
ORACLE®

JavaOne™

ORACLE®

Unified JVM Logging in JDK 9

Marcus Larsson
Senior Member of Technical Staff
Oracle, HotSpot Runtime Team
September, 2016

Java Your Next (Cloud)

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.

Agenda

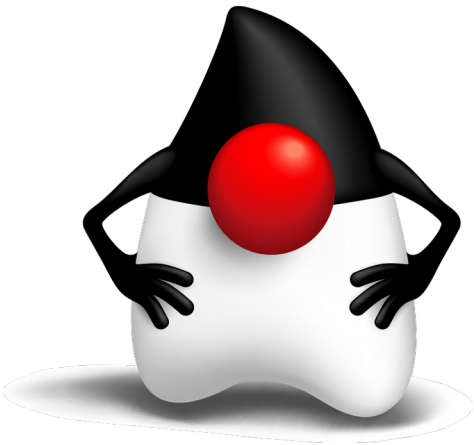
1. Logging History
2. Unified Logging Concepts
3. Additional Configuration
4. Usage and Recommendations

Agenda

1. Logging History
2. Unified Logging Concepts
3. Additional Configuration
4. Usage and Recommendations

Note

- This presentation is **not** about logging in Java
  - Unrelated to java.util.logging
- Only handles logging for the Java Virtual Machine (JVM)



History of Logging in the JVM

- Simple printouts to terminal guarded by flags
  - E.g. `-XX:+PrintGC -XX:+PrintGCDetails, -XX:+Print…, -XX:+Trace…`
- Features tied to certain types of logging
  - GC logging to file (`-Xloggc:…`)
    - Optional log file rotation (`-XX:+UseGCLogFileRotation`)
  - Also supports decorating messages with timestamps
    - E.g. `-XX:+PrintGCTimeStamps`, `-XX:+PrintGCDateStamps`

Problems With Previous Solution

- Difficult to configure
  - Requires numerous command line arguments
  - Hard to discover what logging is available
    - Requires knowledge about each logging flag
  - Not always re-configurable during runtime
- Only supports logging to file for a fixed set of messages
  - Single file logging
- No consistent use of log message decorations

Logging Flags

-XX:+TraceClassResolution

-XX:+TraceExceptions

-XX:+TraceClassUnloading

-XX:+PrintGCDetails

-XX:+PrintGC

-XX:+PrintVtables

-XX:+TraceMonitorInflation

-XX:+TraceClassLoaderData

-XX:+TraceVMOperation

-XX:+PrintGCCause

-XX:+TraceClassInitialization

-XX:+TraceMonitorMismatch

-XX:+TraceDynamicGCThreads

-XX:+TraceSafepoint

-XX:+TraceRedefineClasses

-XX:+VerboseVerification

-XX:+TraceStartupTime

-XX:+PrintCMSStatistics

-XX:+TraceDefaultMethods

-XX:+PrintTaskQueue

-XX:+TraceBiasedLocking

-XX:+PrintCompressedOopsMode

-XX:+PrintJNIGCStalls

-XX:+PrintTLAB

-XX:+TraceReferenceGC

-XX:+PrintPLAB

-XX:+TraceClassResolution

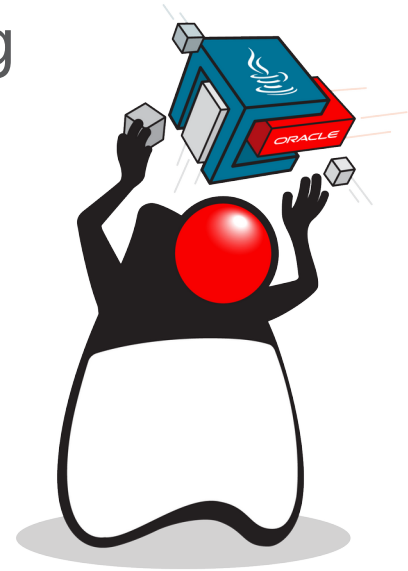
-XX:+TraceMonitorInflation

Sample Logging from JDK 8

```
$ java -XX:+TraceClassLoading -XX:+TraceMonitorInflation -XX:+PrintGCDetails ...

...

[GC (Allocation Failure) [PSYoungGen: 559296K->49792K(575488K)] 675307K->184715K(746496K),
0,0983604 secs] [Times: user=0,25 sys=0,03, real=0,10 secs]

[Full GC (Ergonomics) [PSYoungGen: 49792K->8772K(575488K)] [ParOldGen: 134923K-
>170112K(309248K)] 184715K->178884K(884736K), [Metaspace: 7744K->7744K(1056768K)],
1,6429547 secs] [Times: user=3,80 sys=0,01, real=1,64 secs]

[Loaded java.math.BigDecimal$StringBuilderHelper from /usr/lib/jvm/java-8-
jdk/jre/lib/rt.jar]

Deflating object 0x00000006cb4519c0 , mark 0x00007f6cb8004c1a , type org.h2.engine.Database

[GC (Allocation Failure) [PSYoungGen: 506948K->35616K(649728K)] 677060K->207952K(958976K),
0,1414756 secs] [Times: user=0,30 sys=0,01, real=0,14 secs]

...
```

The Unified JVM Logging Project

- Introduce a framework for all logging in the JVM
  - Use the concept of log levels
  - Categorize messages
  - Allow fine-grained control of what, where and how to log
  - Make it easier to use
- Adds the `-Xlog` java command line argument



Command Line Usage

- JVM configured for warnings and errors to *stdout*
- `-Xlog` enables info level logging
  - Customize with additional arguments (`-Xlog:<options>`)
- `-Xlog:help` for reference
  - Syntax, available options, examples
- `-Xlog:disable` to silence

