

# GraalVM™ Capabilities Cheat sheet

## Run Java Applications

Compile a Java class:

**javac MyApp.java**

Run the application from a JAR file:

**java -jar MyApp.jar**

Specify the class path for the app:

**java -cp target/myapp.jar  
com.mycompany.app.MyApp**

Run the JIT compiler as JAR or native library (default):

**-XX:+UseJVMCINativeLibrary**

Select the GraalVM compiler configuration:

**-Dgraal.CompilerConfiguration=  
enterprise|community|economy**

Print the details for the JIT compiled code:

**-Dgraal.PrintCompilation=true**

Produce the diagnostic data for the compilation:

**-Dgraal.Dump**

Load a javaagent:

**-javaagent:<jarpath>[=<options>]  
-agentlib:<libname>[=<options>]**

## Compile to Native Executables

Install the native image builder from a local file:

**gu install -L native-image.jar**

Native Image command syntax:

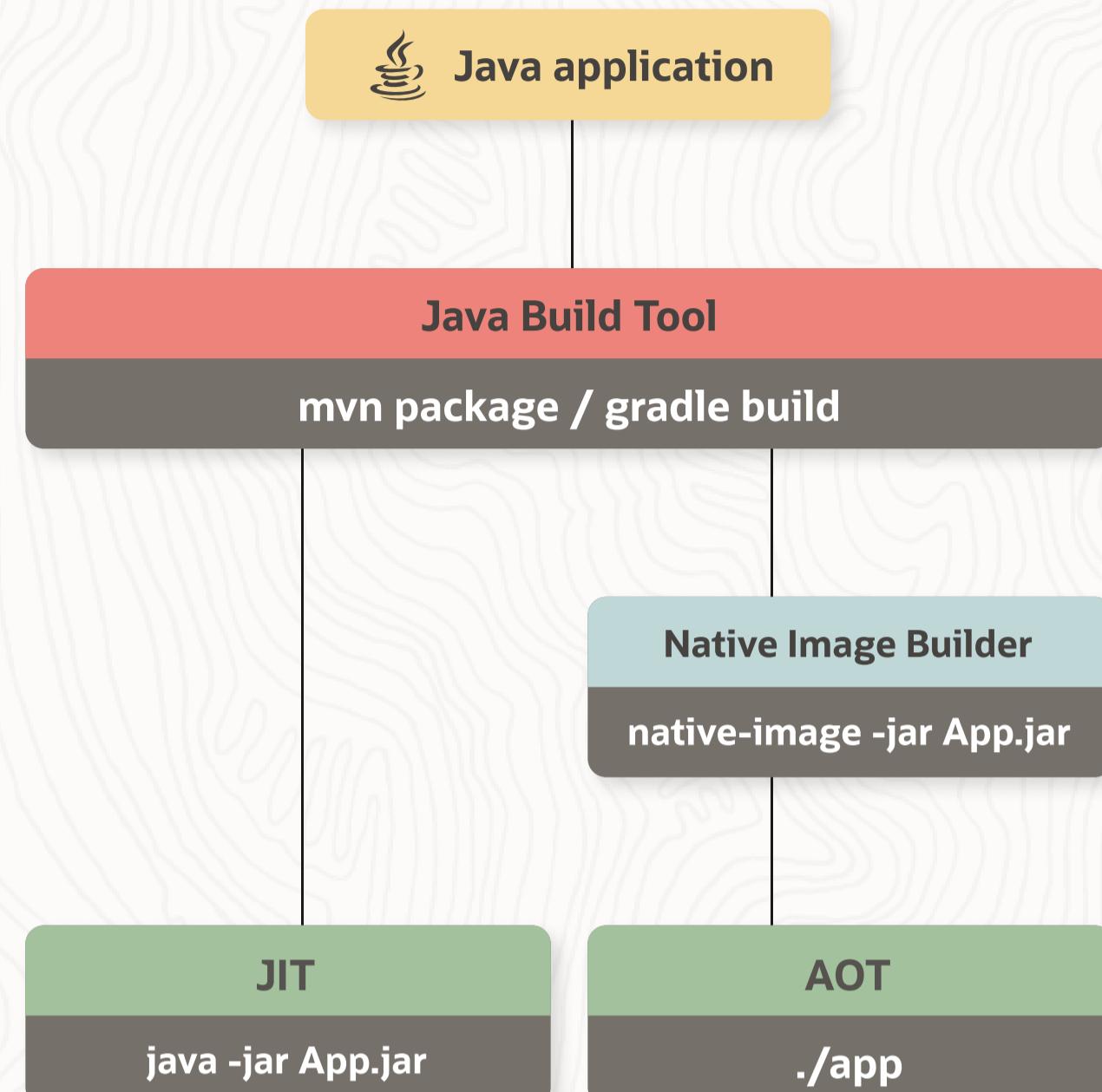
**native-image [options] MyClass**

Build a native image of a JAR file:

**native-image -jar MyApp.jar**

Run a native image:

**./myapp**



Build a shared library:

**--shared**

Build a statically linked native image:

**--static --libc=glibc|musl**

Include a language runtime in the native image:

**--language:js|graalpython|ruby|R|lli**

Use profile-guided optimizations:

**native-image --pgo-instrument MyApp**

**./myapp #and apply load**

**native-image --pgo profile.iprof MyApp**

Attach a debugger:

**--debug-attach=[port]**

Trace classes initialization:

**--trace-class-initialization**

List all image build options for experts:

**--expert-options-all**

## Enable Polyglot Programming

Run a Node.js application:

**node myApp.js**

Run a JavaScript, R, Ruby, Python, LLVM application:

**js myApp.js**

**graalpython myApp.py**

**ruby myApp.rb**

**R myApp.r**

**lli myApp**

Run other languages in a Java application:

```
org.graalvm.polyglot
  .Context
    .createContext()
      .eval("languageId", "code");
```

Enable polyglot capabilities for an application:

**--polyglot --jvm**

Limit resources for the application:

**--sandbox.MaxCPUTime=<ms>**

**--sandbox.MaxStatements=N**

Debug the application:

**--inspect[=[host:]<port number>]**

**--inspect-brk**

Profile the application:

**--cpusampler**

**--cputracer**

**--memtracer**



JS



GraalVM™



WA

