Modularizing AcmeAir

Software versions used for this demo

- **OS**

  $ cat /etc/issue

  ```
  Ubuntu 17.04 \n \l
  ```

  $ uname -a

  ```
  Linux modularity 4.10.0-35-generic #39-Ubuntu SMP Wed Sep 13 07:46:59 UTC 2017 x86_64
  x86_64 x86_64 GNU/Linux
  ```

- **Java**

  $ java -version

  ```
  java version "1.8.0"
  Java(TM) SE Runtime Environment (build pxa6480sr4fp10-20170727_01(SR4 FP10))
  IBM J9 VM (build 2.8, JRE 1.8.0 Linux amd64-64 Compressed References
  20170722_357405 (JIT enabled, AOT enabled)
  J9VM - R28_20170722_0201_B357405
  JIT  - tr.r14.java_20170722_357405
  GC   - R28_20170722_0201_B357405_CMPRSS
  J9CL - 20170722_357405)
  JCL - 20170726_01 based on Oracle jdk8u144-b01
  ```

  $ java -version

  ```
  openjdk version "9"
  OpenJDK Runtime Environment (build 9+181)
  OpenJDK 64-Bit Server VM (build 9+181, mixed mode)
  ```

- **Gradle**

  $ gradle -version

  ```
  WARNING: An illegal reflective access operation has occurred
  WARNING: Illegal reflective access by org.codehaus.groovy.reflection.CachedClass
  (file:/home/dino/data/java/tools/gradle-4.2/lib/groovy-all-2.4.11.jar) to method
  java.lang.Object.finalize()
  WARNING: Please consider reporting this to the maintainers of
  org.codehaus.groovy.reflection.CachedClass
  WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective
  access operations
  WARNING: All illegal access operations will be denied in a future release

  ------------------------------------------------------------
  Gradle 4.2
  ------------------------------------------------------------

  Build time:   2017-09-20 14:48:23 UTC
  Revision:     5ba503cc17748671c83ce35d7da1cffd6e24dfbd

  Groovy:       2.4.11
  Ant:          Apache Ant(TM) version 1.9.6 compiled on June 29 2015
  ```

```
JVM:        9-internal (Eclipse OpenJ9 2.9)
OS:           Linux 4.10.0-35-generic amd64
```

- It appears that the latest gradle version at this time (4.2) is using internal classes of the JDK
  $ jdeps -jdkinternals /home/dino/data/java/tools/gradle-4.2/lib/groovy-all-2.4.11.jar

```
groovy-all-2.4.11.jar -> jdk.unsupported
   groovy.json.internal.FastStringUtils               -> sun.misc.Unsafe
JDK internal API (jdk.unsupported)
   groovy.json.internal.FastStringUtils$StringImplementation$1 -> sun.misc.Unsafe
JDK internal API (jdk.unsupported)
   groovy.json.internal.FastStringUtils$StringImplementation$2 -> sun.misc.Unsafe
JDK internal API (jdk.unsupported)

Warning: JDK internal APIs are unsupported and private to JDK implementation that are
subject to be removed or changed incompatibly and could break your application.
Please modify your code to eliminate dependence on any JDK internal APIs.
For the most recent update on JDK internal API replacements, please check:
https://wiki.openjdk.java.net/display/JDK8/Java+Dependency+Analysis+Tool

JDK Internal API                          Suggested Replacement
----------------                          ---------------------
sun.misc.Unsafe                           See http://openjdk.java.net/jeps/260
```