Usage Example

```
$ java -Xlog -version
[0.003s][info][os] SafePoint Polling address: 0x00007fd18e125000
[0.003s][info][os] Memory Serialize Page address: 0x00007fd18e124000
[0.003s][info][os] HotSpot is running with glibc 2.24, NPTL 2.24
[0.003s][info][biasedlocking] Aligned thread 0x00007fd184012f70 to 0x00007fd184013000
[0.003s][info][os,thread    ] Thread attached (tid: 25659, pthread id: 140538008258304).
[0.003s][info][class,path   ] bootstrap loader class path=/home/mlarsson/jdk-9/lib/modules
[0.003s][info][class,path   ] classpath:
[0.003s][info][class,path   ] opened: /home/mlarsson/jdk-9/lib/modules
[0.003s][info][class,load   ] opened: /home/mlarsson/jdk-9/lib/modules
[0.003s][info][pagesize     ] CodeHeap 'non-nmethods':  min=2496K max=5696K
base=0x00007fd16bcb2000 page_size=4K size=5696K
[0.003s][info][pagesize     ] CodeHeap 'profiled nmethods':  min=2496K max=120032K
base=0x00007fd16c242000 page_size=4K size=120032K
[0.003s][info][pagesize     ] CodeHeap 'non-profiled nmethods':  min=2496K max=120032K
base=0x00007fd17377a000 page_size=4K size=120032K
...
```

Agenda

1. Logging History
2. Unified Logging Concepts
3. Additional Configuration
4. Usage and Recommendations

Logging

- Free text messages
- Line oriented
  - Typically one message – one line
- Not a low overhead profiling and instrumentation tool
  - Use Java Flight Recorder instead

Log Message Classification

- Classify messages according to content
  - Not all messages fit in a single category
    - Some messages span multiple categories
- Use a message tagging scheme
  - Each message is associated with a set of tags (its **tag set**)
    - A tag set is one or more tags identifying a category of messages
    - Maximum of 5 tags in a tag set
  - Selecting what to log is a matter of listing tags of interest

Log Tags

- A tag might describe a particular JVM subsystem
  - GC, interpreter, JIT compiler, OS interaction, etc.
- Or an action
  - Update, load, cleanup
- Can also distinguish messages of interest
  - For example, the **start** tag signifying start of some event

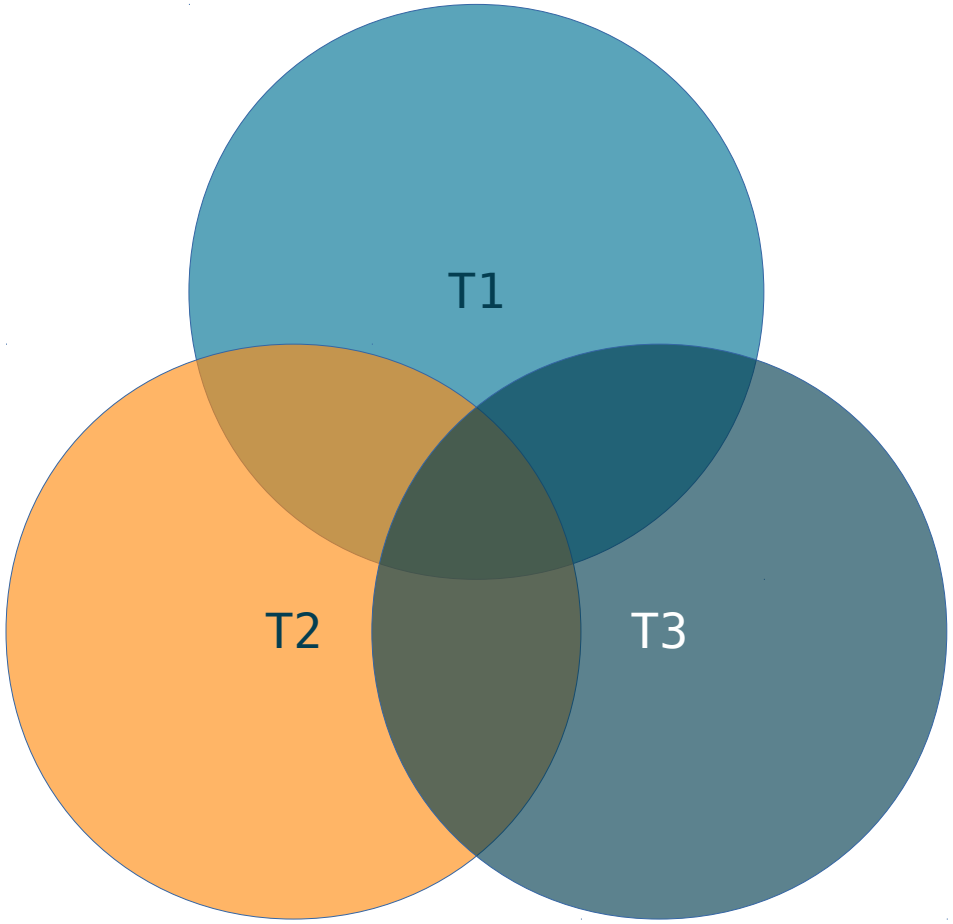
Selecting Log Tags

- First parameter to `-Xlog` describes what to log
  - E.g. -Xlog:gc enables all ‘gc’ tagged logging
  - Multiple selections comma separated: `-Xlog:gc,os`
  - Combine tags to form tag sets using ‘+’
    - E.g. -Xlog:gc+heap
  - Add ‘*’ to select **at least** the given tags
    - E.g. `-Xlog:gc*` selects all tag sets containing the ‘gc’ tag
- Syntax reference: `-Xlog:[<selection>]`
  - Where `<selection> = t1[+t2…][*][,…]`
    `‘all’`

