

## Die Migration von Java-Applikationen

Mit der Umstellung auf das Modulsystem wird sicherlich sowohl von Entwickler als auch Betreiber erwartet, dass irgendwann alle Java-Applikationen modular vorliegen, sodass auch eigene Applikationen modularisiert werden sollten. Dies wird allerdings seine Zeit brauchen. Die von Oracle vorgenommene Modularisierung des JDK hat ja auch fünf Jahre lang angedauert.

Obwohl mit Java 9 der Modul-Path neu eingeführt wurde, ist es weiter möglich, für das Auffinden von Komponenten in einer Applikation den Classpath zu benutzen, sodass weiterhin sowohl JAR-Dateien als auch einfache `.class`-Dateien darauf abgelegt werden können.

So kann in einer reinen Plattform-Migration zunächst eine bestehende Java-9-Anwendung einfach auf dem Classpath bleiben. Diese Vorgehensweise nutzt keine Module (außer die System-Module der Java-Plattform selbst, wie z.B. `java.base`).

Wie in der Java-Literatur beschrieben, wird die Migration von Anwendungen durch verschiedene Kategorien von Modulen unterstützt. Diese werden zunächst in »Unnamed Modules« und »Named Modules« aufgeteilt. Wie vom Namen her zu erwarten ist, besitzen Letztere einen Modulnamen. Sie werden ihrerseits in drei weitere Gruppen aufgeteilt: Explicit Modules, Open Modules und Automatic Modules.

Wird der Classpath benutzt, werden all seine Inhalte automatisch zu einem »Unnamed Module« zusammengefügt. Dieses beinhaltet alle Typen, die auf dem Classpath abgelegt sind, hat »Reads«-Beziehungen auf alle Module eines eventuell zusätzlich benutzten Modul-Path und exportiert sämtliche Packages all seiner Klassen und JAR-Dateien.

Explicite Modules besitzen eine `module-info`-Datei und benennen darin ihre »Reads«-Abhängigkeiten sowie die »Exports« für ihre Packages, wie dies mit den Beispielen aus dem Kapitel 12 gezeigt wird. Automatic Modules haben »Reads«-Beziehungen zu allen anderen Modulen, sodass sie sämtliche exportierten Packages von Explicit Modules benutzen können. Gleichzeitig werden alle ihre Packages automatisch exportiert. Open Modules sind den Explicit Modules ähnlich und exportieren zusätzlich ihre sämtlichen Packages zur Laufzeit für Reflection.

Unbenannte und automatische Module dienen der Übergangsphase zur Modularisierung. Automatische Module entstehen, wenn wir JAR-Dateien von nicht modularen Komponenten auf den Module-Path legen. Dann wird für jede nicht modulare JAR-Datei, wie die weiteren Beispiele zeigen werden, ein Modul erzeugt, dessen Name aus dem der JAR-Datei abgeleitet wird. Für das Anzeigen von Abhängigkeiten zwischen diesen Modulen und ihren Packages kann das `jdeps`-Tool benutzt werden und für das Erstellen der zugehörigen `module-info`-Dateien das `jar`-Kommando: `jar dateiname.jar --describe-module`.

Als einfaches Beispiel soll die Java-Applikation aus der Aufgabe 12.1 erstmals in ein JAR-File (mit oder ohne Manifest-Datei) gepackt und danach nach Java 9 migriert werden.

Dafür erstellen wir in einem Verzeichnis `jarlib` (das mit `mkdir` im Arbeitsverzeichnis angelegt wurde) eine Manifest-Datei `Manifest.txt` mit dem nachfolgenden Inhalt:

```
Manifest-Version: 1.0
Main-Class: ListSetMapFactoryMethoden.class
```

Achten Sie darauf, dass die leere Zeile am Ende der Datei nicht vergessen wird (darum die Enter-Taste drücken).

Mit `jar -cvfm factorymethoden.jar jarlib\Manifest.txt ListSetMapFactoryMethoden.class` erstellen wir ein JAR-Archiv mit dem Namen `factorymethoden.jar` und der angegebenen Manifest-Datei. Dieses JAR-Archiv kann bis einschließlich Java 8 auf dem Klassenpfad abgelegt werden. Dabei wird es nicht modularisiert und die Anwendung kann wie folgt gestartet werden:

```

C:\Users\Lissi\Documents\java9\jarlib>cd ..

C:\Users\Lissi\Documents\java9>jar -cvfm factorymethoden.jar jarlib\Manifest.txt
ListSetMapFactoryMethoden.class
Manifest wurde hinzugef"gt
ListSetMapFactoryMethoden.class wird hinzugef"gt(ein = 4160) (aus = 1949)(53 % u
erkleinert)

C:\Users\Lissi\Documents\java9>java --class-path factorymethoden.jar ListSetMapF
actoryMethoden
[java 5, java 6, java 7, java 8]
[java 5, java 6, java 7, java 8]
Leere Liste: []
Immutable-Liste mit Java 9: [java 7, java 8, java 9]
Immutable-Liste mit Java 9:
1 2 3 4 5 6 7 8 9
Immutable-Set mit Java 9: [java 7, java 8, java 9]
Immutable-Map mit Java 9 und Map.of(): {9=java 9, 8=java 8, 7=java 7}
Immutable-Map mit Java 9 und Map.ofEntries(): {9=java 9, 8=java 8, 7=java 7}
Immutable-Map mit Java 9 und Map.ofEntries(): {9=java 9, 8=java 8, 7=java 7}

C:\Users\Lissi\Documents\java9>java --module-path factorymethoden.jar --module L
istSetMapFactoryMethoden
Error occurred during initialization of boot layer
java.lang.module.FindException: Unable to derive module descriptor for factoryme
thoden.jar
Caused by: java.lang.module.InvalidModuleDescriptorException: ListSetMapFactoryM
ethoden.class found in top-level directory (unnamed package not allowed in modul
e)

```

```

C:\Users\Lissi\Documents\java9>jar --file factorymethoden.jar -d --describe-modu
le
Moduldescriptor kann nicht abgeleitet werden f"r: factorymethoden.jar
ListSetMapFactoryMethoden.class found in top-level directory (unnamed package no
t allowed in module)

C:\Users\Lissi\Documents\java9>jdeps factorymethoden.jar
factorymethoden.jar -> java.base
<unnamed>                -> java.io
<unnamed>                java.base -> java.lang
<unnamed>                java.base -> java.lang.invoke
<unnamed>                java.base -> java.util
<unnamed>                java.base -> java.util.function
java.base
C:\Users\Lissi\Documents\java9>_