Setup

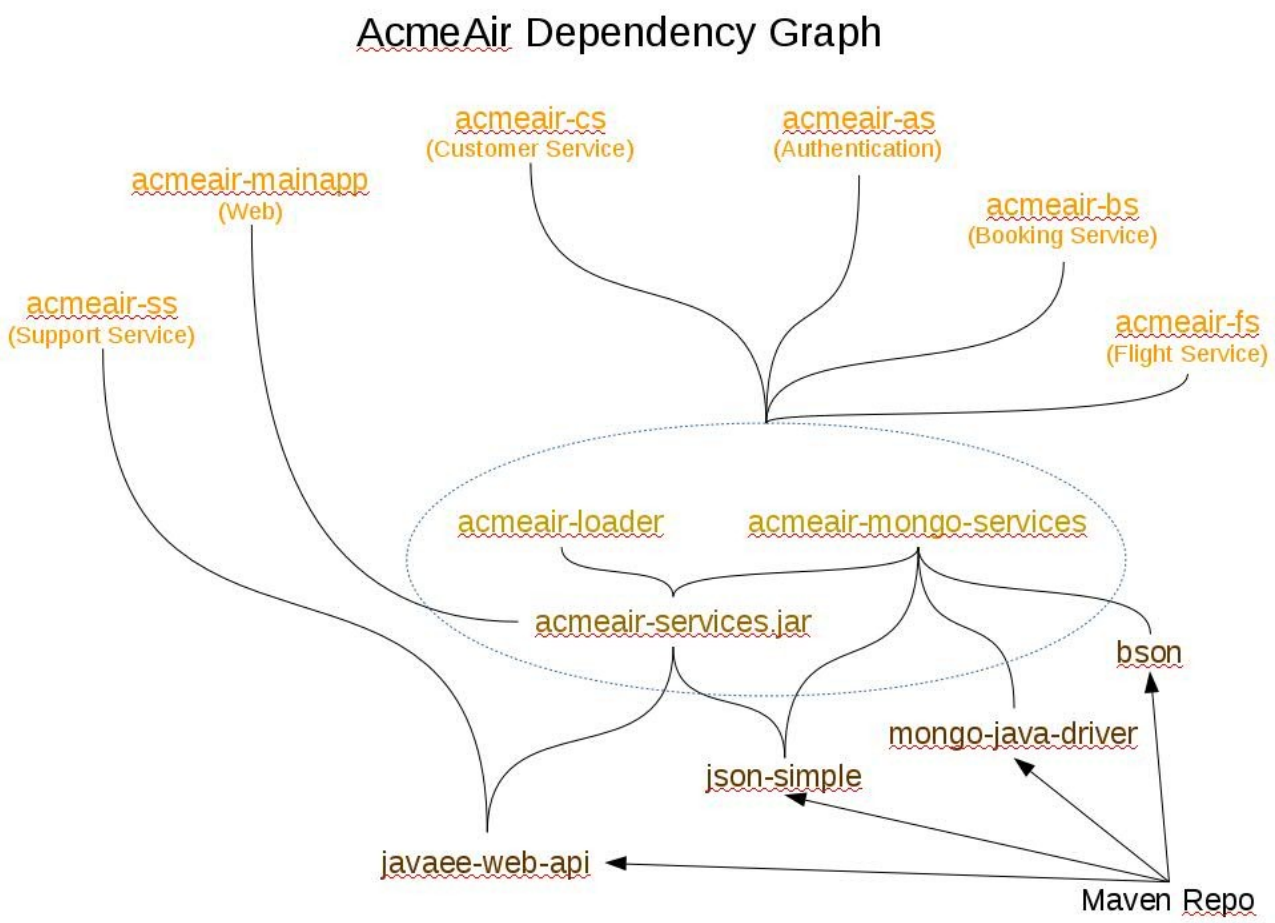
1. Get the Acmeair benchmark
   $ git clone https://www.github.com/ibmruntimes/acmeair-modular
   $ cd acmeair-modular
2. Use the Java 8 compiler and build the project first
   (This is needed as on some OSes, gradle fails to download the dependent jar files with the Java 9 SDK. However this works on the older sdk and once these jars are available on the local cache, you can switch over to the Java 9 compiler). This appears to be a gradle issue.
   $ gradle build
3. Now change to the Java 9 compiler and redo the previous step. It is expected to fail.
   $ gradle build

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.gradle.internal.reflect.JavaMethod
(file:/home/dino/data/java/tools/gradle-3.5.1/lib/gradle-base-services-3.5.1.jar) to method
java.lang.ClassLoader.getPackages()
WARNING: Please consider reporting this to the maintainers of
org.gradle.internal.reflect.JavaMethod
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access
operations
WARNING: All illegal access operations will be denied in a future release
Starting a Gradle Daemon (subsequent builds will be faster)
:acmeair-services:compileJava
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/CustomerInfo.java:20: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
```

```
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/CustomerInfo.java:21: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessorType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/CustomerInfo.java:22: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/CustomerInfo.java:23: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlRootElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/AddressInfo.java:20: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/AddressInfo.java:21: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessorType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
/home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/web/dto/AddressInfo.java:22: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlRootElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is not in the
module graph)
Note: /home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/util/Util.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: /home/dino/data/java/github/blueperf/9/acmeair/acmeair-
services/src/main/java/com/acmeair/util/Util.java uses unchecked or unsafe operations
Note: Recompile with -Xlint:unchecked for details.
7 errors
:acmeair-services:compileJava FAILED

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':acmeair-services:compileJava'.
> Compilation failed; see the compiler error output for details.