Usage Example With Selection

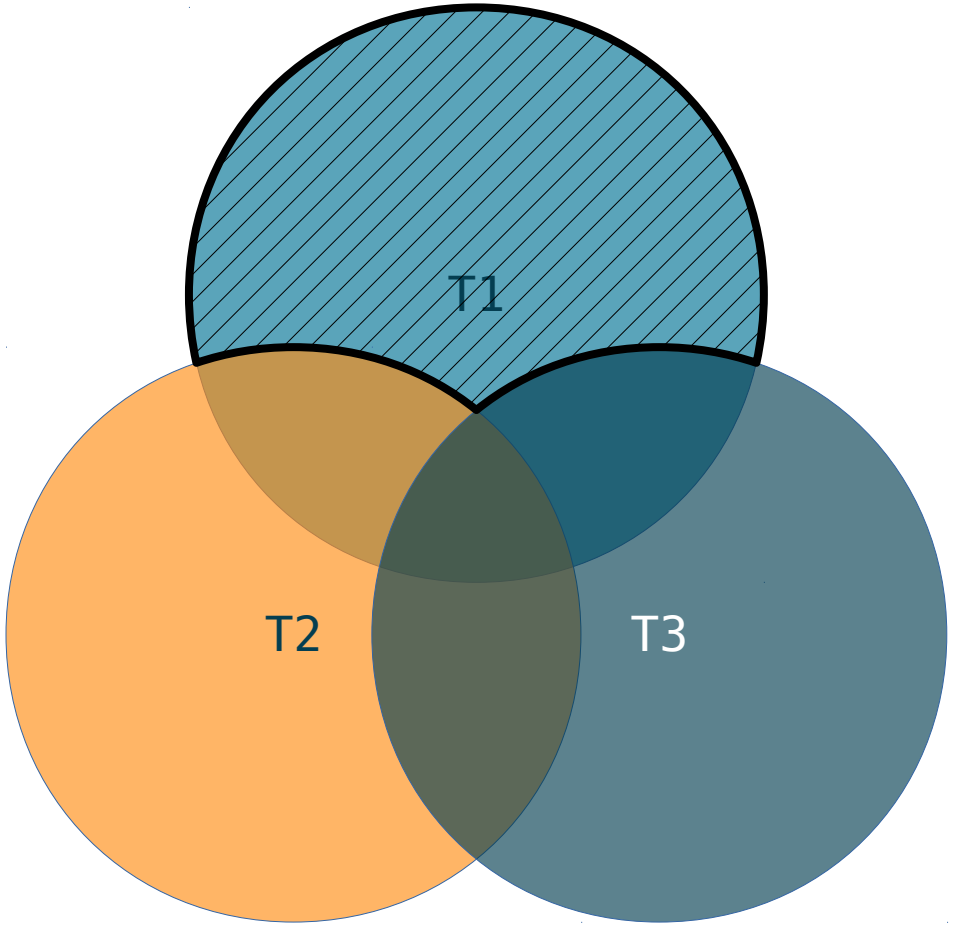
```
$ java -Xlog:gc* -version
[0.007s][info][gc,heap] Heap region size: 1M
[0.010s][info][gc     ] Using G1
[0.010s][info][gc,heap,coops] Heap address: 0x00000006c6c00000, size: 3988 MB, Compressed
Oops mode: Zero based, Oop shift amount: 3
java version "9-ea"
Java(TM) SE Runtime Environment (build 9-ea+135)
Java HotSpot(TM) 64-Bit Server VM (build 9-ea+135, mixed mode)
[0,100s][info][gc,heap,exit ] Heap
[0,100s][info][gc,heap,exit ]  garbage-first heap   total 256000K, used 2048K
[0x00000006c6c00000, 0x00000006c6d007d0, 0x00000007c0000000)
[0,100s][info][gc,heap,exit ]   region size 1024K, 3 young (3072K), 0 survivors (0K)
[0,100s][info][gc,heap,exit ]  Metaspace       used 3532K, capacity 4486K, committed 4864K,
reserved 1056768K
[0,100s][info][gc,heap,exit ]   class space    used 337K, capacity 386K, committed 512K,
reserved 1048576K
```

