Modules

Named, Automatic and Unnamed Modules

Named Module

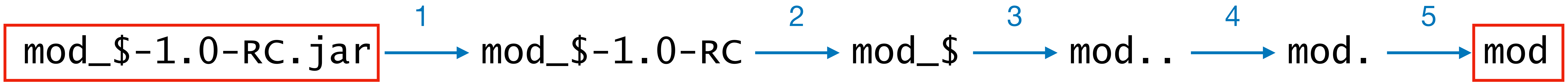
- contains `module-info.java` file
    - this file is in the root of the JAR alongside other package(s)
- module name is specified in `module-info.java`
- named modules appear on the **module path** (not on the classpath!)

Automatic Module

- appear on the **module path**
- it doesn't contain `module-info.java` file
- basically a regular JAR file placed in the module path and treated as a module
- in this case Java *automatically* determines the module name
- unless there is a file called MANIFEST.MF in META-INF folder in .jar file
    - in this file one can set the property `Automatic-Module-Name`
    - this is from the "old days", helping developers to transition from JAR to module
    - in this case, the name of the module is read from property

Algorithm for naming automatic module

1. Remove the file extension from the JAR file name
2. Remove any version information from the end of the name
3. Replace any remaining characters other than letters and numbers with dots
4. Replace any sequences of dots with a single dot
5. Remove the dot if it is the first or last character of the result



`mod_$-1.0-RC.jar` 1 → `mod_$-1.0-RC` 2 → `mod_$` 3 → `mod..` 4 → `mod.` 5 → `mod`

Unnamed Module

- appears on the **classpath** (unlike automatic modules)
- basically a regular JAR in the classpath
- usually doesn't contain `module-info.java` (if does, it is ignored)
- unnamed modules don't export any packages to named or automatic modules
  - it can read from any JARs on the classpath or module path
- **important to remember:**
  - code on the classpath can access the module path
  - code on the module path is unable to read from the classpath