Modular Development with JDK 9

Alan Bateman
Java Platform Group, Oracle
February 2016

http://openjdk.java.net/projects/jigsaw

#Jfokus #Jigsaw

Introduction to Modular Development

Alan Bateman
Java Platform Group, Oracle
February 2016

http://openjdk.java.net/projects/jigsaw

#Jfokus #Jigsaw

Background: Modularity Landscape

- Java Platform Module System
  - JSR 376, targeted for Java SE 9
- Java SE 9
  - JSR XXX, will own the modularization of the Java SE APIs
- OpenJDK Project Jigsaw
  - Reference implementation for JSR 376
  - JEP 200, 201, 220, 260, 261

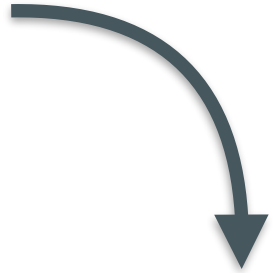
What is a module?

What is a module?

com.foo.bar

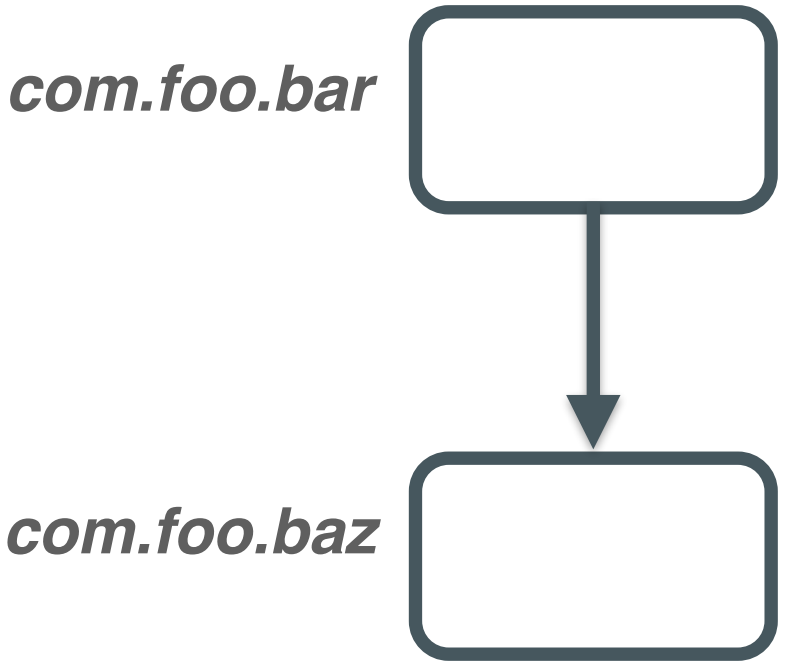
com.foo.bar.alpha.Alpha
com.foo.bar.alpha.AlphaFactory
com.foo.bar.beta.Beta
com.foo.bar.beta.BetaBlocker
:

```
module com.foo.bar {
}
```

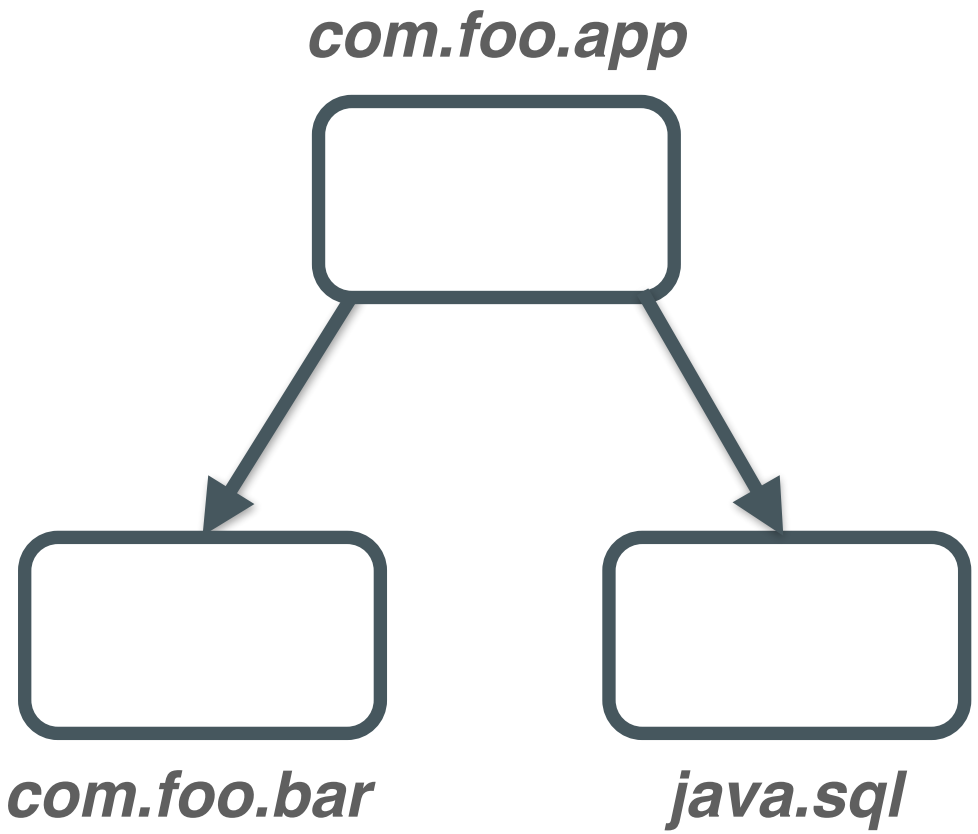


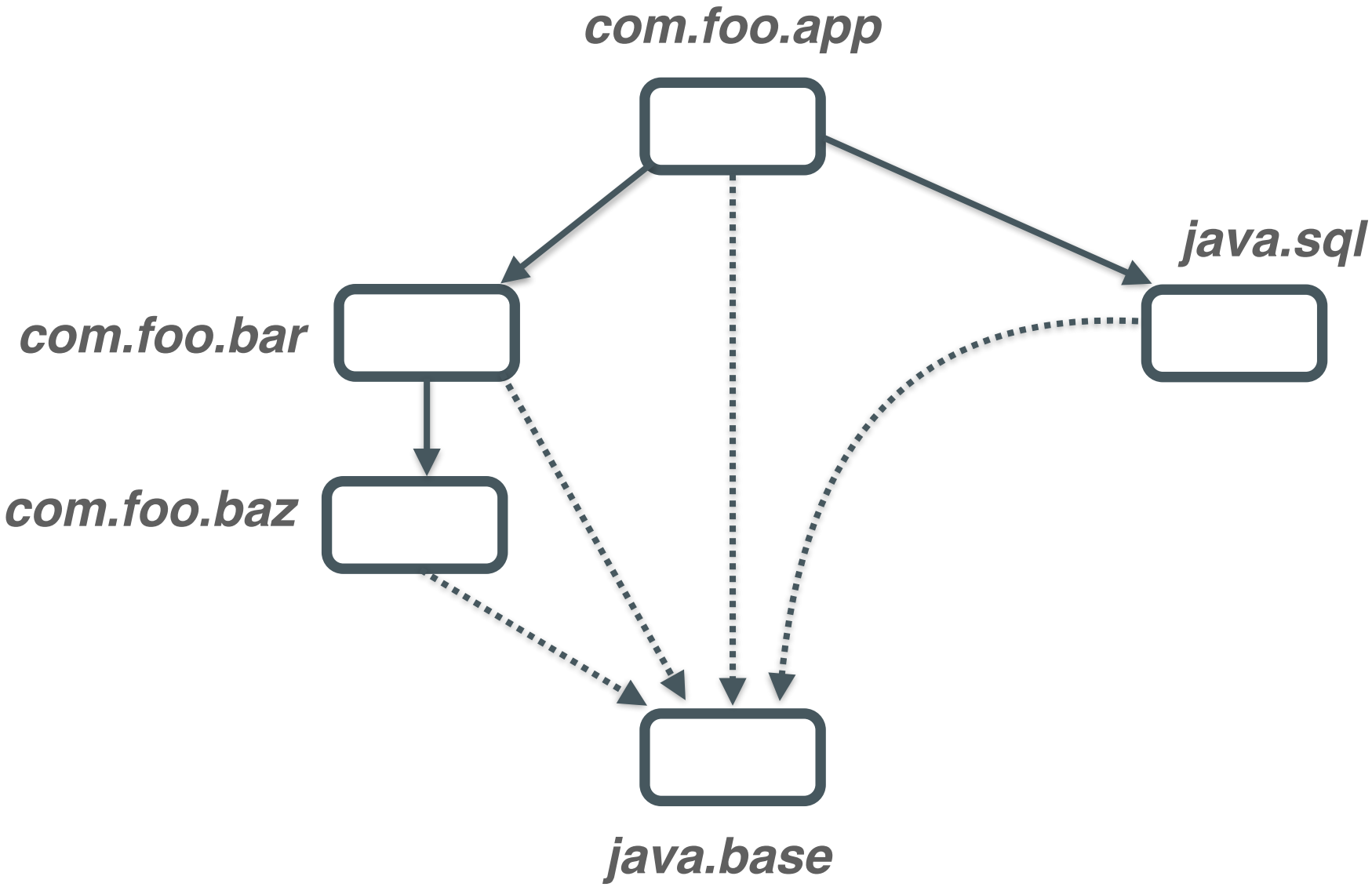
module-info.java
com/foo/bar/alpha/Alpha.java
com/foo/bar/alpha/AlphaFactory.java
com/foo/bar/beta/Beta.java
com/foo/bar/beta/BetaBlocker.java
:

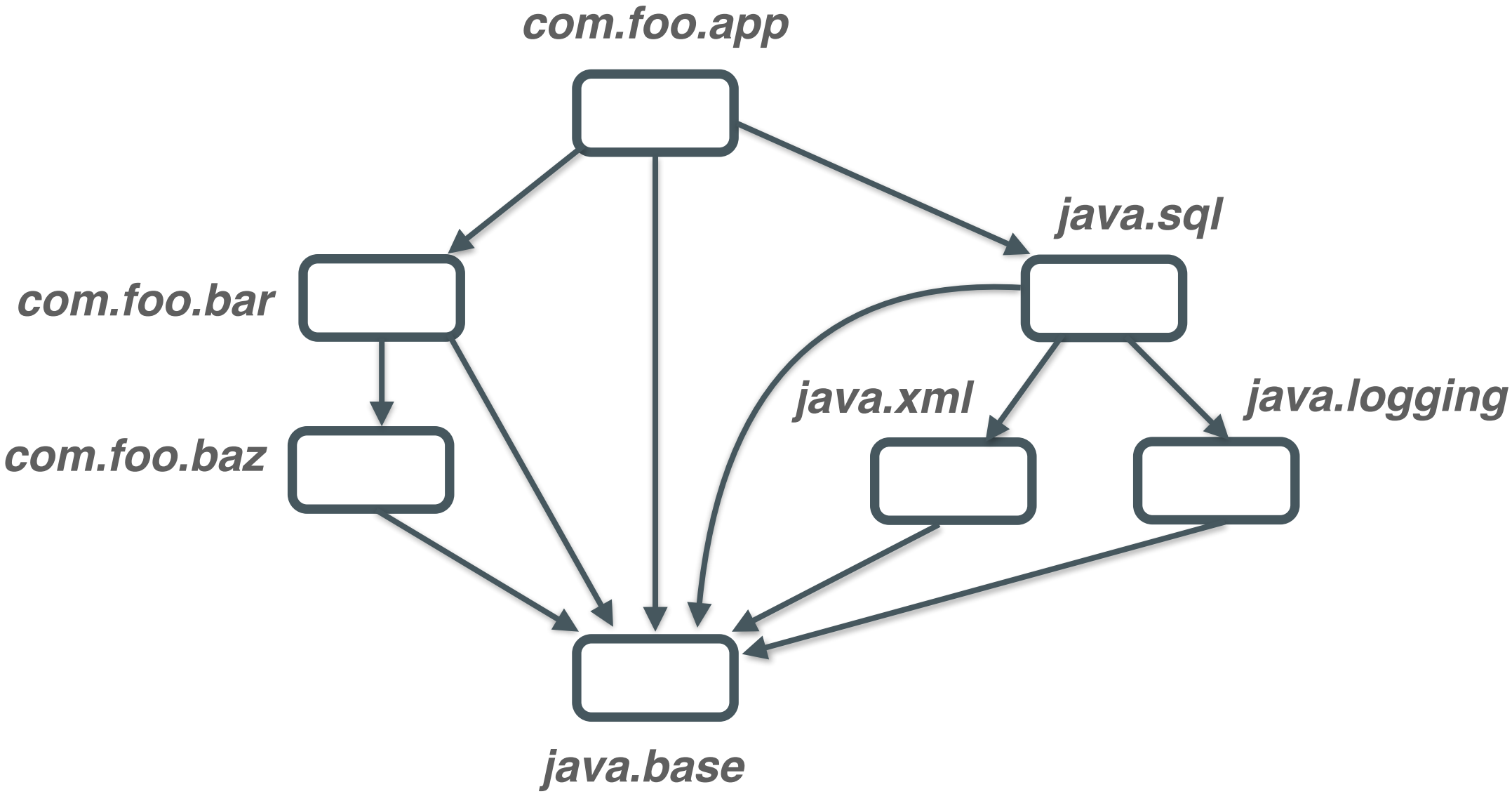
```
module com.foo.bar {
    requires com.foo.baz;
}
```

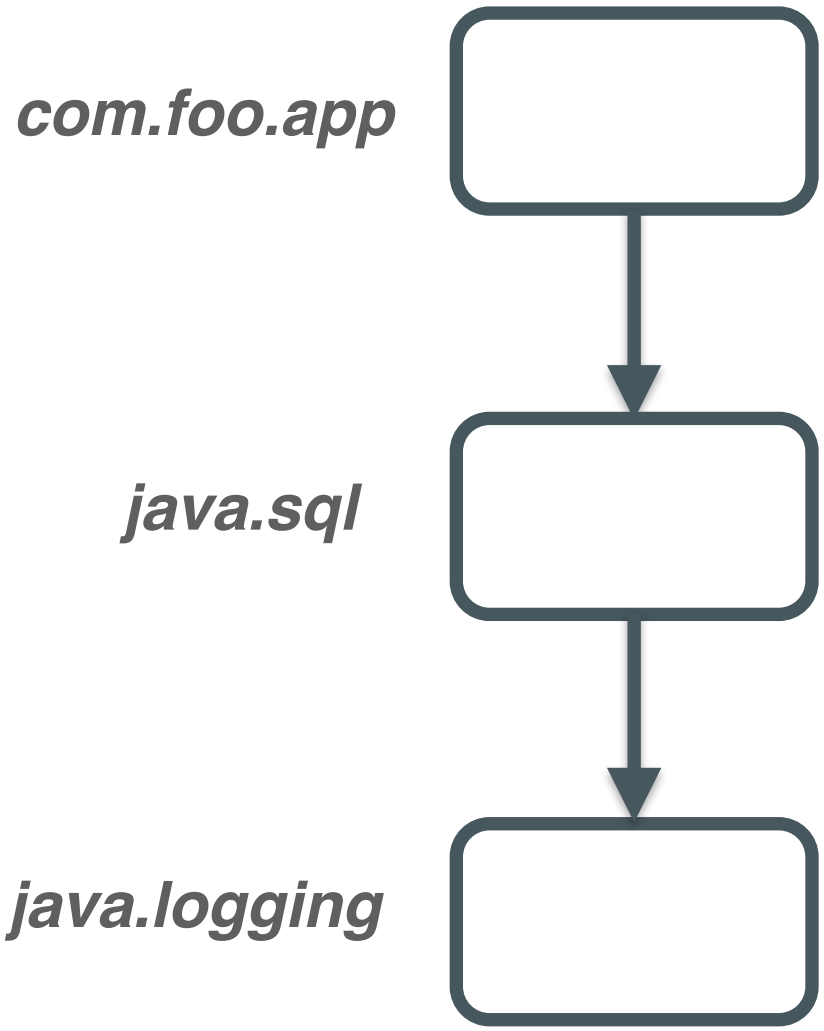


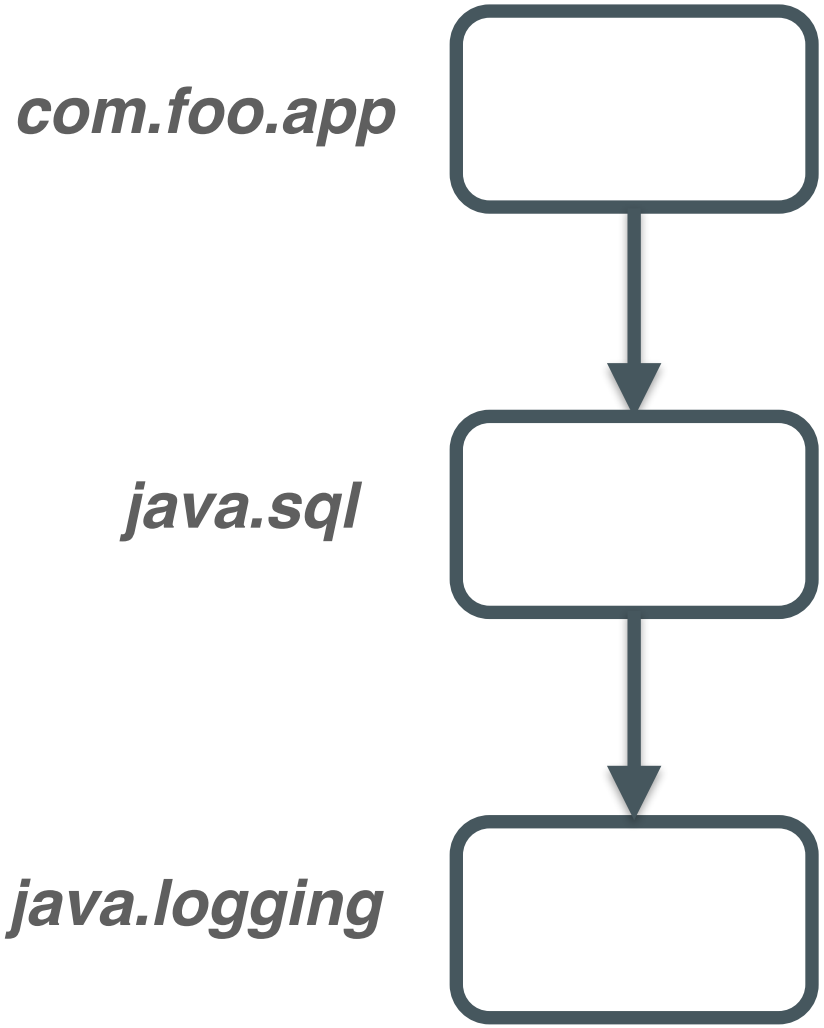
```
module com.foo.app {
    requires com.foo.bar;
    requires java.sql
}
```