Log Tag Selection Visualized



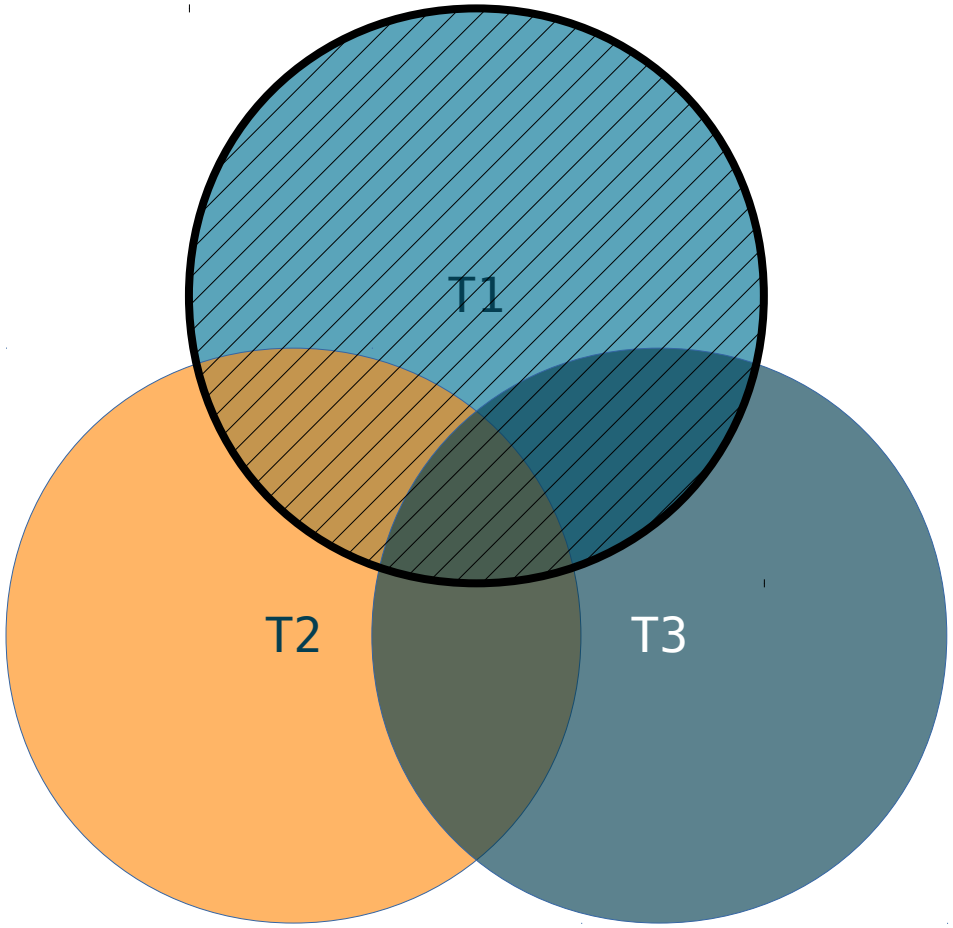
- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - T1 `-Xlog:T1*`
  - Only T1 and T2 `-Xlog:T1+T2`
  - T1 and T2 `-Xlog:T1+T2*`
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - All `-Xlog:all`

Log Tag Selection Visualized



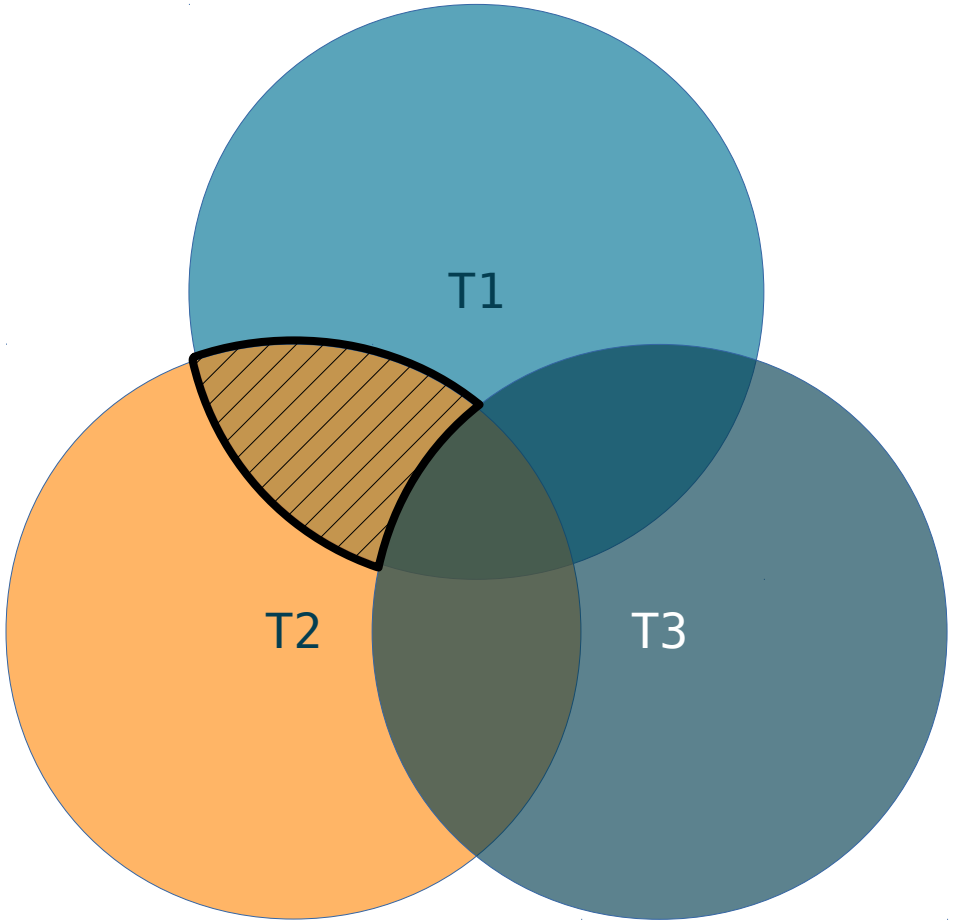
- Log tags T1, T2, T3
- Example selections:
  - **Only T1** **`-Xlog:T1`**
  - T1 `-Xlog:T1*`
  - Only T1 and T2 `-Xlog:T1+T2`
  - T1 and T2 `-Xlog:T1+T2*`
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - All `-Xlog:all`