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get
more log output.

BUILD FAILED

Total time: 6.054 secs
```

Acmeair basics

The original acmeair version is available here → https://github.com/blueperf/acmeair



This application shows an implementation of a fictitious airline called "Acme Air". The application was built with some key business requirements: the ability to scale to billions of web API calls per day, the need to develop and deploy the application targeting multiple cloud platforms (including Public, Dedicated, Private and hybrid) and the need to support multiple channels for user interaction (with mobile enablement first and browser/Web 2.0 second).The application can be deployed both on-prem as well as on Cloud platforms.

This application has been restructured to be a microservices application in addion to a monolithic application. Monolithc appliction is refactored to make sure the application can scale with multiple Cloud Data Servces. Microservices version is being developed actively to support multiple Service Proxy, Service Discovery and Resliency technologies. This application is also being enhanced with additional services using IBM Cognitive capabilites like Watson Dialog etc.

Modularizing package acmeair-services

As seen in the above graph, acmeair-services is a base package that all other application packages depend on. So let us start by modularizing the acmeair-services package.

1. As a first step let us compile the acmeair-services package with the Java9 compiler. (libs folder has manually downloded jar files javaee-web-api-7.0.jar and json-simple-1.1.1.jar.

   $ javac -d out -cp libs/javaee-web-api-7.0.jar:libs/json-simple-1.1.1.jar $(find src -name '*.java')

```
src/main/java/com/acmeair/web/dto/CustomerInfo.java:20: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/CustomerInfo.java:21: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessorType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/CustomerInfo.java:22: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/CustomerInfo.java:23: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlRootElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/AddressInfo.java:20: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/AddressInfo.java:21: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlAccessorType;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
src/main/java/com/acmeair/web/dto/AddressInfo.java:22: error: package
javax.xml.bind.annotation is not visible
import javax.xml.bind.annotation.XmlRootElement;
                    ^
  (package javax.xml.bind.annotation is declared in module java.xml.bind, which is
not in the module graph)
Note: src/main/java/com/acmeair/util/Util.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: src/main/java/com/acmeair/util/Util.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
```

```
7 errors
```

2. The above failures are because, by default java.se (aggregate module) is considered as the root if it is present (which it is in this case) and java.se does not have the java.xml.bind module.
3. An easy way to solve this problem is to include the java.se.ee aggregate module, which is a super-set of all java modules included in the sdk.

```
$ javac -d out -cp libs/javaee-web-api-7.0.jar:libs/json-simple-1.1.1.jar --add-modules java.se.ee $(find src -name '*.java')
Note: src/main/java/com/acmeair/util/Util.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: src/main/java/com/acmeair/util/Util.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
```

4. Let us now create a jarfile with this

   $ mkdir nmlibs
   $ jar -cvf nmlibs/acmeair-services-nm.jar -C out .

5. While adding java.se.ee solves the problem, it is not the best approach. We need to understand what individual modules from java.se.ee are needed. How do we do that ? Using the jdeps tool ofcourse !

6. jdeps - Java class dependency analyzer. One of the options (-jdkinternals) of jdeps tells us if the given app is using any jdk internal APIs.

   $ jdeps -jdkinternals libs/acmeair-services-nm.jar

7. No output for -jdkinternals is a good thing !

   $ jdeps -jdkinternals libs/acmeair-services-nm.jar

8. Let us now look for the dependencies for jar