```

Soll die JAR-Datei für eine Migration der Applikation auf den Modul-Path gelegt werden, muss, wie der angezeigten Fehlermeldung zu entnehmen ist, die Java-Klasse in einem untergeordneten Verzeichnis zum Arbeitsverzeichnis liegen. Unnamed Packages können nicht in Module übernommen werden:

```

C:\Users\Lissi\Documents\java9\jarlib>jar -cvfm listsetmap.jar Manifest.txt List
SetMapFactoryMethoden.class
Manifest wurde hinzugef"gt
ListSetMapFactoryMethoden.class wird hinzugef"gt(ein = 4167) (aus = 1954)(53 % u
erkleinert)

C:\Users\Lissi\Documents\java9\jarlib>java --module-path listsetmap.jar --module
ListSetMapFactoryMethoden
Error occurred during initialization of boot layer
java.lang.module.FindException: Unable to derive module descriptor for listsetma
p.jar
Caused by: java.lang.module.InvalidModuleDescriptorException: ListSetMapFactoryM
ethoden.class found in top-level directory (unnamed package not allowed in modul
e)

```

Darum erstellen wir für den Test in unserem Arbeitsverzeichnis das Verzeichnis `listsetmap/list/set/map` und kopieren die Java-Datei `ListSetMapFactoryMethoden.java`, der die `package list.set.map`-Anweisung hinzugefügt wurde, in dieses Verzeichnis. Wie auch beim vorangegangenen Test legen wir nach dem Übersetzen der so abgeänderten Java-Klasse (im `listsetmap`-Verzeichnis mit `javac list.set.map.ListSetMapFactoryMethoden.java`) im `listsetmap`-Verzeichnis das Verzeichnis `jarlib` an und erstellen darin die Manifest-Datei `Manifest.txt` mit dem Inhalt:

```
Manifest-Version: 1.0
Main-Class: list.set.map.ListSetMapFactoryMethoden.class
```

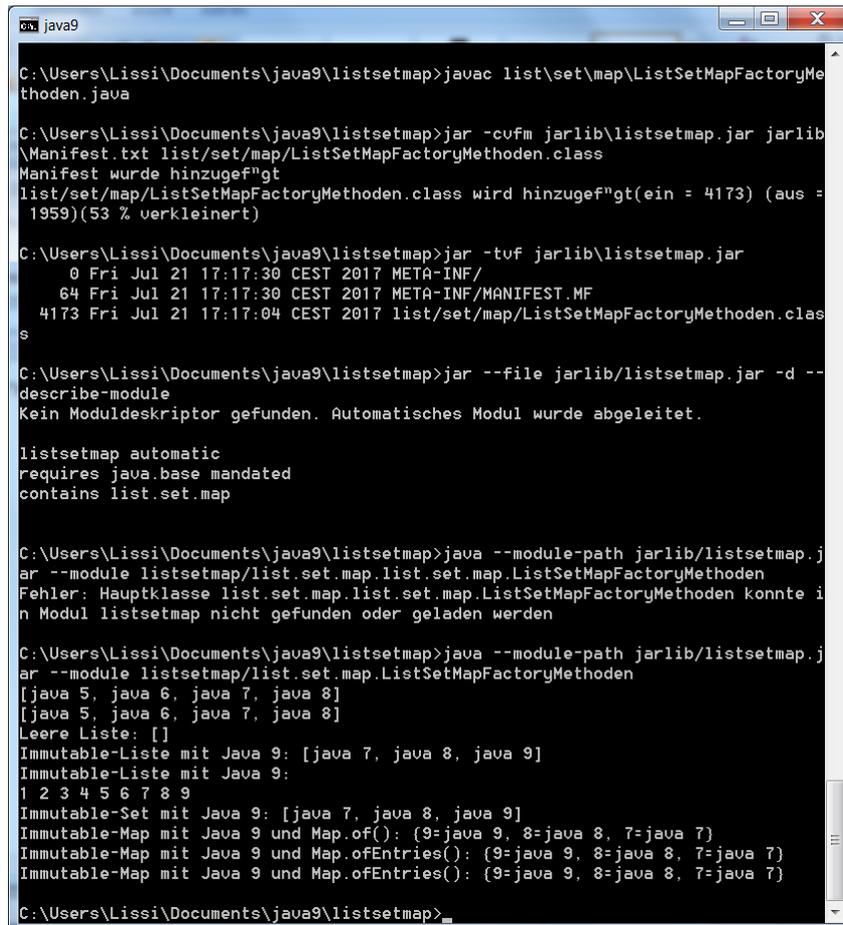
die für das Erzeugen eines JAR-Archivs `listsetmap.jar` benötigt wird:

```
jar -cvfm jarlib\listsetmap.jar jarlib\Manifest.txt
list/set/map/ListSetMapFactoryMethoden.class
```

Dieses kann mit: `jar -tvf jarlib\listsetmap.jar` angezeigt werden.

Dann führt der nachfolgende Programmaufruf (damit wird die JAR-Datei auf den Modul-Path gelegt) zum korrekten Ausführen des JAR-Archivs und seiner Modularisierung:

```
java --module-path jarlib\listsetmap.jar --module
listsetmap/list.set.map.ListSetMapFactoryMethoden
```



```
ca: java9
C:\Users\Lissi\Documents\java9\listsetmap>javac list\set\map>ListSetMapFactoryMethoden.java
C:\Users\Lissi\Documents\java9\listsetmap>jar -cvfm jarlib\listsetmap.jar jarlib\Manifest.txt list/set/map/ListSetMapFactoryMethoden.class
Manifest wurde hinzugef"gt
list/set/map/ListSetMapFactoryMethoden.class wird hinzugef"gt(ein = 4173) (aus = 1959)(53 % verkleinert)
C:\Users\Lissi\Documents\java9\listsetmap>jar -tvf jarlib\listsetmap.jar
  0 Fri Jul 21 17:17:30 CEST 2017 META-INF/
 64 Fri Jul 21 17:17:30 CEST 2017 META-INF/MANIFEST.MF
 4173 Fri Jul 21 17:17:04 CEST 2017 list/set/map/ListSetMapFactoryMethoden.class
C:\Users\Lissi\Documents\java9\listsetmap>jar --file jarlib/listsetmap.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

listsetmap automatic
requires java.base mandated
contains list.set.map

C:\Users\Lissi\Documents\java9\listsetmap>java --module-path jarlib/listsetmap.jar --module listsetmap/list.set.map.list.set.map.ListSetMapFactoryMethoden
Fehler: Hauptklasse list.set.map.list.set.map.ListSetMapFactoryMethoden konnte in Modul listsetmap nicht gefunden oder geladen werden

C:\Users\Lissi\Documents\java9\listsetmap>java --module-path jarlib/listsetmap.jar --module listsetmap/list.set.map.ListSetMapFactoryMethoden
[java 5, java 6, java 7, java 8]
[java 5, java 6, java 7, java 8]
Leere Liste: []
Immutable-Liste mit Java 9: [java 7, java 8, java 9]
Immutable-Liste mit Java 9:
1 2 3 4 5 6 7 8 9
Immutable-Set mit Java 9: [java 7, java 8, java 9]
Immutable-Map mit Java 9 und Map.of(): {9=java 9, 8=java 8, 7=java 7}
Immutable-Map mit Java 9 und Map.ofEntries(): {9=java 9, 8=java 8, 7=java 7}
Immutable-Map mit Java 9 und Map.ofEntries(): {9=java 9, 8=java 8, 7=java 7}
C:\Users\Lissi\Documents\java9\listsetmap>
```