Log Tag Selection Visualized



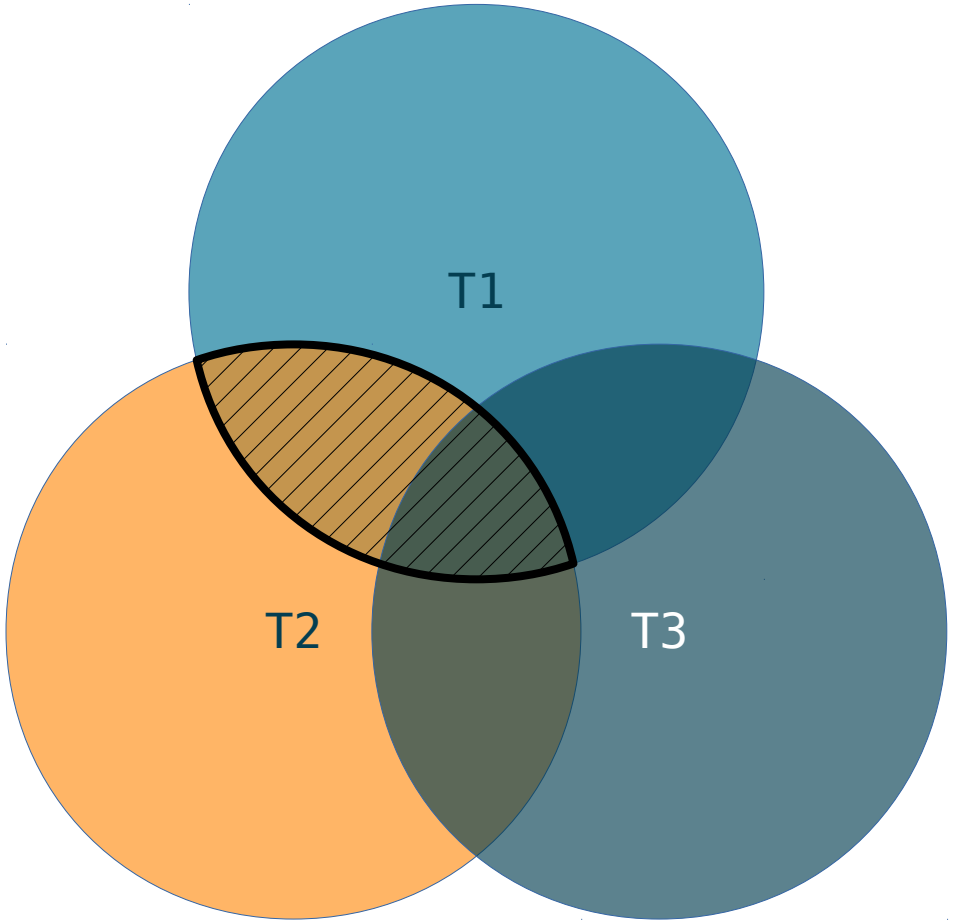
- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - **T1** **`-Xlog:T1*`**
  - Only T1 and T2 `-Xlog:T1+T2`
  - T1 and T2 `-Xlog:T1+T2*`
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - All `-Xlog:all`

Log Tag Selection Visualized



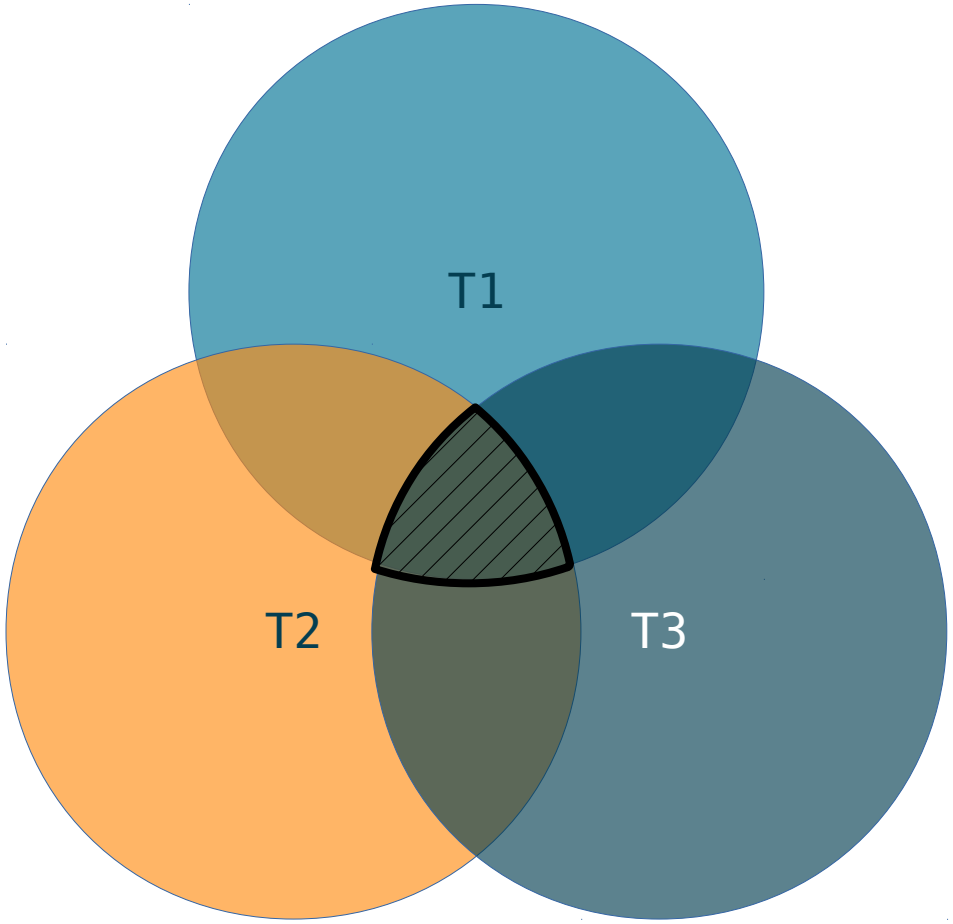
- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - T1 `-Xlog:T1*`
  - **Only T1 and T2** **`-Xlog:T1+T2`**
  - T1 and T2 `-Xlog:T1+T2*`
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - All `-Xlog:all`