```
$ jdeps nmlibs/acmeair-services-nm.jar
acmeair-services-nm.jar -> java.base
acmeair-services-nm.jar -> java.logging
acmeair-services-nm.jar -> java.naming
acmeair-services-nm.jar -> java.xml.bind
acmeair-services-nm.jar -> java.xml.ws.annotation
acmeair-services-nm.jar -> not found
   com.acmeair                                -> java.lang                                java.base
   com.acmeair                                -> java.lang.invoke                         java.base
   com.acmeair.service                        -> com.acmeair                              acmeair-services-
nm.jar
   com.acmeair.service                        -> com.acmeair.web.dto                      acmeair-services-
nm.jar
   com.acmeair.service                        -> java.io                                  java.base
   com.acmeair.service                        -> java.lang                                java.base
   com.acmeair.service                        -> java.lang.annotation                     java.base
   com.acmeair.service                        -> java.lang.invoke                         java.base
   com.acmeair.service                        -> java.lang.reflect                        java.base
   com.acmeair.service                        -> java.util                                java.base
   com.acmeair.service                        -> java.util.concurrent                     java.base
   com.acmeair.service                        -> java.util.concurrent.atomic              java.base
   com.acmeair.service                        -> java.util.logging                        java.logging
   com.acmeair.service                        -> javax.annotation
java.xml.ws.annotation
   com.acmeair.service                        -> javax.enterprise.context.spi             not found
   com.acmeair.service                        -> javax.enterprise.inject                  not found
```

```
com.acmeair.service                     -> javax.enterprise.inject.spi          not found
com.acmeair.service                     -> javax.enterprise.util                not found
com.acmeair.service                     -> javax.inject                         not found
com.acmeair.service                     -> javax.naming                         java.naming
com.acmeair.service                     -> org.json.simple                      not found
com.acmeair.service                     -> org.json.simple.parser               not found
com.acmeair.util                        -> java.io                              java.base
com.acmeair.util                        -> java.lang                            java.base
com.acmeair.util                        -> java.lang.invoke                     java.base
com.acmeair.util                        -> java.net                             java.base
com.acmeair.util                        -> java.nio.charset                     java.base
com.acmeair.util                        -> javax.servlet                        not found
com.acmeair.util                        -> org.json.simple                      not found
com.acmeair.util                        -> org.json.simple.parser               not found
com.acmeair.web.dto                     -> java.io                              java.base
com.acmeair.web.dto                     -> java.lang                            java.base
com.acmeair.web.dto                     -> java.lang.invoke                     java.base
com.acmeair.web.dto                     -> javax.xml.bind.annotation java.xml.bind
```

9. Need to add javaee-web-api-7.0.jar and json-simple-1.1.1.jar as before.

```
$ jdeps -s -cp libs/javaee-web-api-7.0.jar:libs/json-simple-1.1.1.jar  nmlibs/acmeair-services-nm.jar
split package: javax.annotation [jrt:/java.xml.ws.annotation, ../libs/javaee-web-api-7.0.jar]

split package: javax.transaction [jrt:/java.transaction, ../libs/javaee-web-api-7.0.jar]

split package: javax.transaction.xa [jrt:/java.sql, ../libs/javaee-web-api-7.0.jar]

acmeair-services-2.0.0-SNAPSHOT.jar -> java.base
acmeair-services-2.0.0-SNAPSHOT.jar -> java.logging
acmeair-services-2.0.0-SNAPSHOT.jar -> java.naming
acmeair-services-2.0.0-SNAPSHOT.jar -> java.xml.bind
acmeair-services-2.0.0-SNAPSHOT.jar -> java.xml.ws.annotation
acmeair-services-2.0.0-SNAPSHOT.jar -> ../libs/javaee-web-api-7.0.jar
acmeair-services-2.0.0-SNAPSHOT.jar -> ../libs/json-simple-1.1.1.jar
```

10. Split package here means that the same classes are being provided by two modules. Let us look at which classes are the issue here

javax/annotation classes in javaee-web-api-7.0.jar

$ tree javax/annotation/

```
javax/annotation/
├── Generated.class
├── ManagedBean.class
├── PostConstruct.class
├── PreDestroy.class
├── Priority.class
├── Resource$AuthenticationType.class
├── Resource.class
├── Resources.class
├── security
│   ├── DeclareRoles.class
│   ├── DenyAll.class
│   ├── PermitAll.class
│   ├── RolesAllowed.class
│   └── RunAs.class
└── sql
    ├── DataSourceDefinition.class
    └── DataSourceDefinitions.class
```

```
2 directories, 15 files
```

javax/annotation classes in java.xml.ws.annotation.jmod

$ tree classes/

