

## Übungen zu Graphikprogrammierung in C++

Was man wissen muss:

- Konzepte der Objektorientierung
- Java-Syntax zu Klassen, Objekten usw..

### Aufgabe 12 (H) Objektorientierung in C++

Das Konzept der Objektorientierung sollte Ihnen von früheren Semestern bekannt sein. Sie sind auch mit der Java-Syntax für OO-Programmierung vertraut. Diese Aufgabe soll dazu dienen, Sie in die C++-Syntax für die OO-Programmierung einzuführen und die Unterschiede zu Java aufzuzeigen.

- Erzeugen Sie folgende Klassen in C++: `Polygon`, eine Klasse, die Polygone modelliert, `Triangle`, eine Klasse für Dreiecke, `Rectangle`, eine Klasse für Rechtecke, `Transform`, eine Klasse, die Transformationen auf Graphikobjekten ausführen kann und `Point2D`, eine Klasse, um Punkte im 2-dimensionalen Raum, die von den anderen Klassen verwendet werden können, zu modellieren. Erzeugen Sie die Klassen anhand des in Abbildung ?? aufgezeigten UML-Diagramms und erzeugen Sie daraus die `polylib` (ähnlich der `imagelib`). Alle Klassen müssen dem namespace `POLYLIB` zugeordnet werden.
- Implementieren Sie alle Klassen und deren Eigenschaften bis auf die `render()`-Methode. Im SVN ist unter `solution/Aufgabe12` die Datei `mainA12.cpp` eingchecked. Diese testet die Eigenschaften der Klasse und das Interface. *Kopieren Sie diese Datei in den Ordner Ihres Projektes Aufgabe12.* Fügen Sie `mainA12.cpp` als einzige Datei dem Projekt Aufgabe12 hinzu und testen Sie Ihre Implementierung.

### Aufgabe 13 (H) 2D Objekte anzeigen und manipulieren

Lesen Sie sich im Red Book (Link auf der Homepage des Graphikprogrammierpraktikums) das Kapitel 2, *Drawing Geometric Objects* durch. Ziel dieser Aufgabe ist, ein Dreieck und ein Rechteck in einem Fenster nebeneinander darzustellen und Transformationen auf ihnen auszuführen. Um die Transformationen auszuführen, können Sie entweder die von Ihnen in der vorherigen Aufgabe programmierte `Transform`-Klasse verwenden oder die von OpenGL angebotenen Funktionen `glRotatef`, `glTranslatef` und `glScalef`. In `solution/Aufgabe13` befindet sich eine Datei `mainA13.cpp`. *Kopieren Sie diese Datei in den Ordner Ihres Projektes Aufgabe13.* Fügen Sie `mainA13.cpp` in Ihr Projekt Aufgabe13 ein. Diese Datei stellt eine kleine Benutzeroberfläche zur Verfügung (dafür muss `glut`, das Open GL Utility Toolkit, installiert sein. Man kann es von <http://www.opengl.org/resources/libraries/glut.html> herunterladen).

- Erweitern Sie nun die `Render`-Methoden der beiden Klassen `Triangle` und `Rectangle` so, dass ein Dreieck und ein andersfarbiges Rechteck gezeichnet werden. Testen Sie die `Render`-Methoden mit der `main`-Methode aus `mainA13.cpp`

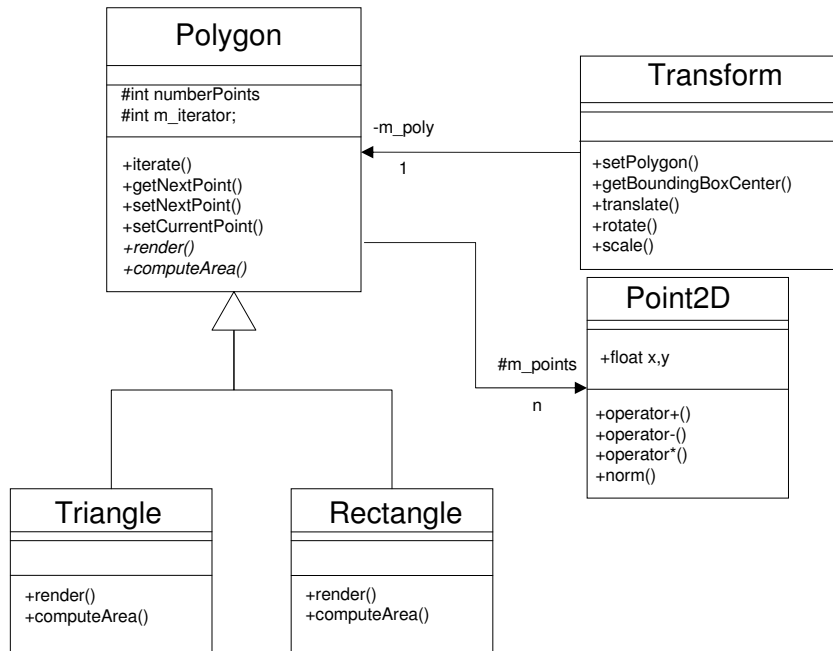


Abbildung 1: UML-Klassendiagramm

- b) Erweitern Sie die Methode `rotate(int reverse)` der Datei `mainA13.cpp` so, dass das in dieser Datei definierte `Triangle` im Uhrzeigersinn und das `Rectangle` gegen den Uhrzeigersinn um 5 Grad rotiert, falls `reverse = 1`, bei `reverse = -1` soll sich die Drehrichtung umkehren. `mainA13.cpp` wurde so programmiert, dass Sie diese Funktion testen können, indem Sie in dem Fenster die Maus mit dem gedrückten linken Mausknopf nach oben oder unten bewegen.
- c) Erweitern Sie die Methode `scale(int reverse)` der Datei `mainA13.cpp` so, dass sich die in dieser Datei definierten Polygone vergrößern, falls `reverse = 1`, bzw. verkleinern, falls `reverse = -1`. `mainA13.cpp` wurde so programmiert, dass Sie diese Funktion testen können, indem Sie in dem Fenster die Maus mit dem gedrückten rechten Mausknopf nach oben oder unten bewegen.
- d) Erweitern Sie die Methode `reset()` so, dass das original Polygon wieder hergestellt wird. Testen Sie die Methode, indem Sie den "r"-Knopf auf der Tastatur drücken.
- e) Sie können nun Objekte auf einer 2D Fläche transformieren. Programmieren Sie eine Funktion `doWeirdThingsWithPolygons()`, die verschiedene Polygone in einem Fenster erzeugt, und diese zufällig transformiert. Man könnte beispielsweise hüpfende Polygone programmieren. Sie können diese Funktion ausführen, indem Sie den "w"-Knopf der Tastatur drücken.