Log Tag Selection Visualized



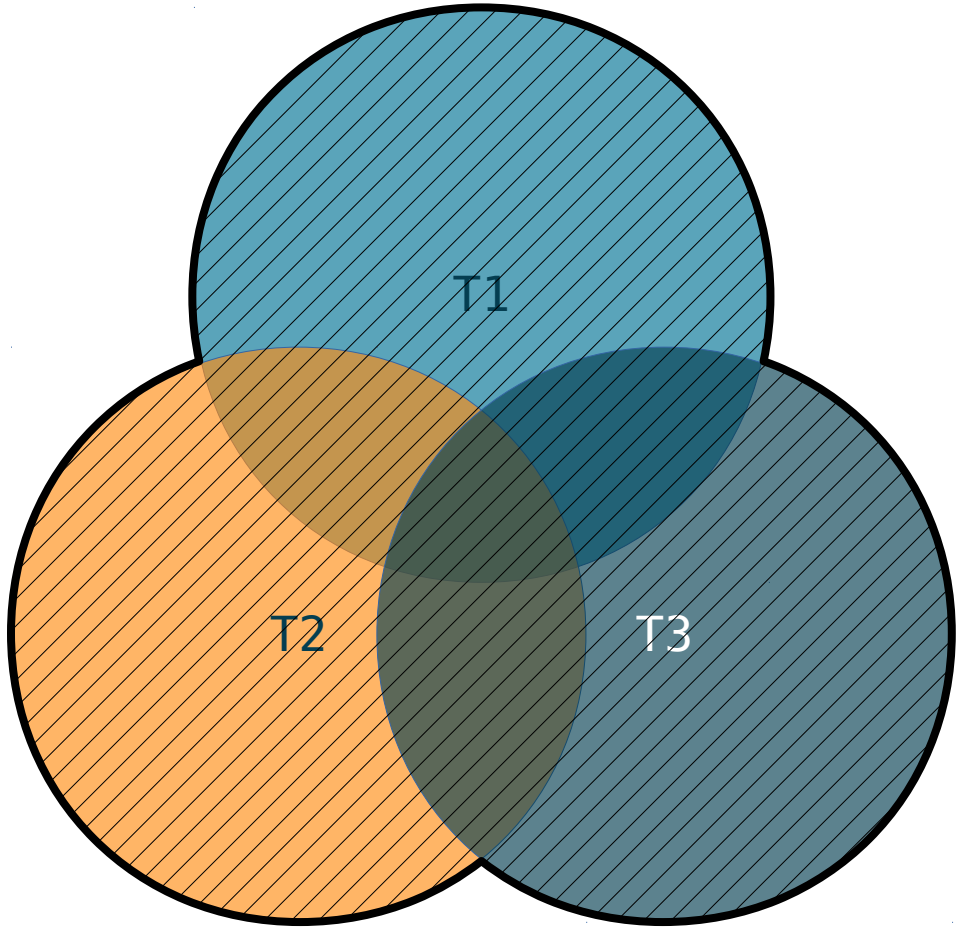
- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - T1 `-Xlog:T1*`
  - Only T1 and T2 `-Xlog:T1+T2`
  - **T1 and T2** **`-Xlog:T1+T2*`**
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - All `-Xlog:all`

Log Tag Selection Visualized



- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - T1 `-Xlog:T1*`
  - Only T1 and T2 `-Xlog:T1+T2`
  - T1 and T2 `-Xlog:T1+T2*`
  - **Only T1, T2, T3** **`-Xlog:T1+T2+T3`**
  - All `-Xlog:all`

Log Tag Selection Visualized



- Log tags T1, T2, T3
- Example selections:
  - Only T1 `-Xlog:T1`
  - T1 `-Xlog:T1*`
  - Only T1 and T2 `-Xlog:T1+T2`
  - T1 and T2 `-Xlog:T1+T2*`
  - Only T1, T2, T3 `-Xlog:T1+T2+T3`
  - **All** **`-Xlog:all`**

Log Levels

- Common set of levels
  - Separate messages according to verbosity and importance
  - In increasing verbosity: `error, warning, info, debug, trace`
  - Aggregated
    - `info` implies `warning` and `error`
- Log level selection can be done per tag selection
  - Enables fine grained control over what to log
    - E.g. log GC on debug level but class loading on info level