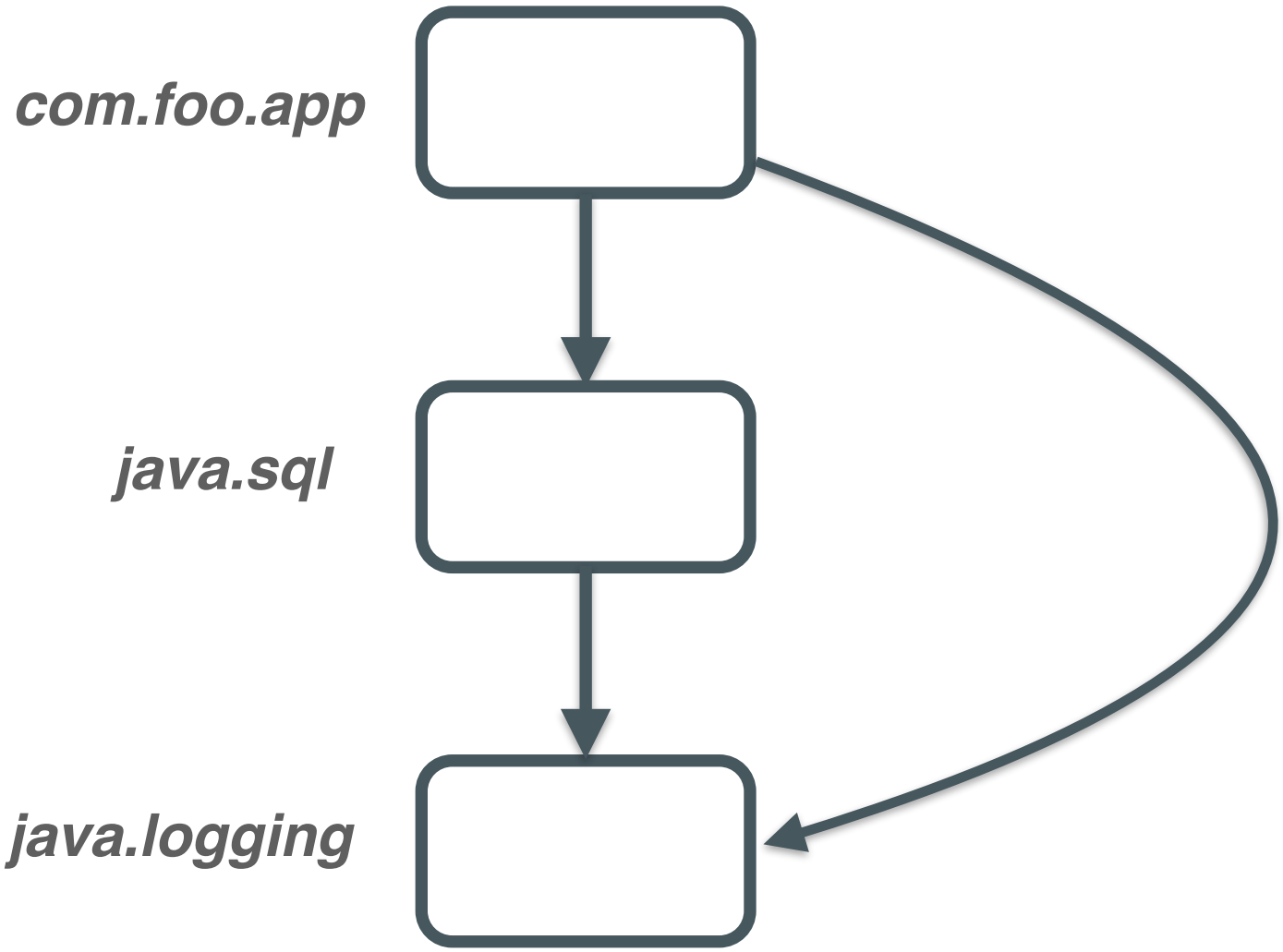


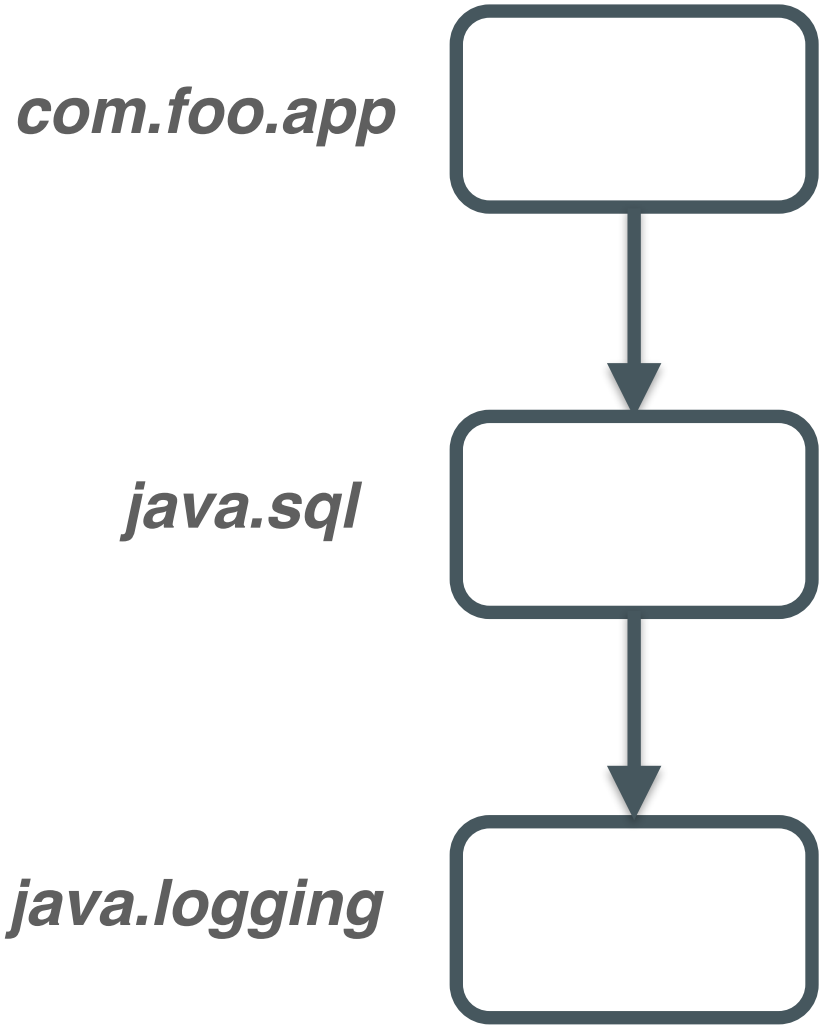




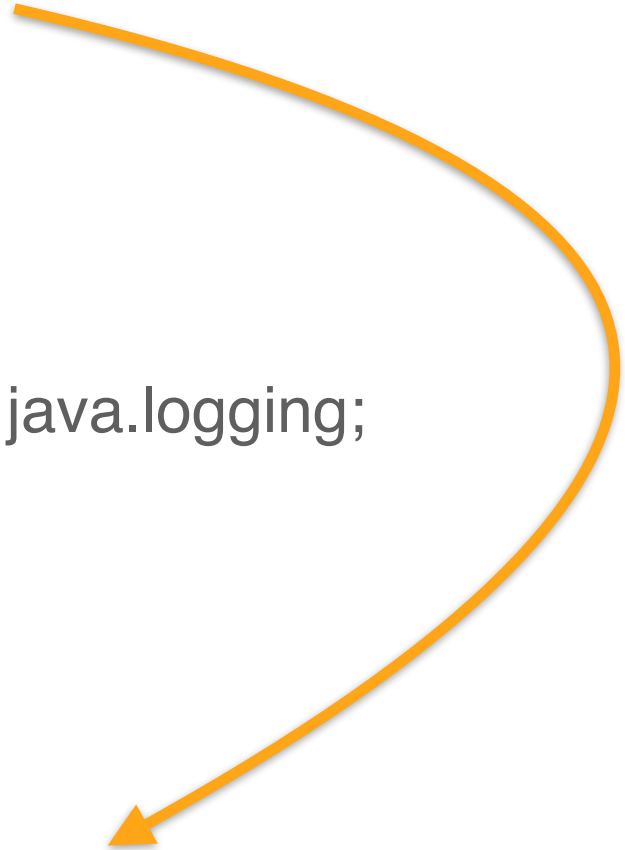
Driver d = …
d.getParentLogger().log(…)