Mit `jar --file jarlib/listsetmap.jar -d --describe-module` kann die Modularisierung der JAR-Datei bestätigt werden. Es wurde kein Moduldeskriptor gefunden und ein automatisches Modul mit den Eigenschaften:

```
listsetmap automatic
requires java.base mandated
contains list.set.map
```

erzeugt. Der Modulname `listsetmap` wird aus dem Namen der JAR-Datei `listsetmap.jar` abgeleitet.

Für das Erzeugen der zugehörigen module-info-Datei kann das mit Java 8 bereits zur Verfügung gestellte Tool Java Dependency Analysis (`jdeps`) benutzt werden. Es wurde eingeführt, um alle statischen Abhängigkeiten einer Bibliothek zu entdecken, und für Java 9 mit weiteren nützlichen Optionen erweitert.

Der Aufruf `jdeps name.jar` zeigt alle Abhängigkeiten, die in einem angegebenen JAR-Archiv zwischen Packages definiert sind, und mit der Option `--generate-module-info` kann automatisch ein Moduldeskriptor für das Archiv erzeugt werden:

```
jdeps --generate-module-info <Output-Verzeichnis> <Liste von .jar-Dateien>:
```

```

C:\Users\Lissi\Documents\java9\listsetmap>jdeps list\set\map\ListSetMapFactoryMethoden.class
ListSetMapFactoryMethoden.class -> java.base
list.set.map                       -> java.io
list.set.map                       -> java.lang
list.set.map                       -> java.lang.invoke
list.set.map                       -> java.util
list.set.map                       -> java.util.function
list.set.map                       -> java.base

C:\Users\Lissi\Documents\java9\listsetmap>javap list\set\map\ListSetMapFactoryMethoden.class
Compiled from "ListSetMapFactoryMethoden.java"
public class list.set.map.ListSetMapFactoryMethoden {
    public list.set.map.ListSetMapFactoryMethoden();
    public static void main(java.lang.String[]);
}
C:\Users\Lissi\Documents\java9\listsetmap>

```

```

C:\Users\Lissi\Documents\java9\listsetmap>jdeps --generate-module-info out\jarlib\listsetmap.jar
writing to out\listsetmap\module-info.java

C:\Users\Lissi\Documents\java9\listsetmap>type out\listsetmap\module-info.java
module listsetmap {
    exports list.set.map;
}
C:\Users\Lissi\Documents\java9\listsetmap>_

```

Anschließend sollte für jedes automatische Modul der Modulkonfigurationsdatei `module-info.java` angepasst werden, das heißt, ggf. `requires`-Direktiven hinzugefügt bzw. gewünschte `exports`, um die interne Implementierung zu verbergen, entfernt werden.

Die Automatic Modules bilden eine wesentliche Brücke zu dem alten Classpath. Im Vergleich zu Explicit Modules ist ein Unnamed Module für jedes Automatic Module sichtbar.

Werden Classpath und Modul-Path gleichzeitig verwendet, wird eine Klasse zuerst auf dem Modul-Path gesucht.

Alle in diesem Buch enthaltenen Beispiele von Java-Applikationen sind und bleiben zunächst auch ohne Modularisierung unter den nachfolgenden Java-Versionen ablaufbar und dienen in erster Linie dazu, die Java-Programmiersprache kennenzulernen. Sollen diese modularisiert werden, kann man selbstverständlich auch den in Unterkapitel 12.9 aufgezeichneten Weg gehen. Dies bedeutet, dass alle Source-Dateien in Packages abgelegt werden, falls dies nicht schon der Fall war, und oberhalb der Package-Directories Modul-Directories eingeführt werden, die diese Packages zusammen mit beschreibenden Deskriptordateien beinhalten.

Um beide Vorgehensweisen alternativ für eine Applikation mit mehreren Komponenten betrachten zu können, greifen wir auf die Applikation aus der Aufgabe 12.4 zurück, die eine `Main`-Klasse `OptionalMethodenJava9` und mehrere Hilfsklassen `Buch`, `BuchListe2`, `BuchListe3`, `BuchListe4` und `Leser` benutzt.

Damit zwischen den Paketen von Modulen eine größere Vielfalt von Abhängigkeiten definiert werden können, benutzen wir die erweiterte Klasse `Buch` aus der Aufgabe 12.5 und erweitern die Klasse `OptionalMethodenJava9` um die Anweisungen:

```

System.out.println("\nDiese wurden wie folgt bewertet: ");
bestellungenl.stream().forEach(s-> s.getBewertungen()
    .stream().forEach(bewertung-> System.out.println(bewertung.toString())));

```

sodass die ebenfalls in der Aufgabe 12.5 definierte Klasse `BuchBewertung` und die Anpassung der Interfaces `Buchliste2`, `Buchliste3` und `Buchliste4` auf den neu hinzugekommenen Konstruktor der Klasse `Buch` für das Übersetzen und Ausführen erforderlich sind. Sie finden alle Source-Dateien inklusive der Deskriptordateien `module-info.java` in einem Projektverzeichnis `java9migration`, das als Unterverzeichnis von `java9` (wie alle Projektverzeichnisse für die Modularisierung) definiert wurde. Weil wir in den Aufgaben aus dem 12. Kapitel alle konstruierten Module mit nur einem Package definiert haben, wollen wir für die Migration auch die Möglichkeit

in Betracht ziehen, dass ein Modul mehrere Packages beinhalten kann.

Es sollen erstmals zwei Module generiert werden `com.java.migration.app` (mit der `Main`-Klasse) und `com.java.migration.buchleser` (mit allen Hilfsklassen und Interfaces), die jeweils ein Package mit gleichem Namen beinhalten und anfangs leere `module-info`-Dateien definieren.