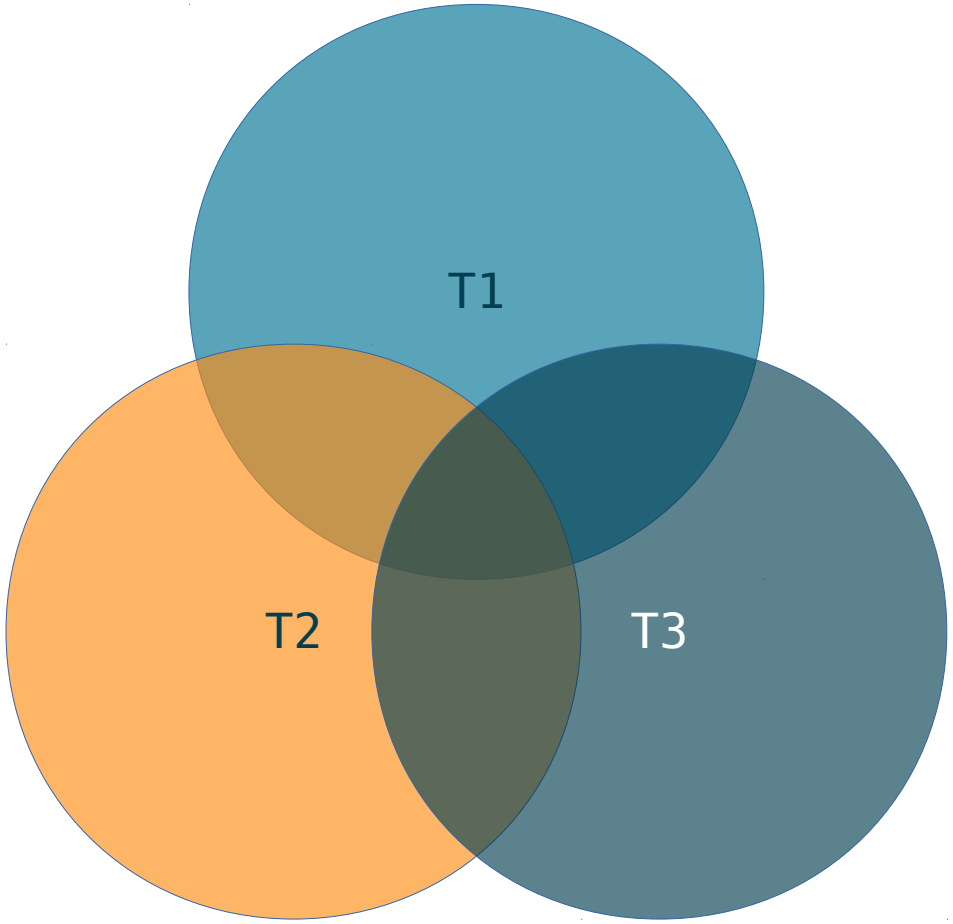
Selecting Log Level

- Level is a part of the selection syntax
  - Each tag selection may have a '`=<level>`' suffix
    - E.g. `-Xlog:gc=debug` or `-Xlog:gc*=debug,os=info`
    - Assume `info` level unless specified
- Can use '`off`' to selectively disable logging
  - E.g. `-Xlog:gc*,gc+heap=off`
- Amended -Xlog selection syntax:
  - `<selection> = t1[+t2…][*][=<level>][,…]`

Usage Example With Selection Including Level

```
$ java -Xlog:gc+heap*=debug -version
[0.007s][info][gc,heap] Heap region size: 1M
[0.007s][debug][gc,heap] Minimum heap 8388608  Initial heap 262144000  Maximum heap
4181721088
[0.008s][debug][gc,ergo,heap] Expand the heap. requested expansion amount:262144000B
expansion amount:262144000B
[0.009s][info ][gc,heap,coops] Heap address: 0x00000006c6c00000, size: 3988 MB, Compressed
Oops mode: Zero based, Oop shift amount: 3
java version "9-ea"
Java(TM) SE Runtime Environment (build 9-ea+135)
Java HotSpot(TM) 64-Bit Server VM (build 9-ea+135, mixed mode)
[0,093s][info ][gc,heap,exit ] Heap
[0,093s][info ][gc,heap,exit ]  garbage-first heap   total 256000K, used 1024K
[0x00000006c6c00000, 0x00000006c6d007d0, 0x00000007c0000000)
[0,093s][info ][gc,heap,exit ]   region size 1024K, 2 young (2048K), 0 survivors (0K)
[0,093s][info ][gc,heap,exit ]  Metaspace       used 3530K, capacity 4486K, committed
4864K, reserved 1056768K
[0,093s][info ][gc,heap,exit ]   class space    used 337K, capacity 386K, committed 512K,
reserved 1048576K
```

