JLINK (Java Linker)

Until 1.8 version to run a small Java program (like Hello World program) also, we should use a bigger JRE which contains all java's inbuilt 4300+ classes. It increases the size of Java Runtime environment and Java applications. Due to this Java is not suitable for IOT devices and Micro Services. (No one invite a bigger Elephant into their small house).

To overcome this problem, Java people introduced Compact Profiles in Java 8. But they didn't succeed that much. In Java 9, they introduced a permanent solution to reduce the size of Java Runtime Environment, which is nothing but JLINK.

JLINK is Java's new command line tool(which is available in JDK_HOME\bin) which allows us to link sets of only required modules (and their dependencies) to create a runtime image (our own JRE).

Now, our Custom JRE contains only required modules and classes instead of having all 4300+ classes.
It reduces the size of Java Runtime Environment, which makes java best suitable for IOT and micro services.

Hence, Jlink's main intention is to avoid shipping everything and, also, to run on very small devices with little memory. By using Jlink, we can get our own very small JRE.
Jlink also has a list of plugins(like compress) that will help optimize our solutions.

How to use JLINK: Demo Program

```
src
 |-demoModule
    |-module-info.java
    |-packA
       |-Test.java
```

module-info.java:

```
1)  module demoModule
2)  {
3)  }
```

Test.java:

```
1)  package packA;
2)  public class Test
3)  {
4)     public static void main(String[] args)
5)     {
6)        System.out.println("JLINK Demo To create our own customized & small JRE");
7)     }
8)  }
```

Compilation:

C:\Users\Durga\Desktop>javac --module-source-path src -d out -m demoModule

Run with default JRE:

C:\Users\Durga\Desktop>java --module-path out -m demoModule/packA.Test

o/p: JLINK Demo To create our own customized & small JRE

Creation of our own JRE only with required modules:

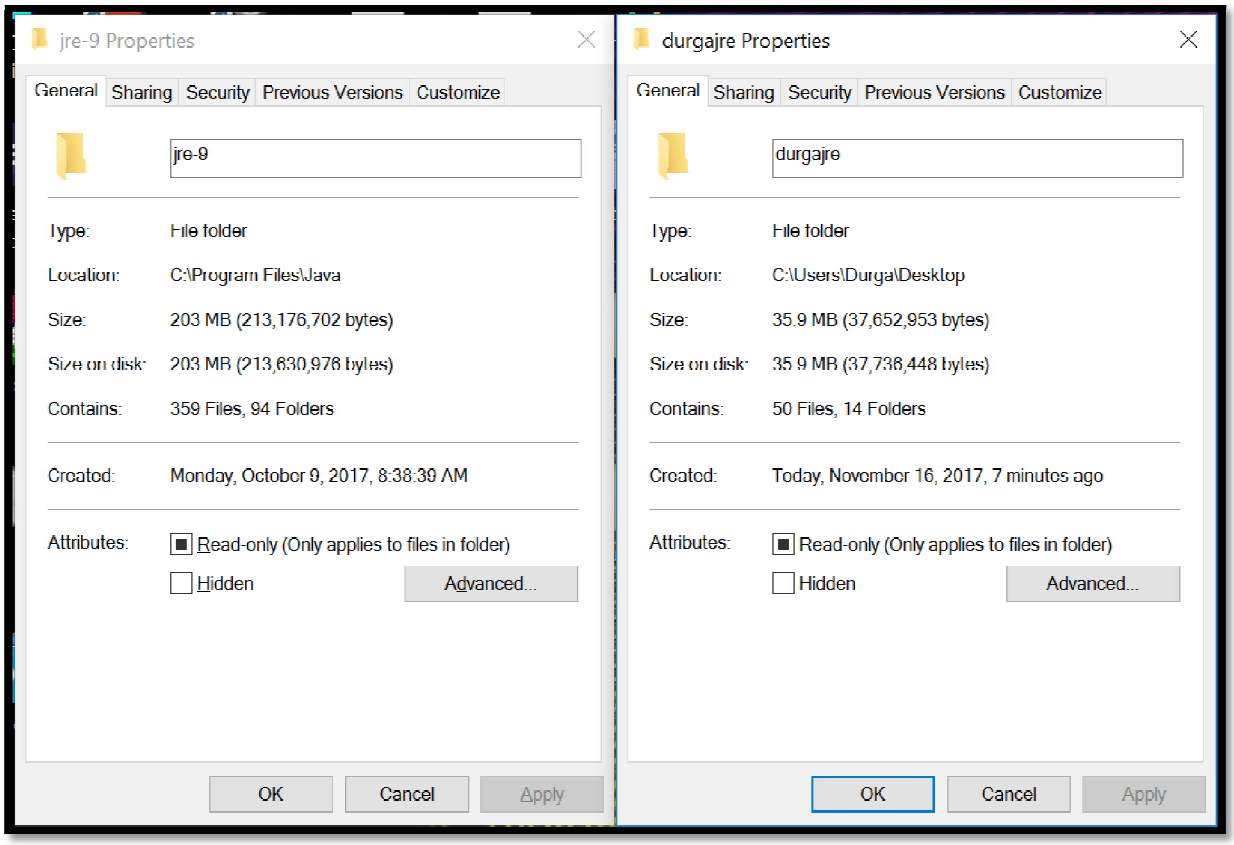
demoModule requires java.base module. Hence add java.base module to out directory (copy java.base.jmod from jdk-9\jmods to out folder)

```
out
 |-java.base.jmod
 |-demoModule
    |-module-info.class
    |-packA
       |-Test.class
```

Now we can create our own JRE with JLINK command

C:\Users\Durga\Desktop>jlink --module-path out --add-modules demoModule,java.base --output durgajre

Now observe the size of durgajre is just 35.9MB which is very small when compared with default JRE size 203MB.