Damit das Kompilieren fehlerfrei vonstattengeht, müssen die Deskriptordateien mit `requires`- und `exports`-Statements erweitert werden:

```
module com.java.migration.app {  
  // Module importieren  
  requires com.java.migration.buchleser;  
}
```

und

```
module com.java.migration.buchleser {  
  exports com.java.migration.buchleser;  
}
```

Erstellen Sie dazu im Verzeichnis `java9migration` eine Kommandodatei `Java9ModularJar` mit dem Inhalt:

```
javac -d mods --module-source-path src src/com.java.migration.app/module-info.java  
src/com.java.migration.app/com/java/migration/app/OptionalMethodenJava9.java  
javac -d mods --module-source-path src src/com.java.migration.buchleser/module-  
info.java src/com.java.migration.buchleser/com/java/migration/buchleser/*.java  
jar --create --file mlib/com.java.migration.app.jar --main-  
class=com.java.migration.app.OptionalMethodenJava9 -C mods/com.java.migration.app .  
jar --create --file mlib/com.java.migration.buchleser.jar -C  
mods/com.java.migration.buchleser .  
java --module-path mlib -m  
com.java.migration.app/com.java.migration.app.OptionalMethodenJava9
```

und prüfen Sie im Nachhinein mit der Option `-describe-module` von `jar` den Inhalt und die Abhängigkeiten der eingerichteten Module und ihrer Packages:

```
cs: java9
C:\Users\Lissi\Documents\java9\java9migration>Java9ModularJar

C:\Users\Lissi\Documents\java9\java9migration>javac -d mods --module-source-path
src src/com.java.migration.app/module-info.java src/com.java.migration.app/com/
java/migration/app/OptionalMethodenJava9.java

C:\Users\Lissi\Documents\java9\java9migration>javac -d mods --module-source-path
src src/com.java.migration.buchleser/module-info.java src/com.java.migration.bu
chleser/com/java/migration/buchleser/*.java

C:\Users\Lissi\Documents\java9\java9migration>jar --create --file mlib/com.java.
migration.app.jar --main-class=com.java.migration.app.OptionalMethodenJava9 -C m
ods/com.java.migration.app .

C:\Users\Lissi\Documents\java9\java9migration>jar --create --file mlib/com.java.
migration.buchleser.jar -C mods/com.java.migration.buchleser .

C:\Users\Lissi\Documents\java9\java9migration>java --module-path mlib -m com.jav
a.migration.app/com.java.migration.app.OptionalMethodenJava9

Die or()-Methode von Optional:
Java 8
Java 8 Java 9 Java 8 Java 9
Optional.empty
Java 8 Java 9

Die ifPresentorElse()-Methode von Optional:
Jung
Der Leser[13] wurde nicht gefunden
Müller

Die stream()-Methode von Optional:

Leser die Bücher bestellt haben:
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]

Der Leser mit der ID = 11 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
```

```

C:\Users\Lissi> java9
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Diese wurden wie folgt bewertet:
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit

Der Leser mit der ID = 10 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Der Leser mit der ID = 12 bestellte die Bücher:
Java 8 Das Übungsbuch Band I Elisabeth Jung
Java 8 Das Übungsbuch Band II Elisabeth Jung
Java 8 Servlets und JavaServer Pages Elisabeth Jung
Android 5 Übungsbuch Elisabeth Jung

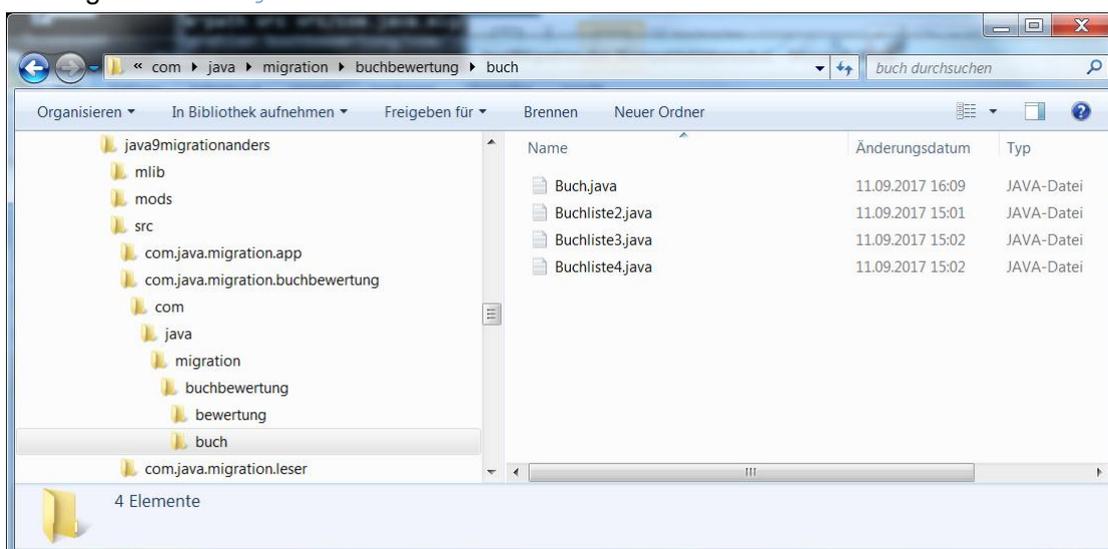
C:\Users\Lissi\Documents\java9\java9migration>jar --file mlib/com.java.migration
.buchleser.jar -d --describe-module
com.java.migration.buchleser jar:file:///C:/Users/Lissi/Documents/java9/java9mig
ration/mlib/com.java.migration.buchleser.jar!/module-info.class
exports com.java.migration.buchleser
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migration>jar --file mlib/com.java.migration
.app.jar -d --describe-module
com.java.migration.app jar:file:///C:/Users/Lissi/Documents/java9/java9migra
tion/mlib/com.java.migration.app.jar!/module-info.class
requires com.java.migration.buchleser
requires java.base mandated
contains com.java.migration.app
main-class com.java.migration.app.OptionalMethodenJava9

C:\Users\Lissi\Documents\java9\java9migration>

```

In einem neuen Projektverzeichnis `java9migrationanders` soll nun eine neue Applikation erstellt werden, die über drei Module `com.java.migration.app`, `com.java.migration.buchbewertung` und `com.java.migration.leser` verfügt, wobei das Modul `com.java.migration.buchbewertung` zwei Packages `bewertung` und `buch` beinhaltet:



Dazu definiert das Verzeichnis `buchbewertung` zwei Unterverzeichnisse `bewertung` und `buch`, die die zwei Namen für die Pakete des Moduls `com.java.migration.buchbewertung` vorgeben (im Package `com.java.migration.buchbewertung.bewertung` wird die Datei `BuchBewertung.java` abgelegt und im Package `com.java.migration.buchbewertung.buch` die Dateien `Buch.java`, `Buchliste2.java`, `Buchliste3.java` und `Buchliste4.java`).

Die zwei anderen Module `com.java.migration.app` und `com.java.migration.leser` bestehen aus nur einem Package, das denselben Namen wie den des Moduls trägt und die Datei `OptionalMethodenJava9.java` bzw. `Leser.java` beinhaltet.

