Kurs Programmieren 1 / Java

Projekt

Prof. Marcus Barkowsky, Prof. Benedikt Elser

Zusammenfassung

Projekt, Object-oriented Programming

Projektbeschreibung

Einleitung

Im Fach "Programmieren 1" wurden Ihnen viele Kompetenzen in der Objektorientieren Programmierung vermittelt und Sie haben sich diese durch Papier- und Rechnerübungen, sowie Hausaufgaben angeeignet und vertieft. Nun sind Sie bereit, ein erstes eigenes Projekt anzugehen, in welchem Sie sich selbst Ziele setzen und diese Ziele auch durch Ihre neuen Kompetenzen erreichen. Die vorliegende Beschreibung gibt dabei einen sehr lockeren Rahmen vor, in dem Sie sich frei entfalten können.

Randbedingungen

Das Projekt beginnt in der Woche vom 7.Dezember und endet am 17.Januar. In dieser Zeit gibt es insgesamt drei betreute Kurstermine, in denen wir Ihnen gern zur Verfügung stehen. Der Umfang des Projekts liegt bei ca. 18 Stunden pro Person. In der Woche vom 18. Januar findet dann eine Peer-Review Beurteilung statt, wie Sie sie bereits von den Hausaufgaben gewohnt sind.

Die Projektarbeit soll in Zweiergruppen durchgeführt werden. Sie sind selbst für die Gruppeneinteilung und die Aufteilung der Arbeit verantwortlich, allerdings soll Ihre Dokumentation (siehe unten) die jeweilige Aufgabenaufteilung enthalten, welche auch stichprobenartig überprüft werden kann.

Aufgabenbeschreibung

Die Projektaufgabe besteht darin, ein einfaches Spiel zu implementieren. Die Wahl des Spiels bleibt Ihnen überlassen, beachten Sie jedoch, dass sich im Rahmen von 36 Stunden Arbeitszeit nur sehr begrenzte Spielideen auch umsetzen lassen. Lassen Sie sich hierzu von uns beraten, bevor Sie sich zu viel vornehmen und

planen Sie die Umsetzung in Etappen und mit solcher Priorisierung, dass Sie auf jeden Fall die notwendigen Bedingungen erfüllen.

Sie programmieren ein Spiel für ein sehr stark eingeschränktes Display. Dieses enthält nur 24x48 Bildpunkte (Pixel), d.h. 24 Reihen mit jeweils 48 Spalten. Jeder Bildpunkt kann 16 Millionen Farben annehmen, wobei die Rot, Grün und Blau-Komponente mit jeweils einem Byte angesprochen wird (wie üblich in der Bilddarstellung). Als Steuermöglichkeit stehen Ihnen vier Tasten zur Verfügung, die wie im Cursorblock üblich angeordnet sind. Es gibt nur einen Spieler. Die Zeit für eine Spielrunde sollte bei 20-30 Sekunden liegen.

Zur Ein- und Ausgabe erhalten Sie eine Klasse mit zwei Methoden:

- `public int getKeyboard()`: Liefert für die 4 Cursortasten der Tastatur folgende Werte zurück: 0 für "hoch", 1 für "runter", 2 für "links" und 3 für "rechts", -1 für keine Taste gedrückt
- `public void showImage(short[] image)`: Zeigt ein komplettes Bild auf dem Display an, wobei der erste Wert des Arrays die Rot-Komponente des linken oben Bildpunkts ist und der letzte Wert die Blau-Komponente von 0 bis 255 des rechten unteren Bildpunktes. Das übergebene Array muss exakt `24*48*3` Elemente haben für die 24 Zeilen, 48 Spalten und 3 Farbkomponenten pro Pixel. Das Display wird zeilenweise durchlaufen.

Ihr Spiel soll umfassen:

- Eine "interaktive Spielerfigur", die mit den Tasten gesteuert werden kann. Es müssen nicht alle Tasten verwendet werden und die Tasten können auch anders als oben beschrieben verwendet werden.
- Eine "automatisch gesteuerte Spielerfigur", die vom Programm bewegt wird. Diese Bewegung kann auch zufällig sein.
- Einen "Hintergrund", der das Spielfeld darstellt
- Ein Score-System, welches für bestimmte Aktionen eine Erhöhung des Score-Wertes anzeigt
- Ein Highscore-System, welches die Scores von mehreren Runden anzeigt.

Bitte beachten Sie, dass eine minimale Umsetzung akzeptabel ist. So wäre eine rote Spielfeldbegrenzung unter Einsatz der äußeren LEDs ein Hintergrund, ein sich willkürlich bewegender blauer Punkt eine "automatisch gesteuerte Spielfigur" und ein weiterer grüner Punkt, der sich über die Cursor-Tasten bedienen lässt eine "interaktive Spielerfigur". Für das Bewegen der Spielerfigur wird der Score um 10 erhöht und auf der Konsole ausgegeben, für das Berühren des blauen Punktes gibt es 100 Punkte. Nach 100 Bewegungen der "automatisch gesteuerten Spielerfigur" sind ca. 20 Sekunden erreicht und eine neue Runde wird gestartet, wobei der erreichte Score nun für die Dauer der Programmlaufzeit mit in einer sortierten Liste angezeigt wird.

Ihre Implementierung soll beinhalten:

- Eine generische Klasse für Spielobjekte

- Drei davon abgeleitete Klassen, eine für den “Spieler-Charakter”, eine für die “automatisch gesteuerte Spielfigur” und eine für den “Hintergrund”
- Eine Klasse zur Manipulation von Score und Highscore

Dokumentation

Zur Dokumentation gehört:

- Spielidee (ca. 0.5-1 DIN-A4 Seite)
- Spielbeschreibung (ca. 1 DIN-A4 Seite)
- Eine Kurzfassung von Spielidee und Spielbeschreibung soll in Form einer JPEG-Grafik in der Auflösung 1920x1080 erstellt werden - als elektronische Spielanleitung parallel zur LED-Wand.
- Klassendiagramm (ca. 1 DIN-A4 Seite)
- Kurze Beschreibung des Programmablaufs (ca. 0.5 DIN-A4 Seiten)
- Tabellarische Aufstellung der Arbeitsaufteilung mit geleisteten Stunden im Team (stärkere Ungleichverteilungen sollten kurz begründet werden, ca. 0.5 DIN-A4 Seite).
- Persönliche Kommentare zum Projekt und Zusammenfassung (ca. 0.5 DIN-A4 Seite)
- Programmcode, textuell im Anhang (“Ausdruck”) und als ZIP oder JAR Datei

Bitte beachten Sie, dass Sie die Dokumentation fortlaufend schreiben müssen: Spielidee und Spielbeschreibung stehen bereits am Anfang fest, die tabellarische Aufstellung der geleisteten Stunden müssen Sie mitprotokollieren. Die Form der Dokumentation ist weniger relevant als der Inhalt, d.h. zum Beispiel: ein Smartphone-Photo eines guten handgezeichneten Klassendiagramms ist besser als ein in draw.io gezeichnetes unvollständiges Klassendiagramm.

Hinweise

- Es wird Ihnen ein Beispielprojekt zur Verfügung gestellt, das ein Spielfeld anzeigt und das mit den Cursortasten navigierbar ist
- Das Beispielprojekt läuft nur mit Java 8. Bitte achten Sie auf eine entsprechende Konfiguration