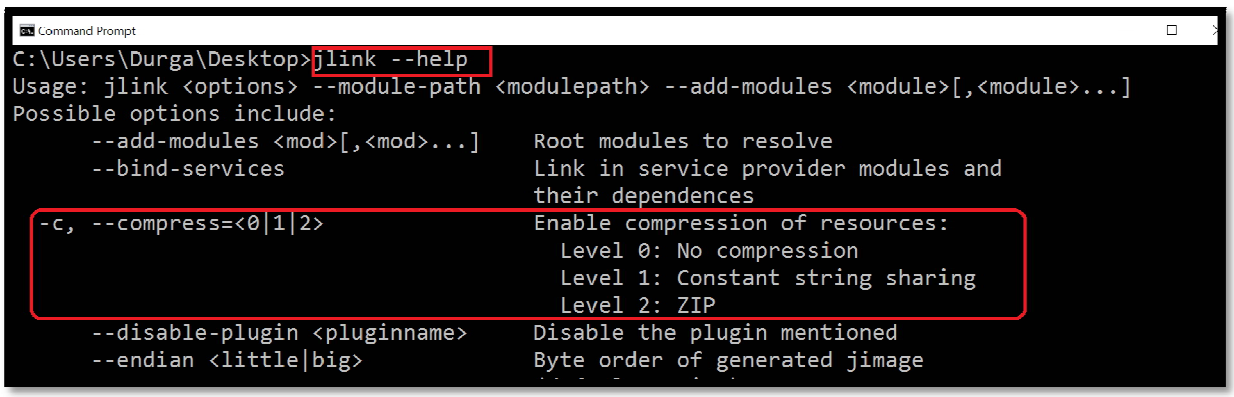


We can run our application with our own custom jre (durgajre) as follows

C:\Users\Durga\Desktop\durgajre\bin>java -m demoModule/packA.Test
o/p: JLINK Demo To create our own customized & small JRE

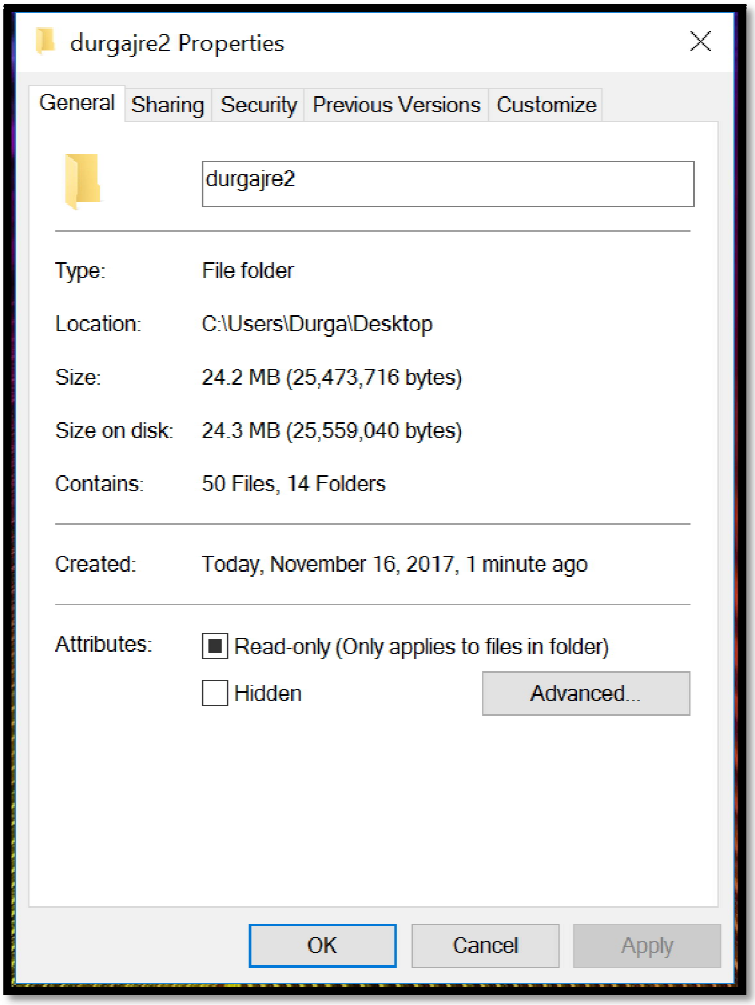
Compressing the size of JRE with compress plugin:

Still we can compress the size of JRE with compress plugin.



C:\Users\Durga\Desktop>jlink --module-path out --add-modules demoModule,java.base --compress 2 --output durgajre2

C:\Users\Durga\Desktop\durgajre2\bin>java -m demoModule/packA.Test
o/p: JLINK Demo To create our own customized & small JRE



Providing our own name to the application with launcher plugin:

C:\Users\Durga\Desktop>jlink --module-path out --add-modules demoModule,java.base --launcher demoapp=demoModule/packA.Test --compress 2 --output durgajre3

Now we can run our application only with the name demoapp

C:\Users\Durga\Desktop\durgajre3\bin>demoapp
JLINK Demo To create our own customized & small JRE

If we set the path PATH = C:\Users\Durga\Desktop\durgajre3\bin

Then we can run our application from anywhere.
D:\>demoapp
E:\>demoapp