Definieren Sie für diese Form der Modularisierung die modul-info-Dateien mit:

```
module com.java.migration.app {
// Module importieren
  requires com.java.migration.leser;
  requires com.java.migration.buchbewertung;
}
module com.java.migration.buchbewertung {
  exports com.java.migration.buchbewertung.buch;
  exports com.java.migration.buchbewertung.bewertung;
}
module com.java.migration.leser {
  exports com.java.migration.leser;
}
```

und die nachfolgende `Java9ModularJar.cmd`-Kommandodatei zum Kompilieren und Ausführen der Applikation:

```
javac -d mods --module-source-path src src/com.java.migration.buchbewertung/module-
info.java
src/com.java.migration.buchbewertung/com/java/migration/buchbewertung/bewertung/BuchB
ewertung.java
javac -d mods --module-source-path src src/com.java.migration.buchbewertung/module-
info.java
src/com.java.migration.buchbewertung/com/java/migration/buchbewertung/buch/*.java
javac -d mods --module-source-path src src/com.java.migration.app/module-info.java
src/com.java.migration.app/com/java/migration/app/OptionalMethodenJava9.java
javac -d mods --module-source-path src src/com.java.migration.leser/module-info.java
src/com.java.migration.leser/com/java/migration/leser/*.java
jar --create --file mlib/com.java.migration.app.jar --main-
class=com.java.migration.app.OptionalMethodenJava9 -C mods/com.java.migration.app .
jar --create --file mlib/com.java.migration.buchbewertung.jar -C
mods/com.java.migration.buchbewertung .
jar --create --file mlib/com.java.migration.leser.jar -C
mods/com.java.migration.leser .
java --module-path mlib -m
com.java.migration.app/com.java.migration.app.OptionalMethodenJava9
```

Die damit erzeugten Ausgaben und die von `jar` mit der Option `-describe-module` zeigen, dass ein Modul auch aus mehreren Packages bestehen kann:

```
cmd: java9
C:\Users\Lissi\Documents\java9\java9migrationanders>Java9ModularJar

C:\Users\Lissi\Documents\java9\java9migrationanders>javac -d mods --module-source-path src src/com.java.migration.buchbewertung/module-info.java src/com.java.migration.buchbewertung/com/java/migration/buchbewertung/bewertung/BuchBewertung.java

C:\Users\Lissi\Documents\java9\java9migrationanders>javac -d mods --module-source-path src src/com.java.migration.buchbewertung/module-info.java src/com.java.migration.buchbewertung/com/java/migration/buchbewertung/buch/*.java

C:\Users\Lissi\Documents\java9\java9migrationanders>javac -d mods --module-source-path src src/com.java.migration.app/module-info.java src/com.java.migration.app/com/java/migration/app/OptionalMethodenJava9.java

C:\Users\Lissi\Documents\java9\java9migrationanders>javac -d mods --module-source-path src src/com.java.migration.leser/module-info.java src/com.java.migration.leser/com/java/migration/leser/*.java

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --create --file mlib/com.java.migration.app.jar --main-class=com.java.migration.app.OptionalMethodenJava9 -C mods/com.java.migration.app .

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --create --file mlib/com.java.migration.buchbewertung.jar -C mods/com.java.migration.buchbewertung .

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --create --file mlib/com.java.migration.leser.jar -C mods/com.java.migration.leser .

C:\Users\Lissi\Documents\java9\java9migrationanders>java --module-path mlib -m com.java.migration.app/com.java.migration.app.OptionalMethodenJava9

Die or()-Methode von Optional:
Java 8
Java 8 Java 9 Java 8 Java 9
Optional.empty
Java 8 Java 9

Die ifPresentOrElse()-Methode von Optional:
Jung
Der Leser[13] wurde nicht gefunden
Müller

Die stream()-Methode von Optional:

Leser die Bücher bestellt haben:
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
```

```

C:\Users\Lissi> java9
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]

Der Leser mit der ID = 11 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Diese wurden wie folgt bewertet:
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit

Der Leser mit der ID = 10 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Der Leser mit der ID = 12 bestellte die Bücher:
Java 8 Das Übungsbuch Band I Elisabeth Jung
Java 8 Das Übungsbuch Band II Elisabeth Jung
Java 8 Servlets und JavaServer Pages Elisabeth Jung
Android 5 Übungsbuch Elisabeth Jung

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --file mlib/com.java.migration.app.jar -d --describe-module
com.java.migration.app jar:file:///C:/Users/Lissi/Documents/java9/java9migrationanders/mlib/com.java.migration.app.jar!/module-info.class
requires com.java.migration.buchbewertung

```

```

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --file mlib/com.java.migration.app.jar -d --describe-module
com.java.migration.app jar:file:///C:/Users/Lissi/Documents/java9/java9migrationanders/mlib/com.java.migration.app.jar!/module-info.class
requires com.java.migration.buchbewertung
requires com.java.migration.leser
requires java.base mandated
contains com.java.migration.app
main-class com.java.migration.app.OptionalMethodenJava9

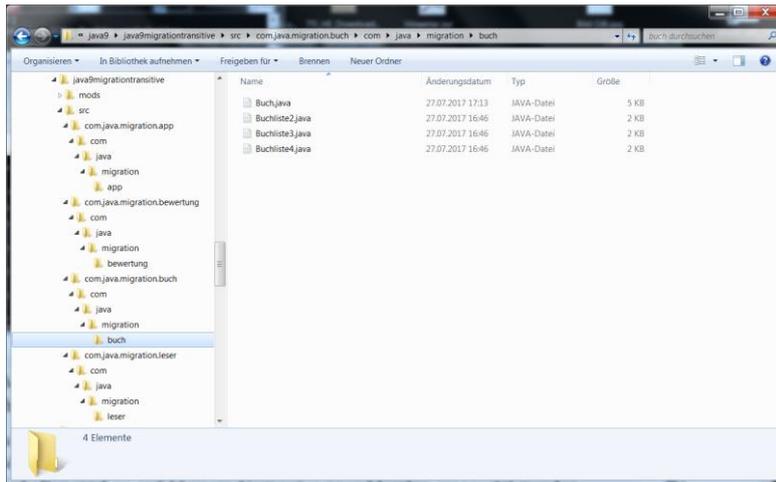
C:\Users\Lissi\Documents\java9\java9migrationanders>jar --file mlib/com.java.migration.buchbewertung.jar -d --describe-module
com.java.migration.buchbewertung jar:file:///C:/Users/Lissi/Documents/java9/java9migrationanders/mlib/com.java.migration.buchbewertung.jar!/module-info.class
exports com.java.migration.buchbewertung.bewertung
exports com.java.migration.buchbewertung.buch
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migrationanders>jar --file mlib/com.java.migration.leser.jar -d --describe-module
com.java.migration.leser jar:file:///C:/Users/Lissi/Documents/java9/java9migrationanders/mlib/com.java.migration.leser.jar!/module-info.class
exports com.java.migration.leser
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migrationanders>_

