Technology
Arts Sciences
TH Köln

Fakultät für Informations-, Medien- und Elektrotechnik
Institut für Nachrichtentechnik

Prof. Dr. Hans W. Nissen
Email: hans.nissen@th-koeln.de

Software Engineering

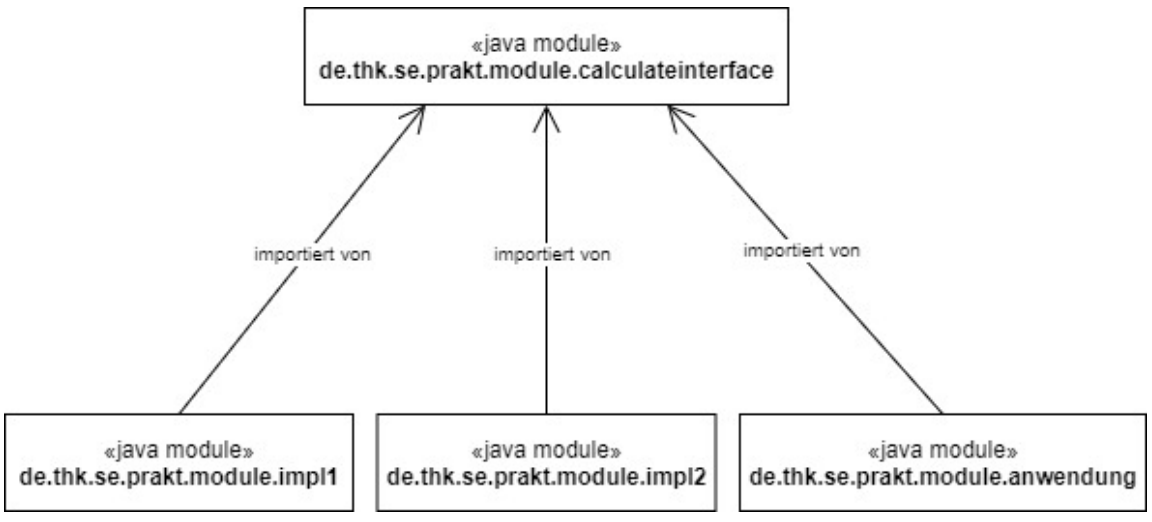
Praktikumsversuch 3, Gruppe E
- Hausaufgaben -

Ziel: Realisierung einer losen Kopplung von Java-Modulen

Abgabe der Lösungen: Bis zum 18.01., 08:00 Uhr morgens, im Master-Branch Ihres Gitlab-Repositories P3SE<IhreTeilnehmernummer> (z.B. https://gitlab.nt.fh-koeln.de/gitlab/se/SE29/P3SE29.git für Teilnehmer 29). Abzugeben ist das vollständige IntelliJ-Projekt.
In Ihrem Gitlab-Repository zu diesem Praktikumsversuch habe ich Ihnen ein IntelliJ-Projekt zur Verfügung gestellt, welches Sie als Grundlage Ihrer Implementierung verwenden müssen.

Es soll eine Systemarchitektur aus insgesamt 4 Java-Modulen aufgebaut werden (siehe grafische Darstellung unten):

- Modul `de.thk.se.prakt.module.calculateinterface` enthält die Schnittstellen-Klasse `CalculateArray` und exportiert diese.
- Modul `de.thk.se.prakt.module.impl1` importiert das Modul `de.thk.se.prakt.module.calculateinterface`, enthält eine Implementierung der Schnittstellen-Klasse `CalculateArray` und veröffentlicht diese – besitzt aber keinen Export der Implementierung.
- Modul `de.thk.se.prakt.module.impl2` importiert das Modul `de.thk.se.prakt.module.calculateinterface`, enthält eine weitere Implementierung der Schnittstellen-Klasse `CalculateArray` und veröffentlicht diese – besitzt aber keinen Export der Implementierung.
- Modul `de.thk.se.prakt.module.anwendung` importiert das Modul `de.thk.se.prakt.module.calculateinterface` und greift auf beide Implementierungen der Schnittstellen-Klasse `CalculateArray` über lose Kopplung zu.