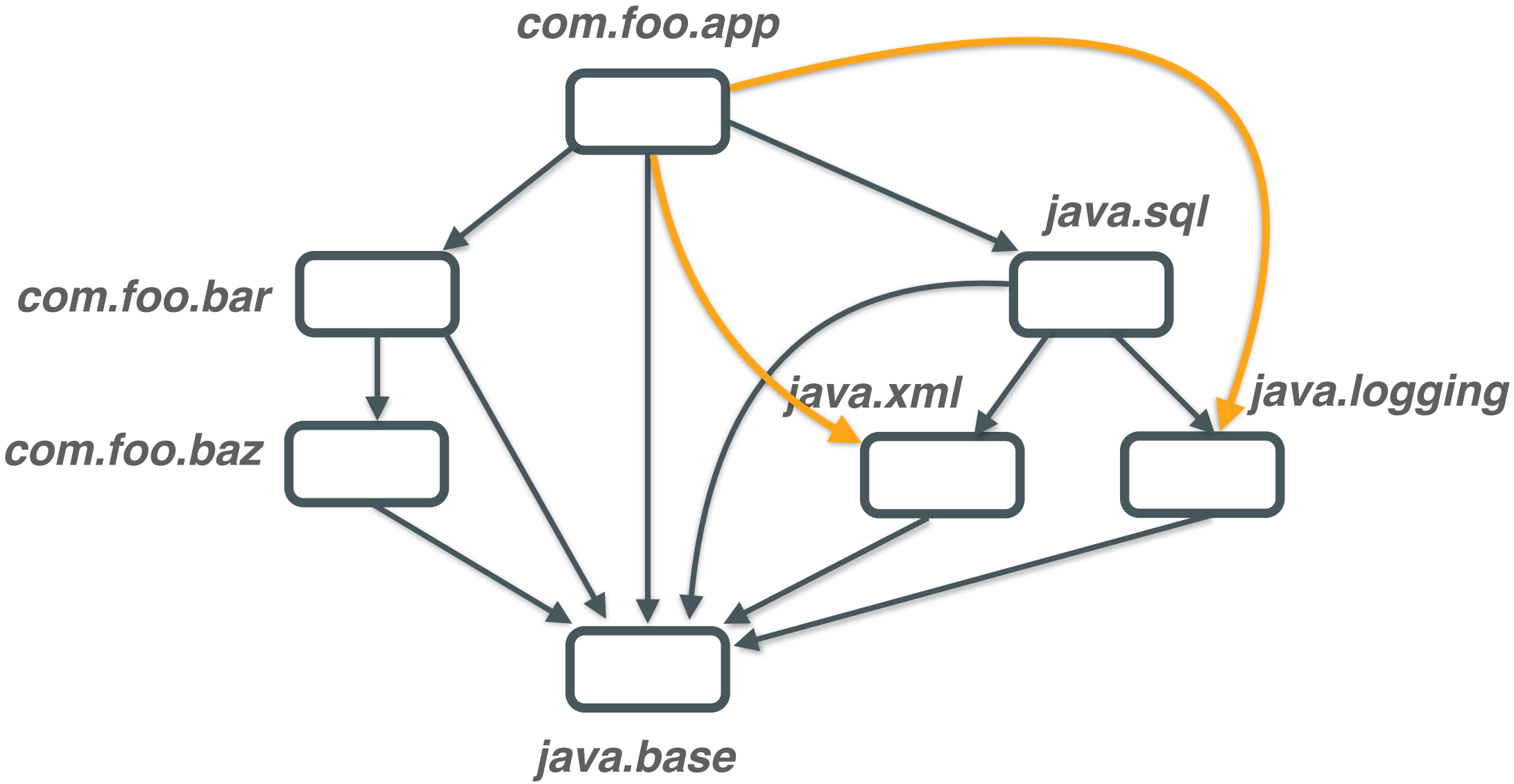
```
package java.sql;
import java.util.logging.Logger;

public class Driver {
    public Logger getParentLogger() { .. }
    :
}
```



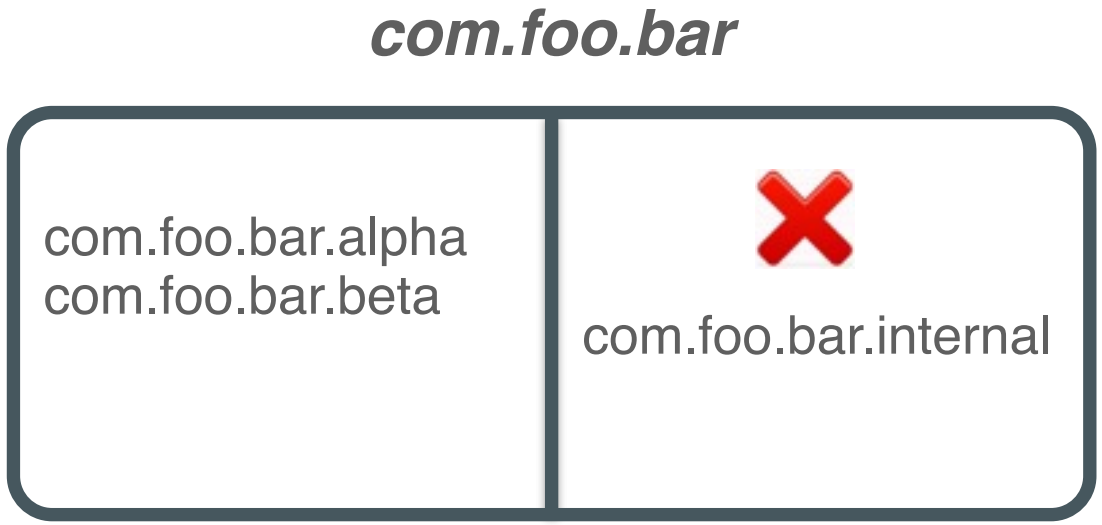



```
module java.sql {
        requires public java.logging;
        :
}
```

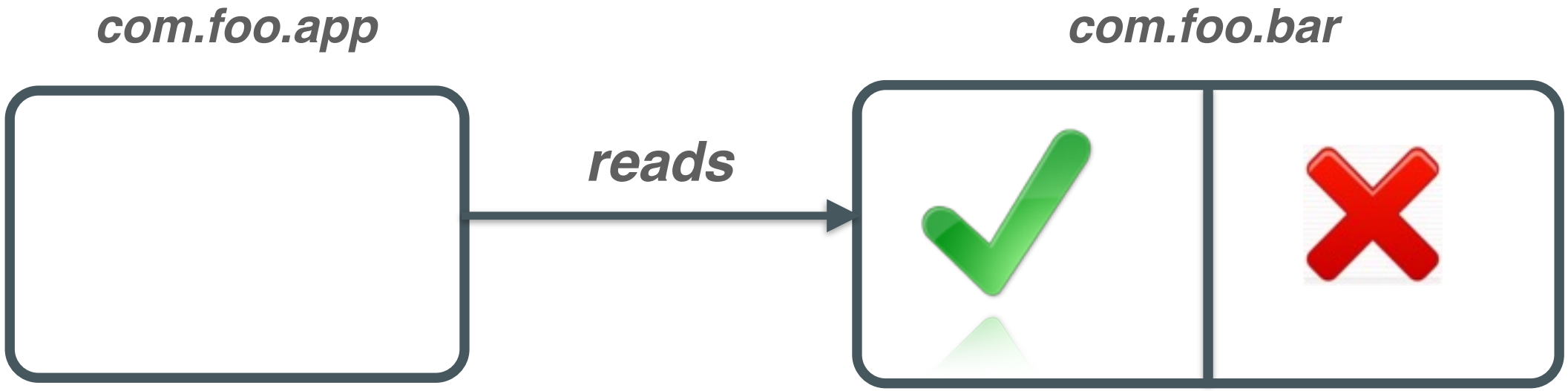




```
module com.foo.bar {
    exports com.foo.bar.alpha;
    exports com.foo.bar.beta;
}
```