```

Richten Sie sich nun nach dem Beispiel aus der Aufgabe 12.11, um das »implizite Lesen« von Modulen, die den `transitive`-Modifizierer in einem `requires`-Statement benutzen, wiederholend einzuüben. Erstellen Sie dazu in einem neuen Arbeitsverzeichnis `java9migrationtransitive` eine Applikation mit vier Modulen, deren Aufbau durch die nachfolgende Verzeichnisstruktur unterstützt wird:



Diese sollen die entsprechenden Java-Klassen (mit den zugehörigen `package`- und `import`-Anweisungen) beinhalten und die nachfolgenden `module-info`-Dateien definieren:

```
module com.java.migration.bewertung {
    exports com.java.migration.bewertung;
}

module com.java.migration.app {
    // Module importieren
    requires com.java.migration.buch;
    requires com.java.migration.leser;
    // Wird in der nachfolgenden Deskriptordatei das Statement
    // requires com.java.migration.bewertung; mit
    // requires transitive com.java.migration.bewertung;
    // ersetzt, ist auch die nachfolgende Anweisung überflüssig
    // requires com.java.migration.bewertung;
}

module com.java.migration.buch {
    requires transitive com.java.migration.bewertung;
    //requires com.java.migration.bewertung;
    exports com.java.migration.buch;
}

module com.java.migration.leser {
    exports com.java.migration.leser;
}
```

Kompilieren und starten Sie diese Applikation mit und ohne implizites Lesen und vergleichen Sie die Ergebnisse mit den nachfolgenden Anzeigen.

Erzeugen Sie beim Übersetzen zusätzlich JAR-Archive mit der nachfolgenden `Java9ModularJar.cmd`:

```
javac -d mods --module-source-path src src/com.java.migration.app/module-info.java
src/com.java.migration.app/com/java/migration/app/OptionalMethodenJava9.java
javac -d mods --module-source-path src src/com.java.migration.bewertung/module-
info.java src/com.java.migration.bewertung/com/java/migration/bewertung/*.java
javac -d mods --module-source-path src src/com.java.migration.buch/module-info.java
src/com.java.migration.buch/com/java/migration/buch/*.java
javac -d mods --module-source-path src src/com.java.migration.leser/module-info.java
src/com.java.migration.leser/com/java/migration/leser/*.java
jar --create --file mlib/com.java.migration.app.jar --main-
class=com.java.migration.app.OptionalMethodenJava9 -C mods/com.java.migration.app .
jar --create --file mlib/com.java.migration.bewertung.jar -C
mods/com.java.migration.bewertung .
jar --create --file mlib/com.java.migration.buch.jar -C mods/com.java.migration.buch
.
jar --create --file mlib/com.java.migration.leser.jar -C
mods/com.java.migration.leser .
java --module-path mlib -m
com.java.migration.app/com.java.migration.app.OptionalMethodenJava9
```

und prüfen Sie wie immer mit der Option `-describe-module` von `jar` den Inhalt und die

## Abhängigkeiten der eingerichteten Module:

```
cs: java9
C:\Users\Lissi\Documents\java9\java9migrationtransitive>Java9ModularJar

C:\Users\Lissi\Documents\java9\java9migrationtransitive>javac -d mods --module-source-path src src/com.java.migration.app/module-info.java src/com.java.migration.app/com/java/migration/app/OptionalMethodenJava9.java

C:\Users\Lissi\Documents\java9\java9migrationtransitive>javac -d mods --module-source-path src src/com.java.migration.bewertung/module-info.java src/com.java.migration.bewertung/com/java/migration/bewertung/*.java

C:\Users\Lissi\Documents\java9\java9migrationtransitive>javac -d mods --module-source-path src src/com.java.migration.buch/module-info.java src/com.java.migration.buch/com/java/migration/buch/*.java

C:\Users\Lissi\Documents\java9\java9migrationtransitive>javac -d mods --module-source-path src src/com.java.migration.leser/module-info.java src/com.java.migration.leser/com/java/migration/leser/*.java

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --create --file mlib/com.java.migration.app.jar --main-class=com.java.migration.app.OptionalMethodenJava9 -C mods/com.java.migration.app .

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --create --file mlib/com.java.migration.bewertung.jar -C mods/com.java.migration.bewertung .

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --create --file mlib/com.java.migration.buch.jar -C mods/com.java.migration.buch .

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --create --file mlib/com.java.migration.leser.jar -C mods/com.java.migration.leser .

C:\Users\Lissi\Documents\java9\java9migrationtransitive>java --module-path mlib -m com.java.migration.app/com.java.migration.app.OptionalMethodenJava9

Die or()-Methode von Optional:
Java 8
Java 8 Java 9 Java 8 Java 9
Optional.empty
Java 8 Java 9

Die ifPresentOrElse()-Methode von Optional:
Jung
Der Leser[13] wurde nicht gefunden
Müller

Die stream()-Methode von Optional:

Leser die Bücher bestellt haben:
Leser[id=10 name=Jung]
```

```
cs: java9
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]

Der Leser mit der ID = 11 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Diese wurden wie folgt bewertet:
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit

Der Leser mit der ID = 10 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Der Leser mit der ID = 12 bestellte die Bücher:
Java 8 Das Übungsbuch Band I Elisabeth Jung
Java 8 Das Übungsbuch Band II Elisabeth Jung
Java 8 Servlets und JavaServer Pages Elisabeth Jung
Android 5 Übungsbuch Elisabeth Jung
```

```
cs: java9
C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --file mlib/com.java
.migration.app.jar -d --describe-module
com.java.migration.app jar:file:///C:/Users/Lissi/Documents/java9/java9migration
transitive/mlib/com.java.migration.app.jar!/module-info.class
requires com.java.migration.buch
requires com.java.migration.leser
requires java.base mandated
contains com.java.migration.app
main-class com.java.migration.app.OptionalMethodenJava9

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --file mlib/com.java
.migration.bewertung.jar -d --describe-module
com.java.migration.bewertung jar:file:///C:/Users/Lissi/Documents/java9/java9mig
rationtransitive/mlib/com.java.migration.bewertung.jar!/module-info.class
exports com.java.migration.bewertung
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --file mlib/com.java
.migration.buch.jar -d --describe-module
com.java.migration.buch jar:file:///C:/Users/Lissi/Documents/java9/java9migratio
ntransitive/mlib/com.java.migration.buch.jar!/module-info.class
exports com.java.migration.buch
requires com.java.migration.bewertung transitive
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migrationtransitive>jar --file mlib/com.java
.migration.leser.jar -d --describe-module
com.java.migration.leser jar:file:///C:/Users/Lissi/Documents/java9/java9migrati
ontransitive/mlib/com.java.migration.leser.jar!/module-info.class
exports com.java.migration.leser
requires java.base mandated

C:\Users\Lissi\Documents\java9\java9migrationtransitive>
```

Parallel zu diesem Weg der expliziten Module, sollen für die Migration derselben Applikation nun automatische Module eingesetzt werden.