Technology
Arts Sciences
TH Köln

Fakultät für Informations-, Medien- und Elektrotechnik
Institut für Nachrichtentechnik

Prof. Dr. Hans W. Nissen
Email: hans.nissen@th-koeln.de

Das Modul `de.thk.se.prakt.module.calculateinterface` mit der Schnittstellen-Klasse `CalculateArray` ist bereits im gegebenen IntelliJ-Projekt implementiert. Die Schnittstellen-Klasse `CalculateArray` besitzt folgende Methoden:

- `calculate(original:int[]):int[]` – diese Methode nimmt als Parameter ein Integer-Array und führt auf jedem Wert dieses Arrays eine Berechnung durch (z.B. Berechnung des ggT zu einer vorgegebenen Zahl) und liefert als Ergebnis ein neues Integer-Array. Bei dieser Methode darf das übergebene Integer-Array `original` nicht verändert werden, d.h. das Ergebnis der Methode muss ein neues Array-Objekt sein. Hinweis: Die statische Methode `copyOf(int[] original, int newLength)` der Klasse `Arrays` liefert eine Kopie des Arrays `original` mit der Länge `newLength`.
- `getImplProperty():String` – Da grundsätzlich mehrere Berechnungsmöglichkeiten existieren liefert diese Methode im Ergebnis-String die Berechnungsart der konkreten Implementierung der Methode `calculate`. Durch Verwendung dieser Methode kann die Anwendung die verfügbaren verschiedenen Implementierungen der Methode `calculate` bzgl. der Berechnungsart unterscheiden und die gewünschte Implementierung auswählen.

H 3.1 Modul `de.thk.se.prakt.module.impl1` realisieren

Erstellen Sie das Modul `de.thk.se.prakt.module.impl1` und realisieren Sie dort eine Implementierung der Schnittstelle `CalculateArray`, welche für jeden Eintrag des übergebenen Arrays den größten gemeinsamen Teiler (ggT) zur Zahl 8 berechnet und diese wieder in einem Array als Ergebnis liefert..
Die Methode `getImplProperty()` soll hierbei den String „`ggT`“ als Ergebnis liefern. Das Modul soll Pakete entsprechend der typischen Namensgebung enthalten.
Dieses Modul soll die Implementierung der Schnittstellen-Klasse für die lose Kopplung anbieten.

H 3.2 Modul `de.thk.se.prakt.module.impl2` realisieren

Erstellen Sie das Modul `de.thk.se.prakt.module.impl2` und realisieren Sie dort eine Implementierung der Schnittstelle `CalculateArray`, welche für jeden Eintrag des übergebenen Arrays das kleinste gemeinsame Vielfache (kgV) zur Zahl 5 berechnet und diese wieder in einem Array als Ergebnis liefert..
.
Die Methode `getImplProperty()` soll hierbei den String „`kgV`“ als Ergebnis liefern. Das Modul soll Pakete entsprechend der typischen Namensgebung enthalten.
Dieses Modul soll die Implementierung der Schnittstellen-Klasse für die lose Kopplung anbieten.

H 3.3 Modul `de.thk.se.prakt.module.anwendung` realisieren

Erstellen Sie das Modul `de.thk.se.prakt.module.anwendung` mit der Klasse `AnwendungsKlasse`, welche eine main-Methode besitzt. Das Modul soll Pakete entsprechend der typischen Namensgebung enthalten.

Hinweis: Die exakte und korrekte Umsetzung dieses Ablaufs ist erforderlich für das Bestehen dieses Blattes. Beispielsweise dürfen Sie sich bei der Realisierung der Schritte 3 und 4 nicht auf die zufällige Sortierung der Rückgabe des `ServiceLoaders` verlassen! Stattdessen müssen Sie das Finden der korrekten Implementierung korrekt programmieren.

Setzen Sie in dieser main-Methode den folgenden Ablauf um:

1. Das Beispiel-Array [64, 112, 46, 9, 30, 23, 33, 440] erzeugen.

Technology
Arts Sciences
TH Köln

Fakultät für Informations-, Medien- und Elektrotechnik
Institut für Nachrichtentechnik

Prof. Dr. Hans W. Nissen
Email: hans.nissen@th-koeln.de

2. Alle Implementierungen für die Schnittstellenklasse per loser Kopplung holen.
3. Implementierung für den größten gemeinsamen Teiler finden.
4. Implementierung für das kleinste gemeinsame Vielfache finden.
5. Die ggT-Berechnung für das Beispiel-Array durchführen und das Ergebnis auf der Console ausgeben: „ggT: <alle Werte in einer Zeile hintereinander, jeweils mit Komma (,) getrennt>“.
6. Die kgV-Berechnung für das Beispiel-Array durchführen und das Ergebnis auf der Console ausgeben: „kgV: <alle Werte in einer Zeile hintereinander, jeweils mit Komma (,) getrennt>“.
7. Das Beispiel-Array auf der Console ausgeben: „Das Original-Array: <alle Werte in einer Zeile hintereinander, jeweils mit Komma (,) getrennt>“.