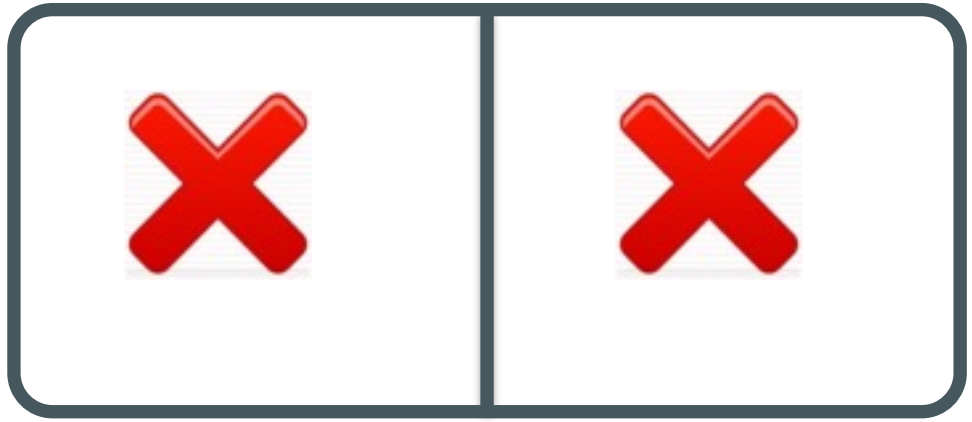
Accessibility



Accessibility

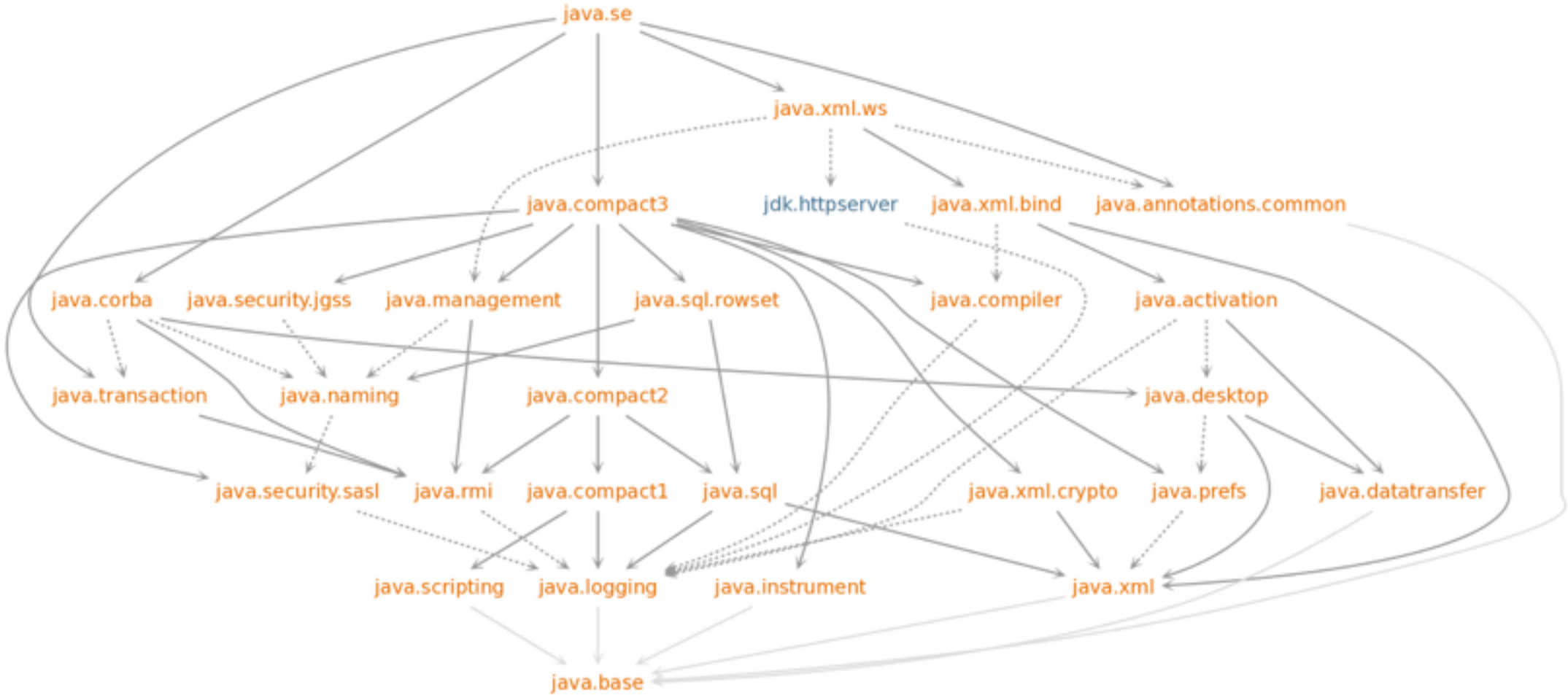
com.foo.app

com.foo.bar



public ⇏ accessible

Platform modules



$ java -listmods

```
java.activation@9
java.annotations.common@9
java.base@9
java.compact1@9
java.compact2@9
java.compact3@9
java.compiler@9
java.corba@9
java.datatransfer@9
java.desktop@9
java.instrument@9
java.logging@9
java.management@9
java.naming@9
java.prefs@9
java.rmi@9
java.scripting@9
java.se@9
java.security.jgss@9
java.security.sasl@9
java.smartcardio@9
java.sql@9
java.sql.rowset@9
java.transaction@9
java.xml@9
java.xml.bind@9
java.xml.crypto@9
java.xml.ws@9
jdk.attach@9
jdk.charsets@9
jdk.compiler@9
jdk.crypto.ec@9
jdk.crypto.pkcs11@9
jdk.hotspot.agent@9
jdk.httpserver@9
jdk.jartool@9
jdk.javadoc@9
jdk.jcmd@9
jdk.jconsole@9
jdk.jdeps@9
jdk.jdi@9
jdk.jdwp.agent@9
jdk.jlink@9
jdk.jvmstat@9
jdk.localedata@9
jdk.management@9
jdk.naming.dns@9
jdk.naming.rmi@9
jdk.pack200@9
jdk.policytool@9
jdk.rmic@9
jdk.scripting.nashorn@9
jdk.sctp@9
jdk.security.auth@9
jdk.security.jgss@9
jdk.xml.bind@9
jdk.xml.dom@9
jdk.xml.ws@9
jdk.zipfs@9
```

Compilation

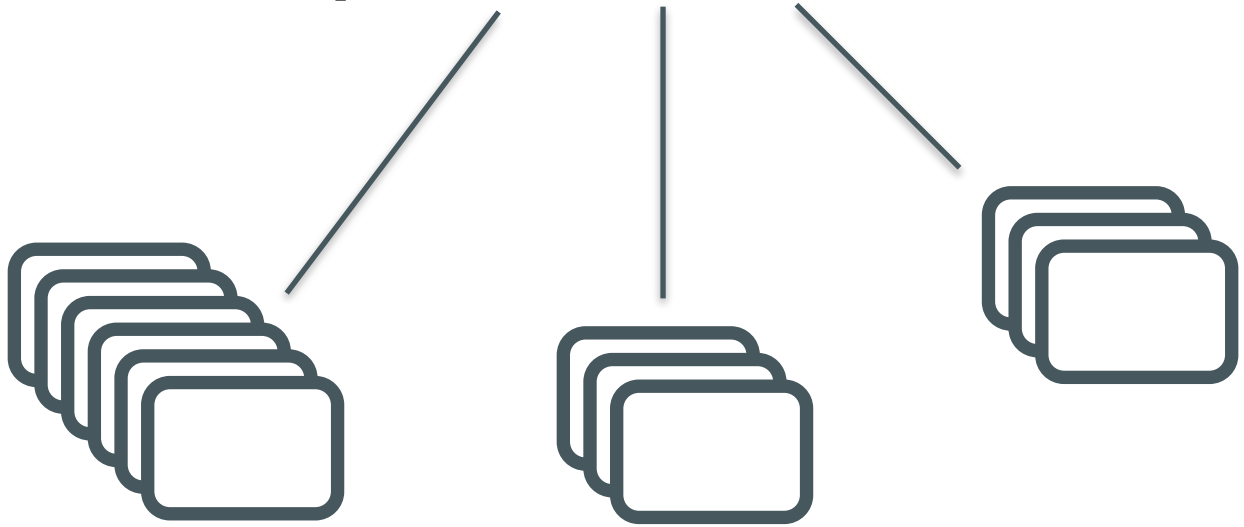
```
$ javac -d mods/com.foo.baz \
    src/com.foo.baz/module-info.java \
    src/com.foo.baz/com/foo/baz/Bazooka.java \
    :
```

src/com.foo.baz/module-info.java
src/com.foo.baz/com/foo/baz/Bazooka.java

mods/com.foo.baz/module-info.class
mods/com.foo.baz/com/foo/baz/Bazooka.class

module path

$ java **-modulepath dir1:dir2:dir3** …



Compilation

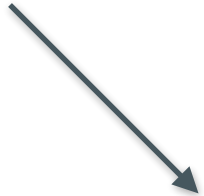
```
$ javac -modulepath mods -d mods/com.foo.bar \
    src/com.foo.bar/module-info.java \
    src/com.foo.bar/com/foo/bar/alpha/Alpha.class \
    :
```

src/com.foo.bar/module-info.java
src/com.foo.bar/com/foo/bar/alpha/Alpha.java
:

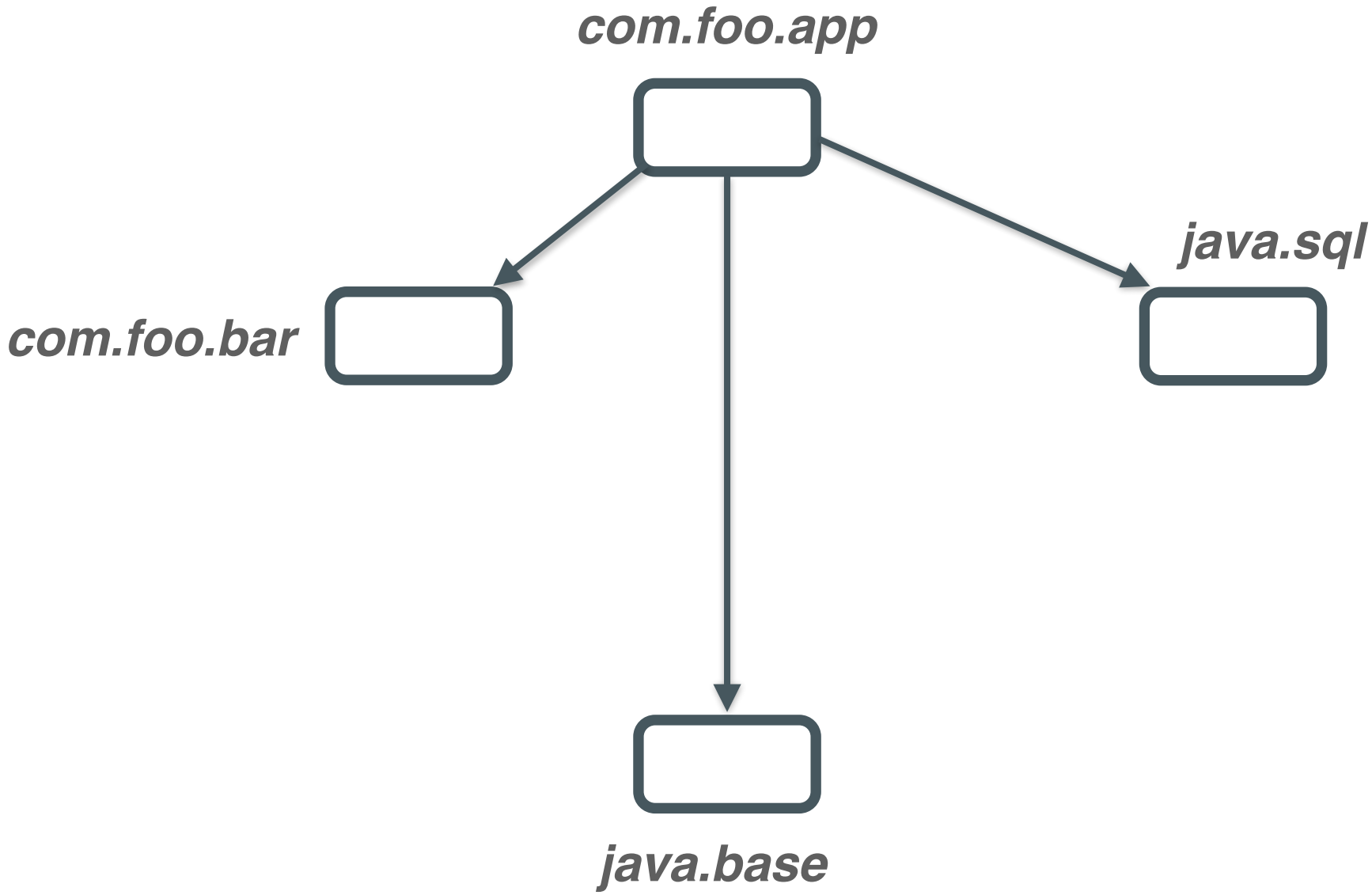
mods/com.foo.bar/module-info.class
mods/com.foo.bar/com/foo/bar/bar/Alpha.class
:

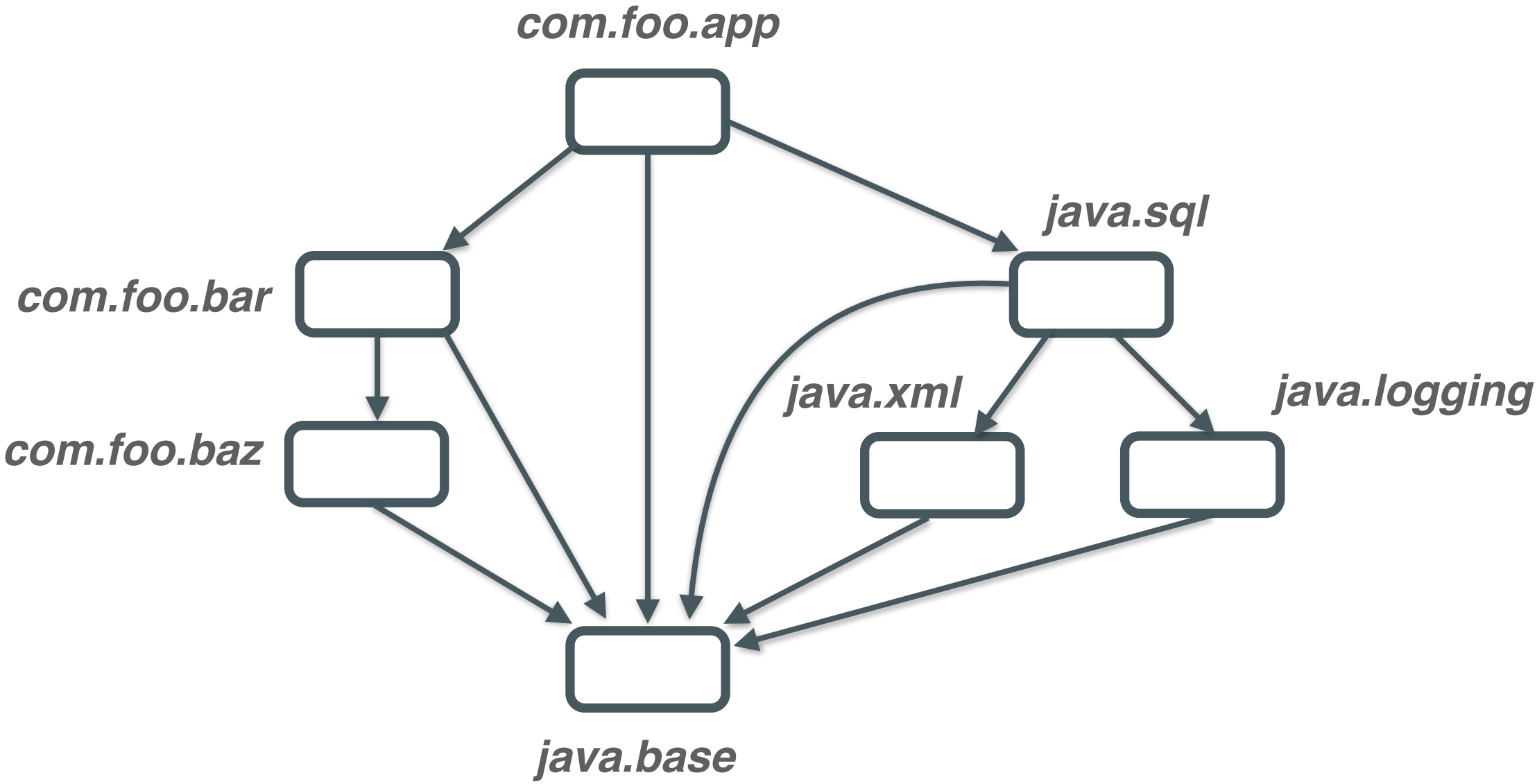
module name

main class



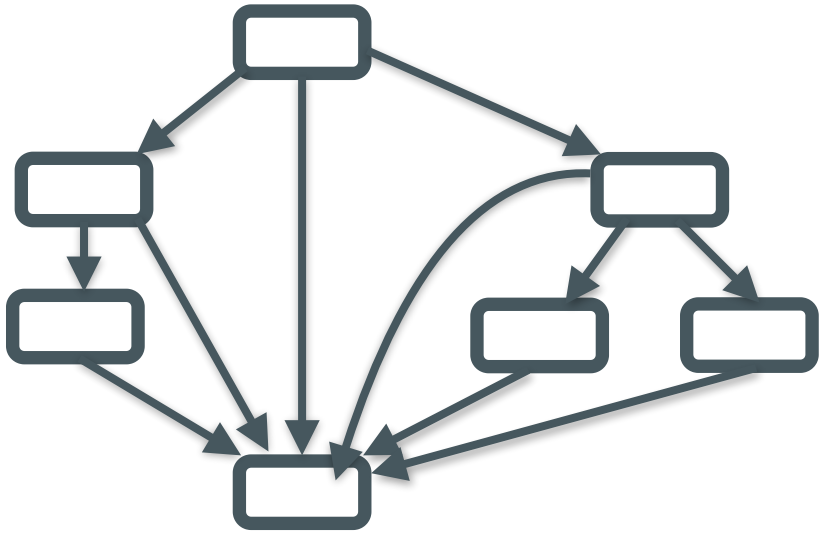
```
$ java -modulepath mods -m com.foo.app/com.foo.app.Main
Greetings from module com.foo.app!
```





