ausgelesen, transformiert und wieder zurück in den Array „punkte“ geschrieben.

- Im richtigen Morphing Tool wird die Transformation in der Klasse MorphingContiner berechnet und durchgeführt. Dazu ist es notwendig, beide Bilder mitsamt den Punkten in das entsprechende MorphingContainer Objekt zu übertragen. Ihr Expertenteil ist jedoch anders aufgebaut, was Sie beim Zusammenbau der Software berücksichtigen müssen. Die Klasse MorphingTest erzeugt zwei MorphingTestContainer, welche je ein rotes und grünes Dreieck anzeigen. Die roten Dreiecke sind in der Klasse MorphingTest definiert, die grünen Dreiecke im MorphingTestContainer. Da Sie nur mit den beiden roten Dreiecken arbeiten, berechnen Sie die affinen Transformationen in der Klasse MorphingTest, übertragen diese anschliessend per Setter Methode in die MorphingTestContainer Objekte und wenden Sie dort auf die roten Dreiecke an (siehe „Anhang: Getter und Setter“).
- Grafiken werden in Containern angezeigt. Die Container Klasse JPanel enthält die Methode void setAffineTransformation(AffineTransform aT). Wird mit dieser Methode einem Container eine affine Transformation zugeordnet, so wird alles was angezeigt werden soll vor dem Anzeigen dieser Transformation unterzogen. Da die Klasse MorphingTestContainer von der Klasse JPanel erbt, kann diese Methode auch direkt auf einem MorphingTestContainer Objekt aufgerufen werden. Dasselbe gilt im richtigen Morphing Tool ebenso für die Objekte der Klasse MorphingContainer.
- Zwei Tipps zu guter Letzt.
  - Wenn Sie basierend auf den grünen Dreiecken eine Transformation berechnen, dann müssen Sie die grünen Dreiecke ebenfalls dieser Transformation unterwerfen, bevor Sie basierend auf den grünen Dreiecken eine nachfolgende Transformation bestimmen.
  - Wählen Sie vernünftige Namen für die Point Objekte la1, ra1, … Diese Namen entsprechen keinem guten Programmierstil.
- **Freiwillig für Schnelle:** Helfen Sie einem anderen Mitglied Ihres Softwareteams!

Anhang: Getter und Setter

Der folgende Abschnitt fasst zusammen, wie man Daten aus einer Klasse ein- und ausliest. Die drei Klassen sind gemäss untenstehender Abbildung gegeben. Der Inhalt des Point Objekts punktA der KlasseA soll in das Point Objekt punktB der KlasseB übertragen werden. Die Übertragung soll über die Klasse Kontroller stattfinden. Die Point Objekte punktA, punktB und punktX sind in diesem Beispiel als Instanzvariablen deklariert.



Üblicherweise werden Instanzvariablen als private deklariert und über public Methoden ausgelesen, den sogenannten „Getter“ Methoden. Die Methode getPunktA() gibt eine Referenz auf das punktA Objekt zurück.

```
public Point getPunktA() {
        return punktA;
}
```

Analog werden die Instanzvariablen über public Methoden geändert, den sogenannten „Setter“ Methoden. Die Methode setPunktB() übergibt mittels des Schlüsselwortes this der Instanzvariablen punktB die Referenz der Parametervariablen punktB.

```
public void setPunktB(Point punktB) {
        this.punktB = punktB;
}
```

In der Klasse Kontroller kann nun das Übertragen wie folgt bewerkstelligt werden

```
punktX = objektA.getPunktA(); // Übertrage Inhalt punktA nach punktX
objektB.setPunktB(punktX); // Übertrage Inhalt punktX nach punktB
```

Dasselbe in einer Kurzvariante, welche ohne das Point Objekt punktX auskommt

```
objektB.setPunktB(objektA.getPunktA()); // Inhalt punktA nach punktB
```

Es können auch Setter Methoden sinnvoll sein, in welchen mehrere Parameter gleichzeitig übergeben werden, z.B.

```
public void uebertrageBeidePunkte(Point punkt1, Point punkt2) {
…
```

Hinweis: Im NetBeans Editor können über das Menu „Source“ und den Unterpunkt „Insert Code…“ automatisch Getter und Setter erzeugt werden.