The JMC Agent

This page is for capturing the demands and discussing the syntax for the JMC agent. There is a very early prototype currently checked into the JMC repo, but it is currently not being built as part of the JMC product. For information on building and playing around with the prototype, please check core/org.openjdk.jmc.agent/README.md.

Design Goals

- Provide a minimal agent for instrumenting methods with flight recorder events
- Piggybacks on the flight recorder capabilities to provide low overhead event generation, cheap time stamping
- Allows subsequent enablement/disablement of generated events using normal JFR templates
  (In other words, to disable an instrumentation point, no subsequent redefinition is required.)
- Clear and precise syntax
  (No wildcards. Instrumentation points meant to be added with assistance from tooling, but syntax should be easy enough to be manually edited, if required.)
- Minimal overhead in terms of added code/generated classes
  (On parity with manually added events.)
- Loadable as an agent into an already running runtime
  (Can redefine classes.)
- Controllable through a JMX MBean
  (Perhaps simply an operation taking a new probe definition.)
- Must be extremely safe out of the box
  - No calling methods (avoiding halting problems) / unexpected exceptions (except possibly when converters enabled?)
  - No implicit type conversions (boxing of wrapper types to primitive types ok?)
  - General catch-all calls to toString(), if at all allowed, must be explicitly enabled
  - Module safe - only module redefinition is making jdk.jfr readable from instrumented modules
- Built for production use
  - Need to have sufficient unit testing
  - Need to be tested against larger bodies of code (Oracle can help with Fusion staging and test environments etc)
  - Need to support OracleJDK 8 as well as OpenJDK 11+

Stretch Goals

- Annotation driven generation
  (Opt-in, adds the overhead of looking for annotations in all classes, i.e. parsing all classes.)
- Get it into the JDK?
  (Could potentially remove some of the overhead of looking for the annotations)

Non-functional Demands

- Should emit the minimum byte code required for emitting the event
- Should generate the minimum amount of support classes and code required for emitting the event

Links

- Development Plan
- Probe definition format
- Implementation Notes