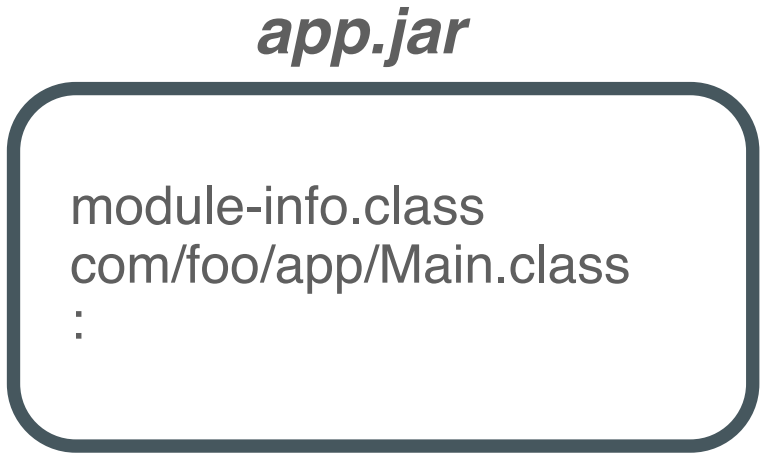
$ java **-Xdiag:resolver** -mp mods -m com.foo.app/com.foo.app.Main

```
[Resolve] Root module com.foo.app located
[Resolve]   (file:///d/mods/com.foo.app/)
[Resolve] Module com.foo.bar located, required by com.foo.app
[Resolve]   (file:///d/mods/com.foo.bar/)
[Resolve] Module java.base located, required by com.foo.app
[Resolve]   (jrt:/java.base)
[Resolve] Module java.sql located, required by com.foo.app
[Resolve]   (jrt:/java.sql)
[Resolve] Module com.foo.baz located, required by com.foo.bar
[Resolve]   (file:///d/mods/com.foo.baz/)
[Resolve] Module java.logging located, required by java.sql
[Resolve]   (jrt:/java.logging)
[Resolve] Module java.xml located, required by java.sql
[Resolve]   (jrt:/java.xml)
[Resolve] Resolve completed
[Resolve]   com.foo.app
[Resolve]   com.foo.bar
[Resolve]   com.foo.baz
[Resolve]   java.base
[Resolve]   java.logging
[Resolve]   java.sql
[Resolve]   java.xml
Greetings from module com.foo.app!
```



Packaging as modular JAR

mods/com.foo.app/module-info.class
mods/com.foo.app/com/foo/app/Main.class
:




```
$ jar --create --file mlib/app.jar \
    --main-class com.foo.app.Main \
     -C mods/com.foo.app .
```

```
$ jar --file mlib/app.jar -p

Name:
   com.foo.app
Requires:
   com.foo.bar
   java.base [ MANDATED ]
   java.sql
Main class:
   com.foo.app.Main
```

$ java **-mp mlib:mods** -m com.foo.app
Greetings from module com.foo.app!

classpath

com.foo.baz (mlib/baz.jar)

```
module-info.class
com/foo/baz/Bazooka.class
:
```

```
$ java -cp mlib/baz.jar:app.jar \
    com.foo.app.Main
```

module path

com.foo.baz (mlib/baz.jar)

```
module-info.class
com/foo/baz/Bazooka.class
:
```

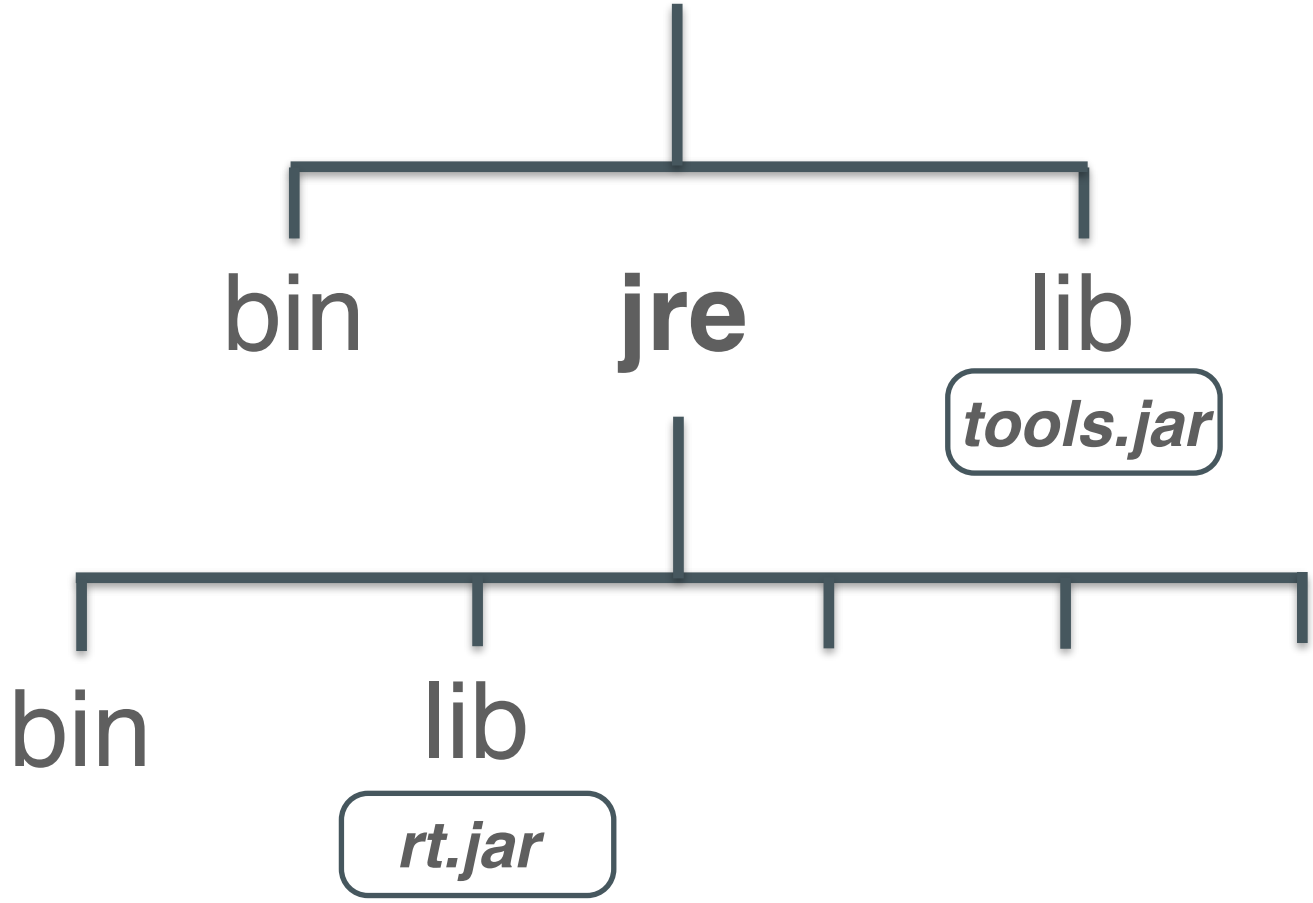
```
$ java -mp mlib -m myapp
```

classpath and module path

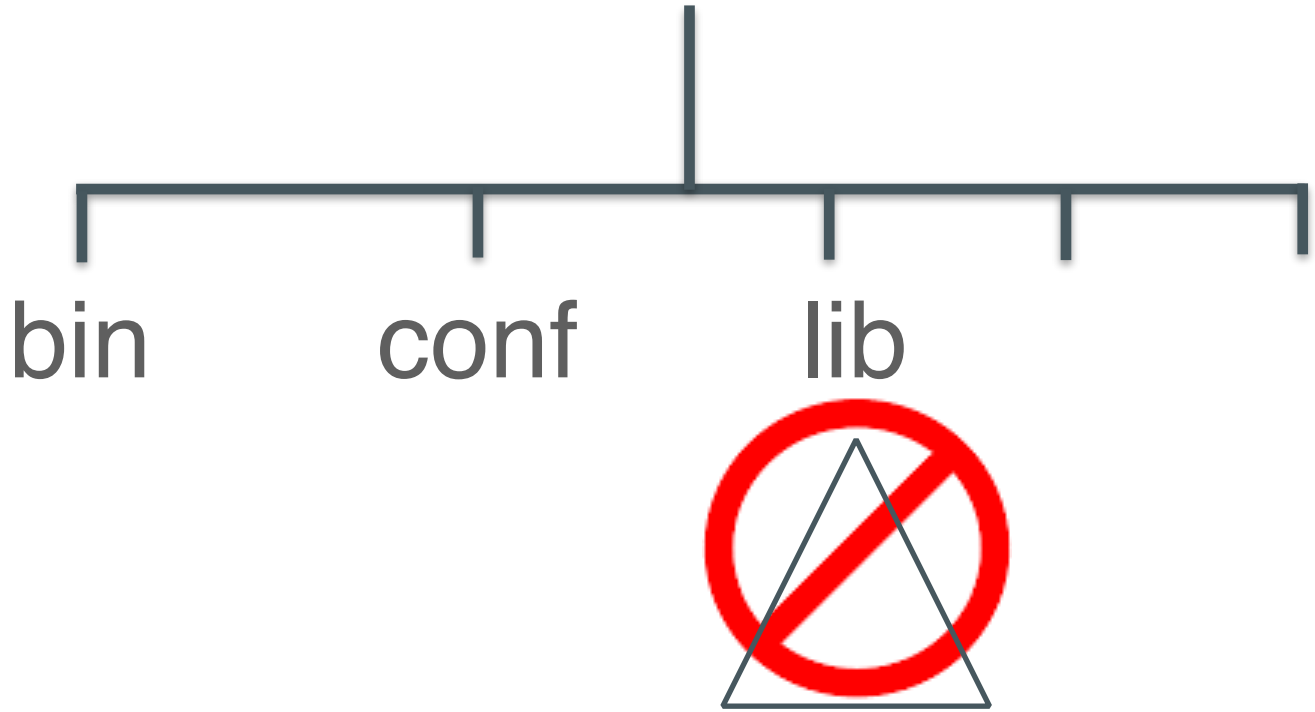
```
$ java -mp mlib -addmods com.foo.baz \
    -cp app.jar com.foo.app.Main
```