Richten Sie dazu das Projektverzeichnis `java9migrationautomaticmodule` mit den Unterverzeichnissen `app`, `buch`, `bewertung` und `leser` ein, in denen die obigen Source-Dateien

ohne weitere Unterverzeichnisse (der Einfachheit halber) den Namen entsprechend integriert werden sollen. Legen Sie auch diesmal die beim Übersetzen erzeugten `.class`-Dateien in JAR-Dateien ab, um für die so erstellte Applikation den Weg der Migration über Automatic Modules zu zeigen. Es soll erstmals eine modularisierte Applikation mit den Modulen `app`, `buch`, `bewertung` und `leser`, die gleichnamige Packages enthalten, erstellt werden. Folgen Sie dabei den Anweisungen:

```
java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac bewertung/BuchBewertung.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac leser/Leser.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar -cvf jarlib\bewertung.jar bewertung/BuchBewertung.class
Manifest wurde hinzugef"gt
bewertung/BuchBewertung.class wird hinzugef"gt(ein = 1130) (aus = 548)(51 % verkleinert)
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar -cvf jarlib\leser.jar leser/Leser.class
Manifest wurde hinzugef"gt
leser/Leser.class wird hinzugef"gt(ein = 1043) (aus = 527)(49 % verkleinert)
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac -cp jarlib\bewertung.jar buch/Buch.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar -cvf jarlib\buch.jar buch/Buch.class
Manifest wurde hinzugef"gt
buch/Buch.class wird hinzugef"gt(ein = 4551) (aus = 1779)(60 % verkleinert)
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac -cp jarlib\buch/Buchliste2.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac -cp jarlib\buch/Buchliste3.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac -cp jarlib\buch/Buchliste4.java
```

```
java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar -cvf jarlib\buch.jar buch/Buch.class buch/Buchliste2.class buch/Buchliste3.class buch/Buchliste4.class
Manifest wurde hinzugef"gt
buch/Buch.class wird hinzugef"gt(ein = 4551) (aus = 1779)(60 % verkleinert)
buch/Buchliste2.class wird hinzugef"gt(ein = 1422) (aus = 781)(45 % verkleinert)
buch/Buchliste3.class wird hinzugef"gt(ein = 1422) (aus = 783)(44 % verkleinert)
buch/Buchliste4.class wird hinzugef"gt(ein = 1422) (aus = 781)(45 % verkleinert)
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>javac -sourcepath . -cp jarlib/* app/OptionalMethodenJava9.java
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>
```

```
cx. java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar --create --file
jarlib\app.jar --main-class=app.OptionalMethodenJava9 app/OptionalMethodenJava9
.class buch/Buch.class bewertung/BuchBewertung.class buch/Buchliste2.class buch/
Buchliste3.class buch/Buchliste4.class leser/Leser.class

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>java -jar jarlib/ap
p.jar

Die or()-Methode von Optional:
Java 8
Java 8 Java 9 Java 8 Java 9
Optional.empty
Java 8 Java 9

Die ifPresenterElse()-Methode von Optional:
Jung
Der Leser[13] wurde nicht gefunden
Müller

Die stream()-Methode von Optional:

Leser die Bücher bestellt haben:
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]

Der Leser mit der ID = 11 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Diese wurden wie folgt bewertet:
```

```
cx. java9
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit

Der Leser mit der ID = 10 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Der Leser mit der ID = 12 bestellte die Bücher:
Java 8 Das Übungsbuch Band I Elisabeth Jung
Java 8 Das Übungsbuch Band II Elisabeth Jung
Java 8 Servlets und JavaServer Pages Elisabeth Jung
Android 5 Übungsbuch Elisabeth Jung

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jdeps --generate-mo
dule-info out\jarlib\bewertung.jar jarlib\buch.jar jarlib\leser.jar jarlib/app.j
ar
writing to out\app\module-info.java
writing to out\bewertung\module-info.java
writing to out\buch\module-info.java
writing to out\leser\module-info.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>type out\app\module
-info.java
module app {
    requires transitive bewertung;
    requires transitive buch;
    requires transitive leser;

    exports app;
    exports bewertung;
```

```
java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>type out\app\module-info.java
module app {
    requires transitive bewertung;
    requires transitive buch;
    requires transitive leser;

    exports app;
    exports bewertung;
    exports buch;
    exports leser;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>type out\bewertung\module-info.java
module bewertung {
    exports bewertung;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>type out\buch\module-info.java
module buch {
    requires transitive bewertung;

    exports buch;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>type out\leser\module-info.java
module leser {
    exports leser;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>
```

```
java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar --file jarlib/bewertung.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

bewertung automatic
requires java.base mandated
contains bewertung

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar --file jarlib/leser.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

leser automatic
requires java.base mandated
contains leser

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar --file jarlib/buch.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

buch automatic
requires java.base mandated
contains buch

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>jar --file jarlib/app.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

app automatic
requires java.base mandated
contains app
contains bewertung
contains buch
contains leser
main-class app.OptionalMethodenJava9

C:\Users\Lissi\Documents\java9\java9migrationautomaticmodule>
```

Achten Sie darauf, dass beim Erzeugen von JAR-Dateien mit der Option `--main-class` die Klasse, die die `main()`-Methode enthält, spezifiziert werden kann.

Nun soll auch für den Fall der Migration mit automatischen Modulen ein Beispiel mit einem Modul, das mehrere Packages beinhaltet, erläutert werden. In einem neuen Projektverzeichnis `java9migrationautomaticanders` mit den Unterverzeichnissen `app` und `buchbewertungleser` (das

seinerseits die Unterverzeichnisse `bewertung`, `leser` und `buch` definiert) soll eine modularisierte Applikation mit zwei Modulen `app` und `buchbewertungleser` erstellt werden, wobei das erste Modul das gleichnamige Package `app` enthält und das zweite drei Packages mit den Namen `bewertung`, `leser` und `buch`.

