Control class initialization at build or run time:

```
--initialize-at-build-time=<comma-separated list of packages and classes>
```

or

```
--initialize-at-run-time=<comma-separated list of packages and classes>
```

Embed a Software Bill of Materials (SBOM)*:

```
--enable-sbom
```

Enable support for GraalVM languages:

```
--language=<python | js | ruby | R | wasm>
```

Enable Profile-Guided Optimizations (PGO)* to improve performance and throughput:

1. Build an instrumented native executable:

```
native-image --pgo-instrument MyApp
```

2. Run the executable to record profiles:

```
./myapp
```

3. Build an optimized native executable:

```
native-image --pgo=default.prof MyApp
```

List all options for native-image:

```
--help or --help-extra
```

**INTEGRATIONS**

Use the **Gradle** plugin for GraalVM Native Image:

```
https://graalvm.github.io/native-build-tools/latest/gradle-plugin
```

Use the **Maven** plugin for GraalVM Native Image:

```
https://graalvm.github.io/native-build-tools/latest/maven-plugin
```

Use the **GitHub Action** for GraalVM:

```
https://github.com/marketplace/actions/github-action-for-graalvm
```

**CONFIGURE**

There are three ways to configure Native Image for external Java libraries:

1. Configuration is provided by a framework such as **Micronaut**, **Helidon**, **Spring**, and **Quarkus**.
2. Configuration is automatically pulled in by the Maven/Gradle plugin from **GraalVM Reachability Metadata Repository**.
3. Configuration is provided manually. The Tracing Agent can help in this process.

```
java -agentlib:native-image-agent=config_output-dir=/path/to/config-dir/ -jar MyApp.jar
```

b. Build a native executable with the configuration:

```
native-image -jar MyApp.jar
```

**MONITOR**

Tune the garbage collector:

```
./myapp -Xmx<m> -Xmn<m> ...
```

Gather garbage collection logs:

```
./myapp -XX:+PrintGC -XX:+VerboseGC
```

Prerequisite: Enable VM inspection when building a native executable with:

```
--enable-monitoring=<all,heapdump,jfr,jvmstat> . Defaults to all.
```

Enable JFR support and record events:

```
./myapp -XX:+FlightRecorder -XX:StartFlightRecording="filename=recording.jfr"
```

Dump the initial heap of a native executable:

```
./myapp -XX:+DumpHeapAndExit
```

**DEBUG**

Build a native executable with GDB or any IDE that integrates GDB:

```
-g -O0
```

Debug a native executable with GDB or any IDE that integrates GDB:

```
```

* Available with GraalVM Enterprise Native Image

More info at  **www.graalvm.org/native-image**