Ludwig-Maximilians-Universität München
LFE Medieninformatik
3D Programmierpraktikum
Dipl. Inf. Otmar Hilliges

Sommersemester 2007
Merkblatt 1
24. April 2007

Merkblatt zu Scheinkriterien und Versionskontrolle mit SVN

Termine:

Dienstag, 12:00 Uhr – Neues Aufgabenblatt online
Mittwoch, 10:15 - 11:45 Uhr – Amalienstr. 17, Raum 107
Montag, 12:00 Uhr – Abgabe der Aufgaben per SVN

Erstes Treffen am 25.April 2007.

Anforderungen:
Zur Teilnahme an dem Programmierpraktikum 3D Grafik sind die Kenntnisse aus dem Grundstudium zwingend nötig. In dem Praktikum werden die Grundzüge der Programmiersprache C++ erläutert - nicht jedoch Basiswissen zur Objektorientierten Programmierung, dieses wird vorrausgesetzt.

Übungsaufgaben und Scheinvergabe:
Aufgabenblätter werden Dienstag vor der jeweiligen Besprechung auf der Internetseite (`http://www.medien.ifi.lmu.de/lehre/ss07/3dp/`) zur Verfügung gestellt. Die Lösungen der Aufgabenblätter sind jeweils zum darauf folgenden Montag um **12:00 Uhr s.t.** abzugeben (per SVN). In Sonderfällen ist der auf dem Blatt genannte Termin bindend.

Für die Bearbeitung der Aufgaben steht der CIP-Raum in der Amalienstraße zur Verfügung. Bei der Abgabe der Aufgaben ist zu beachten, dass Plattform unabhängig programmiert wird, d.h. Die Abgaben müssen sowohl mit Microsoft VS kompilierbar sein als auch unter Linux mit der GNU Compilersuite.

Die Übungsaufgaben sind soweit nicht anders angegeben selbstständig und individuell abzugeben. In der Projektphase wird es Gruppen geben.

Übungsaufgaben werden nur dann als korrekt bewertet, wenn sie *kompilieren*, *funktionieren*, ausreichend dokumentiert und nicht abgeschrieben sind. Während des ganzen Semesters müssen alle Aufgaben abgegeben werden, darüber hinaus muss am Ende des Semesters eine Gruppenarbeit abgegeben und präsentiert werden.

Einen Schein gibt es, wenn *alle* Übungsaufgaben abgegeben und als korrekt bewertet wurden. In Einzelfällen behält sich die Übungsleitung vor auch noch eine zusätzliche mündliche Prüfung am Ende des Semesters durchzuführen. Die Scheine werden benotet wobei sich die Note aus den individuellen Abgaben und der Gruppennote zusammen setzt.

Bitte wenden!

Infrastruktur:
Zur Teilnahme ist eine Anmeldung unter folgender Addresse nötig:

- `https://www.medien.ifi.lmu.de/lehre/anmeldung.php?v=3dp`

Für die Kommunikation der Praktikumsteilnehmer untereinander und mit der Übungsleitung steht eine Mailingliste zur Verfügung

- `https://tools.rz.ifi.lmu.de/mailman/listinfo/3dpss07`
- Mails an `mailto:3dpss07@lists.ifi.lmu.de`

die Teilnehmer-Liste soll auch der Diskussion von auftretenden Problemen, Lösungsstrategien usw. dienen.

Zur Code-Verwaltung wird ein SVN Server zur Verfügung gestellt. Die Verwendung ist verpflichtend und die Abgabe erfolgt per SVN commit. Ein Tutorial zu SVN findet man z.B. unter der Adresse `http://svnbook.red-bean.com/`.

Die Adresse für das Repository ist

- `svn://svn.medien.ifi.lmu.de/ss07/3d-prakt`.

Das Passwort und den Benutzernamen bekommen sie nach abgeschlossener Anmeldung per e-mail zugesandt.

Abwesenheit:
Die wöchentlichen Vorbesprechungen sind Pflichtveranstaltungen. Wenn Sie wegen Krankheit oder einem anderen wichtigen Grund nicht an einer wöchentlichen Sitzung teilnehmen können, melden Sie sich bitte *vorher* per E-Mail oder telefonisch ab.

Versionskontrolle mit subversion (SVN) SVN steht für subversion und ist ein frei erhältlicher Standard für Versionskontrolle. Versionskontrolle ist notwendig, wenn mehrere Programmierer an einem System arbeiten und gleichzeitig Dateien editieren und verändern. Dann muss ein konsistenter Zustand des Gesamtsystems aufrechterhalten werden, wobei SVN behilflich sein kann. Das SVN verfügt über ein gemeinsames Repository, einem großen Pool, wo alle Daten liegen. Das Repository ist in Module aufgeteilt, die abgegrenzte Einheiten darstellen und sich gegenseitig nicht beinflussen sollten. Ein Modul könnte beispielsweise den Quellcode für ein System beinhalten. Wenn nun ein Programmierer an dem System etwas verändern will, kann er sich die Dateien vom Repository auf seinen lokalen Rechner kopieren, er erstellt sich eine sogenannte Working Copy. In SVN-Sprache macht der Programmierer dann einen checkout. Seine kopierte Version kann er nun beliebig verändern; wenn er mit seinen Änderungen fertig ist (diese sollten den Code nach Möglichkeit immer lauffähig halten) kann er seine Version des Codes wieder in das Repository zurückschreiben. In SVN-Sprache sagt man, der Programmierer macht einen commit. Das SVN prüft jetzt, ob auch noch andere Programmierer dieselben Stellen im Code verändert haben und versucht einen konsistenten Zustand herzustellen. Falls dies nicht möglich ist, gibt es einen Konflikt und die Programmierer oder ein Versionsverwalter muss ihn manuell lösen. Falls ein Modul schon ausgecheckt ist und wieder daran gearbeitet werden soll, sollte vorher ein update gemacht werden, um die Version des Moduls auf den aktuellsten Stand zu bringen. Im folgenden werden alle relevanten SVN-Befehle kurz beschrieben. Im Linux-svn client können diese Befehle direkt nach dem executable `svn` geschrieben werden. Für Windows gibt es komfortable GUI Frontends z.B. (`http://tortoisesvn.tigris.org/`).

- `checkout <Modulname>`: Erzeugt eine working copy auf dem lokalen Rechner.
- `update <Modulname | Dateiname | Ordner >`: Bringt die Datei/das Modul auf den aktuellen Stand wie er im Repository abgespeichert ist. Falls die eigene working copy aktueller als die Dateien im Repository sind, werden diese Änderungen nicht gelöscht.
- `add <Ordner | Datei>`: Fügt dem Repository eine Datei hinzu (nachher **muss** svn commit ausgeführt werden).
- `remove <Datei>`: Löscht eine Datei aus dem Repository (dafür darf sie in der working copy nicht mehr vorhanden sein, nachher immer svn commit ausführen).
- `commit <Datei | Ordner | Modul>`: Schreibt alle Änderungen von der lokalen working copy ins Repository (dadurch werden sie für andere zugänglich).

Die Dateien müssen so in das auf den SVN Server eingecheckt sein, dass wir mit einem einzigen Klick das Programm kompilieren können. Bitte also die .sln, .vcproj, .cpp und .h Dateien einchecken, jedoch keine anderen! Insbesondere keine Binärdateien (.exe) oder object-files (.o, .obj etc.) einchecken.

Jeder Teilnehmer sollte ein neues Verzeichnis mit seinem <Nachnamen> anlegen, in diesem Verzeichnis sollen dann die Aufgaben und die dazugehörigen Dateien in Unterordnern liegen.