Legen Sie auch diesmal die beim Übersetzen erzeugten `.class`-Dateien in JAR-Dateien ab, um auch für die so zusammengebaute Applikation den Weg der Migration über Automatic Modules zu zeigen. Folgen Sie dabei den Anweisungen:



```
ca. java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac buchbewertung
leser/leser/Leser.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac buchbewertung
leser/bewertung/BuchBewertung.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jar -cvf jarlib\buc
hbewertungleser.jar buchbewertungleser/bewertung/BuchBewertung.class buchbewertu
ngleser/leser/Leser.class
Manifest wurde hinzugef"gt
buchbewertungleser/bewertung/BuchBewertung.class wird hinzugef"gt(ein = 1149) (a
us = 557)(51 % verkleinert)
buchbewertungleser/leser/Leser.class wird hinzugef"gt(ein = 1062) (aus = 540)(49
% verkleinert)

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac -cp jarlib/*
buchbewertungleser/buch/Buch.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac -cp jarlib/*
buchbewertungleser/buch/Buchliste2.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac -cp jarlib/*
buchbewertungleser/buch/Buchliste3.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac -cp jarlib/*
buchbewertungleser/buch/Buchliste4.java

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jar -cvf jarlib\buc
hbewertungleser.jar buchbewertungleser/buch/Buch.class buchbewertungleser/buch/B
uchliste2.class buchbewertungleser/buch/Buchliste3.class buchbewertungleser/buch
/Buchliste4.class buchbewertungleser/bewertung/BuchBewertung.class buchbewertung
leser/leser/Leser.class
Manifest wurde hinzugef"gt
buchbewertungleser/buch/Buch.class wird hinzugef"gt(ein = 4665) (aus = 1797)(61
% verkleinert)
buchbewertungleser/buch/Buchliste2.class wird hinzugef"gt(ein = 1517) (aus = 793
)(47 % verkleinert)
buchbewertungleser/buch/Buchliste3.class wird hinzugef"gt(ein = 1517) (aus = 800
)(47 % verkleinert)
buchbewertungleser/buch/Buchliste4.class wird hinzugef"gt(ein = 1517) (aus = 793
)(47 % verkleinert)
buchbewertungleser/bewertung/BuchBewertung.class wird hinzugef"gt(ein = 1149) (a
us = 557)(51 % verkleinert)
buchbewertungleser/leser/Leser.class wird hinzugef"gt(ein = 1062) (aus = 540)(49
% verkleinert)

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>javac -cp jarlib/*
app/OptionalMethodenJava9.java
```

```

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jar --create --file
jarlib\app.jar --main-class=app.OptionalMethodenJava9 app/OptionalMethodenJava9
.class buchbewertungleser/bewertung/BuchBewertung.class buchbewertungleser/buch/
Buch.class buchbewertungleser/buch/Buchliste2.class buchbewertungleser/buch/Buch
liste3.class buchbewertungleser/buch/Buchliste4.class buchbewertungleser/leser/L
eser.class

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>java -p jarlib -m a
pp/app.OptionalMethodenJava9
Error occurred during initialization of boot layer
java.lang.module.ResolutionException: Module buchbewertungleser contains package
buchbewertungleser.buch, module app exports package buchbewertungleser.buch to
buchbewertungleser

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>java -jar jarlib/ap
p.jar

Die or()-Methode von Optional:
Java 8
Java 8 Java 9 Java 8 Java 9
Optional.empty
Java 8 Java 9

Die ifPresentorElse()-Methode von Optional:
Jung
Der Leser[13] wurde nicht gefunden
Müller

Die stream()-Methode von Optional:

Leser die Bücher bestellt haben:
Leser[id=10 name=Jung]
Leser[id=11 name=Schmidt]
Leser[id=12 name=Müller]

Der Leser mit der ID = 11 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

```

```

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>java -jar jarlib/ap
p.jar

Diese wurden wie folgt bewertet:
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit
Rezension1 von Spiegel
Rezension2 von Stern
Rezension3 von Zeit

Der Leser mit der ID = 10 bestellte die Bücher:
Java 7 Das Übungsbuch Band I Elisabeth Jung
Java 7 Das Übungsbuch Band II Elisabeth Jung
Java 7 Servlets und JavaServer Pages Elisabeth Jung
Android 4 Übungsbuch Elisabeth Jung

Der Leser mit der ID = 12 bestellte die Bücher:
Java 8 Das Übungsbuch Band I Elisabeth Jung
Java 8 Das Übungsbuch Band II Elisabeth Jung
Java 8 Servlets und JavaServer Pages Elisabeth Jung
Android 5 Übungsbuch Elisabeth Jung

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jdeps --generate-mo
dule-info out\jarlib\buchbewertungleser.jar jarlib/app.jar
writing to out\app\module-info.java
writing to out\buchbewertungleser\module-info.java

```

```
java9
C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>type out\buchbewertungleser\module-info.java
module buchbewertungleser {
    requires transitive app;

    exports buchbewertungleser.bewertung;
    exports buchbewertungleser.buch;
    exports buchbewertungleser.leser;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>type out\app\module-info.java
module app {
    requires transitive buchbewertungleser;

    exports app;
    exports buchbewertungleser.bewertung;
    exports buchbewertungleser.buch;
    exports buchbewertungleser.leser;
}

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jar --file jarlib/app.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

app automatic
requires java.base mandated
contains app
contains buchbewertungleser.bewertung
contains buchbewertungleser.buch
contains buchbewertungleser.leser
main-class app.OptionalMethodenJava9

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>jar --file jarlib/buchbewertungleser.jar -d --describe-module
Kein Moduldeskriptor gefunden. Automatisches Modul wurde abgeleitet.

buchbewertungleser automatic
requires java.base mandated
contains buchbewertungleser.bewertung
contains buchbewertungleser.buch
contains buchbewertungleser.leser

C:\Users\Lissi\Documents\java9\java9migrationautomaticanders>
```

Und noch zwei Anmerkungen zum Schluss: Mithilfe des [jlink](#)-Tools können sogenannte JRE-Images erstellt werden, die nur die benötigten Module (mit einer minimalen Anzahl von Abhängigkeiten) der Java-Plattform beinhalten und so auf kleineren Computern problemlos eingesetzt werden können. Diese Modular Runtime Images sind keine ausführbare Dateien, sondern besitzen ähnlich dem JDK bzw. JRE eine Directory-Struktur, in der alle Module und Ressourcen, von denen das Root-Modul abhängt, abgelegt sind.

So kann z.B. für eine Anwendung, die nur das Modul `java.base` verwendet, ein Laufzeit-Image erstellt werden, das nur aus den Applikationsmodulen und aus dem `java.base`-Modul besteht. [jlink](#)-Images können betriebssystemspezifisch erstellt werden.

Die Applikationen aus diesem Buch verwenden keine JDK-interne APIs aus den `sun.*`-Packages. Ist dies in Java-Anwendungen der Fall, müssen sie im Zuge von Migration eliminiert werden, da diese im neuen modularisierten JDK wirklich »intern« sind und nicht mehr für Entwickler zugänglich sind. Deren Vorhandensein kann mit dem `jdeps`-Tool mit der Option `-jdkinternals` geprüft werden. Dieses liefert auch einen Hinweis auf mögliche alternative APIs, die an deren Stelle benutzt werden können.