Linking

Legacy JDK image



Modular run-time image

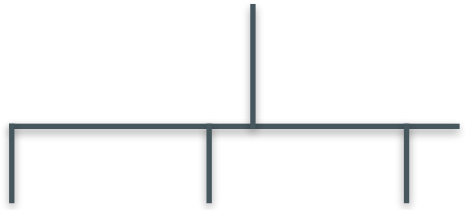


~~jre directory~~
~~rt.jar~~
~~tools.jar~~

Linking

jlink

modular
run-time
image



bin conf …

Linking

```
$ jlink --modulepath $JDKMODS --addmods java.base --output myimage

$ myimage/bin/java -listmods
java.base@9
```

Linking

```
$ jlink --modulepath $JDKMODS:$MYMODS --addmods com.foo.app --output myimage

$ myimage/bin/java -listmods
java.base@9
java.logging@9
java.sql@9
java.xml@9
com.foo.app
com.foo.bar
com.foo.baz
```

Summary

- Introduced basic concepts
- Introduced basic command lines to compile, run, package and link

More Information

OpenJDK Project Jigsaw page, this has links to all the JEPs

http://openjdk.java.net/projects/jigsaw/

mailto:jigsaw-dev@openjdk.java.net

Early Access Builds

http://openjdk.java.net/projects/jigsaw/ea

Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.


ORACLE®

Multi-module compilation

$ javac **-modulesourcepath src** -d mods $(find src -name “*.java”)

src/com.foo.app/module-info.java
src/com.foo.app/com/foo/app/Main/java
src/com.foo.bar/module-info.java
src/com.foo.bar/com/foo/bar/alpha/Alpha.java
src/com.foo.baz/module-info.java
src/com.foo.baz/com/foo/baz/Bazooka.java

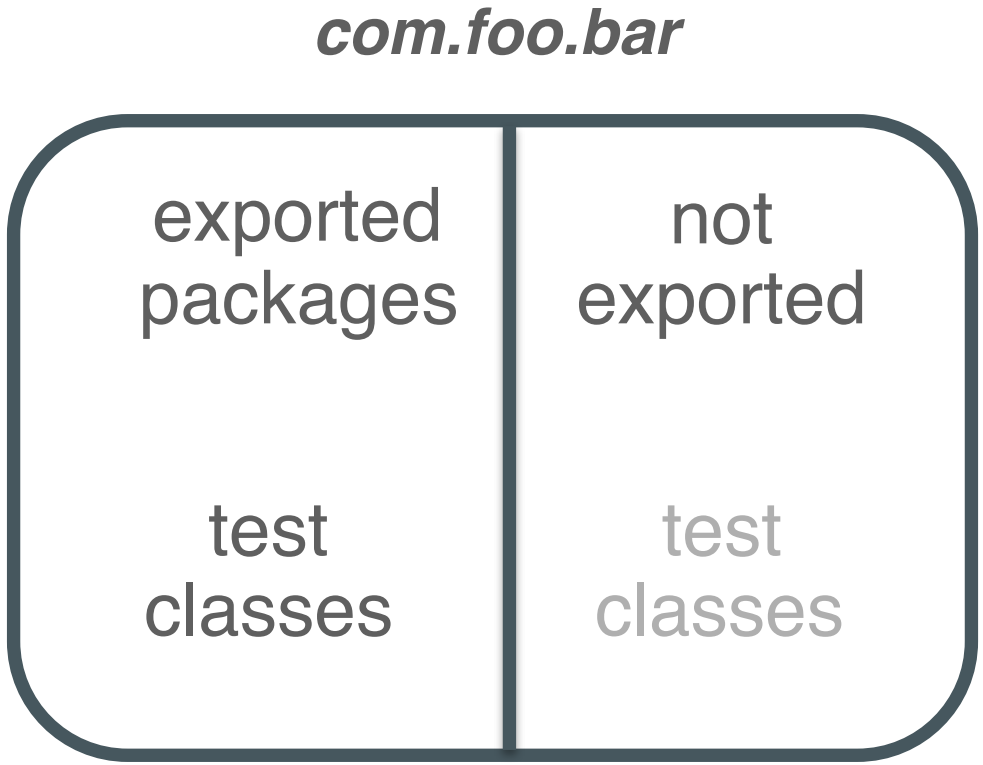
:

mods/com.foo.app/module-info.class
mods/com.foo.app/com/foo/app/Main.class
mods/com.foo.baz/module-info.class
mods/com.foo.baz/com/foo/baz/Bazooka.class
mods/com.foo.bar/module-info.class
mods/com.foo.bar/com/foo/bar/bar/Alpha.class
:

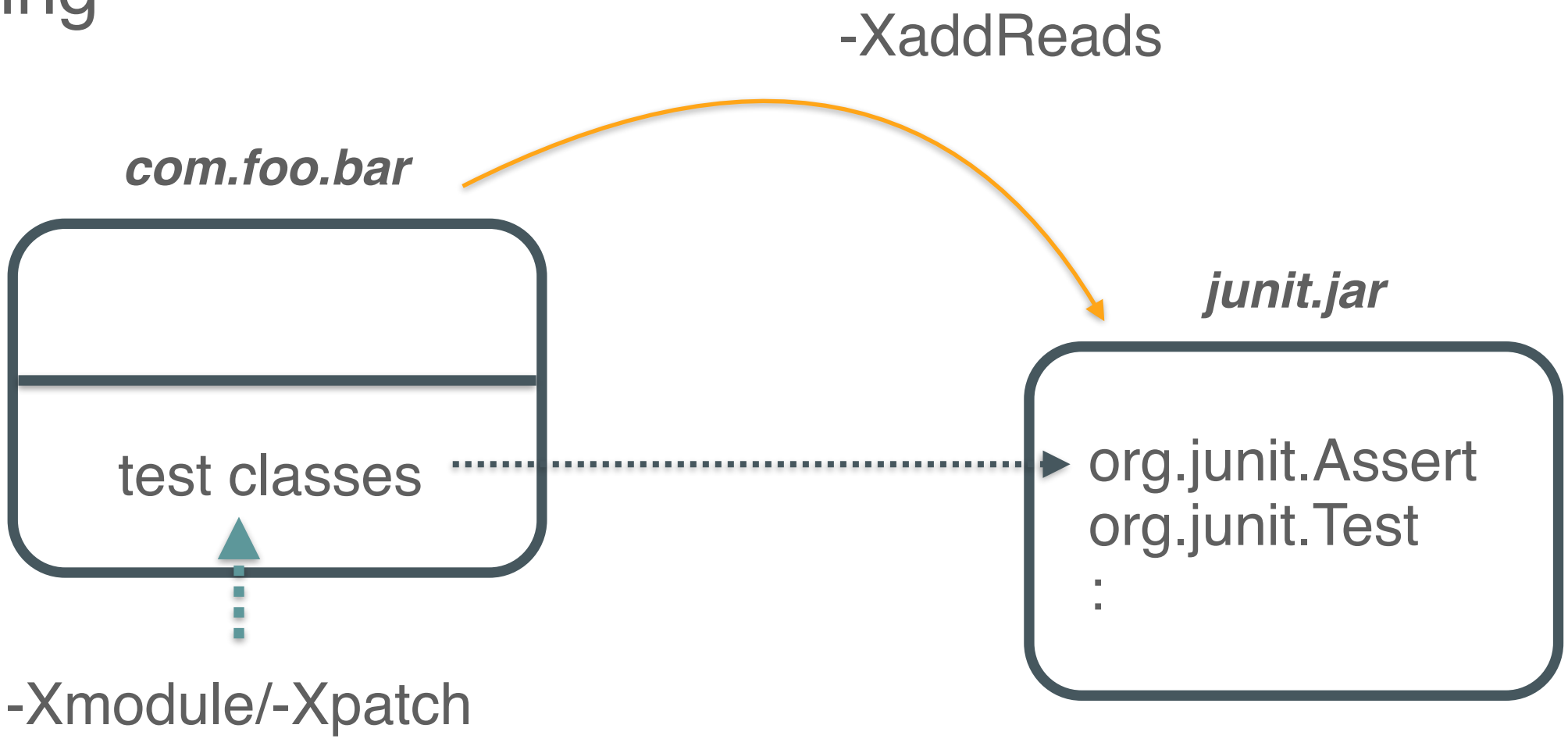
Testing

```
$ javac -cp lib/junit.jar -mp mods -d testclasses \
    test/com/foo/bar/alpha/test/AlphaTest.java

$ java -cp lib/junit.jar:testclasses -mp mods -addmods com.foo.bar \
    org.junit.runner.JUnitCore com.foo.bar.alpha.test.AlphaTest
```

Testing



Testing



Testing

```
$ javac -cp lib/junit.jar -mp mods -d testmods/com.foo.bar
    -Xmodule:com.foo.bar \
    test/com.foo.bar/com/foo/bar/alpha/AlphaTest.java

$ java -cp lib/junit.jar -mp mods -addmods com.foo.bar \
    -Xpatch:testmods -XaddReads:com.foo.bar=ALL-UNNAMED \
    org.junit.runner.JUnitCore com.foo.bar.alpha.AlphaTest
```