```
classes/
├── javax
│   └── annotation
│       ├── Generated.class
│       ├── PostConstruct.class
│       ├── PreDestroy.class
│       ├── Resource$AuthenticationType.class
│       ├── Resource.class
│       └── Resources.class
└── module-info.class
```

2 directories, 7 files

11. Common classes provided by both modules. One way to resolve this is not include java.xml.ws.annotation.jmod. The correct thing to do is for javaee-web-api-7.0.jar to NOT provide javax/annotation classes.

12. So at this point we have an idea about what are the exact dependencies for converting the acmeair-services to a module. The first step in modularity is a slightly different dir structure. The module dir with the right module name should be under src. However gradle expects a certain directory structure. So we are now going to create a module-info.java in the root dir for the java code. The next step to modularize is to create a "module-info.java" file. The module-info.java has the following aspects
    1. Name of the module
    2. What are its dependencies
    3. What does it export if anything

    Based on the jdeps output the module-info.java for acmeair.services should now look like

    ```
    module acmeair.services {
        requires java.base;
        requires java.logging;
        requires java.naming;
        requires java.xml.bind;    What about the external jar files ?
        requires javaee.web.api;       <-- They can be added in the reverse dns form
        requires json.simple;          <-- They can be added in the reverse dns form
    }
    ```

    $ javac -d mods --module-path libs $(find src -name '*.java')

13. So we now have created a module with the right set of dependencies. We still need to figure out what classes are exported by this module. Jdeps is again very helpful here. Recreate the non-modular jar file if it is not already present, jdeps cannot be used to generate the module-info on

a modular jar file.

$ jar -cvf nmlibs/acmeair-services-nm.jar -C out .
$ jdeps --generate-module-info src1 nmlibs/acmeair-services-nm.jar libs/j*

```
writing to src/json.simple/module-info.java
writing to src/acmeair.services/module-info.java
Missing dependence: src/javaee.web.api/module-info.java not generated
split package: javax.annotation [jrt:/java.xml.ws.annotation, ../libs/javaee-web-api-7.0.jar]
split package: javax.transaction [jrt:/java.transaction, ../libs/javaee-web-api-7.0.jar]
split package: javax.transaction.xa [jrt:/java.sql, ../libs/javaee-web-api-7.0.jar]

Error: missing dependencies
   javax.ejb.SessionContext                          -> javax.xml.rpc.handler.MessageContext
not found
   javax.faces.FactoryFinder                         -> com.sun.faces.spi.InjectionProvider
not found
   javax.faces.FactoryFinder$FactoryManager          -> com.sun.faces.spi.InjectionProvider
not found
```

14. The module-info.java now looks like this

```
module acmeair.services {
   requires java.naming;
   requires java.xml.bind;

   requires transitive java.logging;
   requires transitive javaee.web.api;
   requires transitive json.simple;

   exports com.acmeair;
   exports com.acmeair.service;
   exports com.acmeair.util;
   exports com.acmeair.web.dto;
}
```

15. Now let us re-compile with the updated module-info.java

$ javac -d mods --module-path libs $(find src -name '*.java')

```
src/main/java/module-info.java:4: warning: [removal] module java.xml.bind has been
deprecated and marked for removal
    requires java.xml.bind;
                       ^
src/main/java/module-info.java:7: warning: requires transitive directive for an
automatic module
    requires transitive javaee.web.api;
                                  ^
src/main/java/module-info.java:8: warning: requires transitive directive for an
automatic module
    requires transitive json.simple;
                              ^
Note: src/main/java/com/acmeair/util/Util.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: src/main/java/com/acmeair/util/Util.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
```

```
3 warnings
```

16. The above method uses automatic modules for the dependent packages which have not been modularized. However gradle does not yet handle automatic modules gracefully. To overcome this, there is another trick, using unnamed modules.
17. Compile using unnamed modules for dependent packages using “--add-reads”.

```
$ javac -cp libs/javaee-web-api-7.0.jar:libs/json-simple-1.1.1.jar -d mods
--add-reads acmeair.services=ALL-UNNAMED $(find src -name '*.java')
src/main/java/module-info.java:4: warning: [removal] module java.xml.bind has been
deprecated and marked for removal
    requires java.xml.bind;
                     ^
Note: src/main/java/com/acmeair/util/Util.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Note: src/main/java/com/acmeair/util/Util.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
1 warning
```

18. Now let us create a modular jar file for this

    $ jar -cvf libs/acmeair.services.jar -C mods .