ADAP CW#06

Projektdaten

Projekt name:	3D-Printing
Projekt repository:	https://github.com/**ModischFabrications**/wahlzeit
This week's tag:	adap-hw05 on master
Homework diff:	https://github.com/ModischFabrications/wahlzeit/compare/adap-hw04...adap-hw05
CI:	https://travis-ci.org/**ModischFabrications**/wahlzeit
Docker Hub Repo:	https://hub.docker.com/r/**modischfabrications**/wahlzeit

Hausaufgaben

Implementation

Neue Klassen (falls vorhanden)

CartesianCoordinate, SphericCoordinate,

CartesianCoordinateTest, SphericCoordinateTest, CoordinateTest, LocationTest, LocationTestSuite

Veränderte Klassen (falls vorhanden)

Coordinate, Location (+ diversen mit entfernten TODOs, siehe diff),

AllTests

Erklärung

Natürlich habe ich zuerst Tests für alle Klassen geschrieben, um das erwartete Verhalten nachzuweisen, insbesondere die Austauschbarkeit und Konvertierungsfähigkeit. Dabei habe ich mit dem Hinzufügen von Tests für die schon bestehenden "Coordinate" und "Location" angefangen, um eine saubere Basis für die dieswöchigen Änderungen zu schaffen.

Dann habe ich die alte "Coordinate" Klasse zu "CartesianCoordiate" umbenannt und ein Interface eingeführt, der diese und die neue "SphericCoordinate" Klassen zusammenfasst. Dann habe ich in "Cartesian*" die neuen Methoden auf die schon implementierten Methoden abgebildet. Methoden, die Spherisch mehr Sinn machen, nutzen die Implementierung von "Spheric", so wird der Implementierungsaufwand für doppelte Funktionen reduziert.

"Spheric" habe ich analog zu "Cartesian" implementiert, als das ich auf bereits existierende Methoden der "Partnerklasse" umgeleitet habe. Natürlich hat auch diese Klasse eine equals und hashCode Methode bekommen, diese konnten mit IntelliJ einfach generiert werden und wurden dann ergänzt.

Fehlend war dann nur die Konvertierung zwischen den Koordinatensystemen und die Berechnung

des Zentriwinkels, hierfür habe ich mich an den im Code benannten Formeln orientiert. Dabei habe ich gleitkommaoptimierte Formel mit der *law of haversines* verwendet, die auch bei kleinen Differenzen präzise Rückgaben liefern kann.

Dann habe ich noch alte FIXMEs und TODOs aus dem bestehenden Code entfernt, da diese Komponenten hoffentlich niemals von mir angefasst werden und ich so die für mich relevanten Hints leichter wiederfinde.

Zuletzt habe ich alle Änderungen gepusht und den letzten Commit mit dem oben genannten Tag für die Abgabe markiert.

Fragen

How did you implement the equality check and why?

Ich habe die Gleichheitsprüfung nur einmal in “Cartesian” implementiert und von “Spheric” darauf referenziert, um den Implementierungsaufwand zu minimieren. Da die reale Version der Koordinaten hinter dem Interface versteckt ist und somit identisch behandelt wird muss ich den Typ der Koordinate nicht explizit überprüfen.

Die Gleichheitsprüfung von “Cartesian” habe ich weitgehend von den letzten Aufgaben übernommen, wichtig ist vor allem die Prüfung auf “Ähnlichkeit”, um Gleitkommafehler der Konvertierung auszugleichen.