

# Modules - commands

---

## Compiling Module

---

Extended Form

```
javac --module-path mods -d feeding feeding/zoo/animal/feeding/*.java
```

The `-d` option is used to specify the destination directory for the compiled Java class files.

Abbreviated Form

```
javac -p mods -d feeding feeding/zoo/animal/feeding/*.java feeding/r
```

### with classpath

```
--class-path <path>, -classpath <path>, -cp <path>
```

WARN: All of these are equivalent!

1. **--class-path**
2. **-classpath**
3. **-cp**

### add-exports

If you must use an internal API that has been made inaccessible by default, then you can break encapsulation using the `--add-exports` command-line option.

```
javac -p out --add-exports moduleA/com.example.util=moduleB -d out/r
```

## Packaging modules

---

```
jar -cvf mods/zoo.visitor.jar -C consumerModule .
```

## Running module

---

Extended Form

```
java --module-path mods --module zoo.animal.feeding/zoo.animal.feedi
```

Abbreviated Form

```
java -p mods -m zoo.animal.feeding/zoo.animal.feeding.Task
```

### show-module-resolution

show module resolution output during startup

```
java --show-module-resolution -p mods -m zoo.visitor/zoo.visitor.Tou
```

When this option is present, the JVM will output information about how it resolved module dependencies.

### describe modules

---

```
java -p mods -d moduleName  
java --module-path mods --describe-module moduleName
```

### Example

extended:

```
java --module-path mods --describe-module zoo.tours.api
```

compact:

```
java -p mods -d zoo.tours.api
```

output

```
zoo.tours.api file:///Users/enrico/github/ocp17/sybex-120-829-chapter-12
exports zoo.tours.api
requires java.base mandated
```

## list

---

```
java --list-modules
```

## describe-module - jar

---

```
jar --file mods/zoo.staff.jar --describe-module
#equivalent
jar --file mods/zoo.staff.jar -d
```

## jdeps

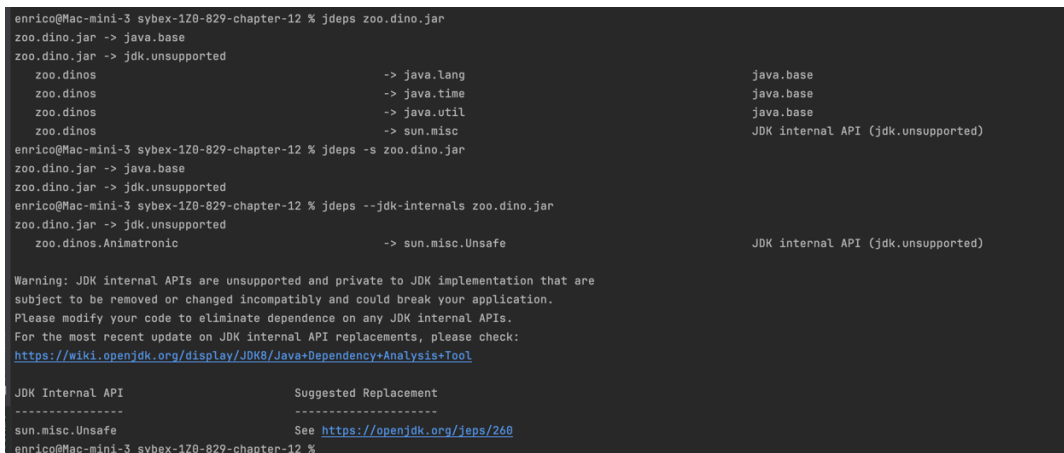
---

The jdeps command lists information about dependencies within a module.

```
jdeps -s zoo.dino.jar
```

- `-s / -summary` - Summarizes output
- `--jdk-internals / -jdkinternals` - The `jdeps -jdkinternals` option includes analysis of dependencies on JDK internal APIs

## Screenshot jdeps commands



```
enrico@Mac-mini-3 sybex-120-829-chapter-12 % jdeps zoo.dino.jar
zoo.dino.jar -> java.base
zoo.dino.jar -> jdk.unsupported
zoo.dinos -> java.lang java.base
zoo.dinos -> java.time java.base
zoo.dinos -> java.util java.base
zoo.dinos -> sun.misc JDK internal API (jdk.unsupported)
enrico@Mac-mini-3 sybex-120-829-chapter-12 % jdeps -s zoo.dino.jar
zoo.dino.jar -> java.base
zoo.dino.jar -> jdk.unsupported
enrico@Mac-mini-3 sybex-120-829-chapter-12 % jdeps --jdk-internals zoo.dino.jar
zoo.dino.jar -> jdk.unsupported
zoo.dinos.Animatronic -> 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.org/display/JDK8/Java+Dependency+Analysis+Tool

JDK Internal API      Suggested Replacement
-----
sun.misc.Unsafe      See https://openjdk.org/jeps/260
enrico@Mac-mini-3 sybex-120-829-chapter-12 %
```

## jlink

---

extended

```
jlink --module-path mods --add-modules zoo.animal.talks --output zoo
```

Abbreviated

```
jlink -p mods --add-modules zoo.animal.talks --output zooApp
```

There is no abbreviated form for the parameter **output**.

- --add-modules - List of modules to package
- --output - Name of output directory

This is the contents of the generated folder:

```
cd zooAPP/  
ls
```

output:

```
bin conf include legal lib man release
```

## jmod

---

```
#generated by CGPT  
jmod create --class-path out/moduleA --module-version 1.0 --module-fo
```

- create
- extract
- describe
- list
- hash