Log Tag Selection Revisited



- Can deselect just as well
- Examples:
  - T1 except with T3

    ```
    -Xlog:T1*,T3*=off
    ```

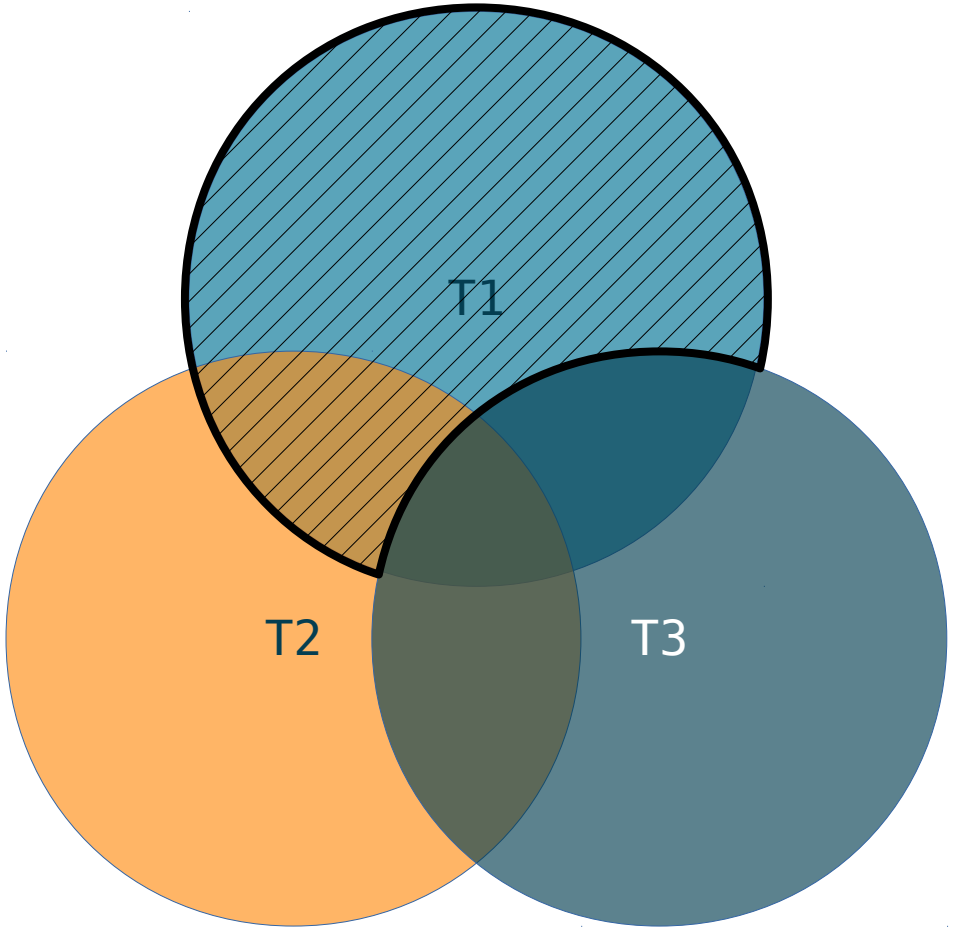
  - T1 except with T1+T3

    ```
    -Xlog:T1*,T1+T3=off
    ```

- Use `warning` to preserve default settings

  ```
  -Xlog:T1*,T3*=warning
  ```

Log Tag Selection Revisited



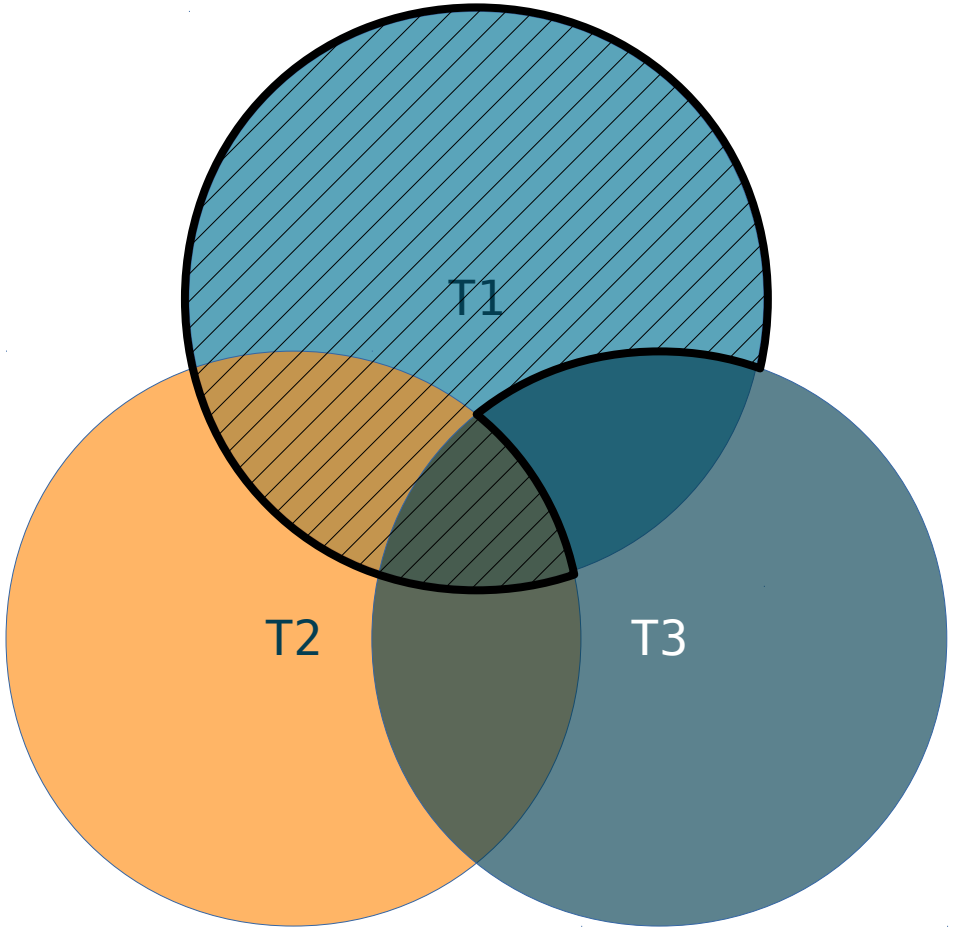
- Can deselect just as well
- Examples:
  - **T1 except with T3**

    `-Xlog:T1*,T3*=off`
  - T1 except with T1+T3

    `-Xlog:T1*,T1+T3=off`
- Use `warning` to preserve default settings

  `-Xlog:T1*,T3*=warning`

Log Tag Selection Revisited



- Can deselect just as well
- Examples:
  - T1 except with T3

    `-Xlog:T1*,T3*=off`
  - **T1 except with T1+T3**

    `-Xlog:T1*,T1+T3=off`
- Use `warning` to preserve default settings

  `-Xlog:T1*,T3*=warning`

Tracking Log Tag Selection

- Use `-Xlog:logging=trace`
  - Lists all available tag sets
  - Logs initial configuration
    - All configured outputs
    - Explicitly lists all tag sets with configured levels
      - Can view and verify the configuration
- Unmatched selections cause warnings

```
$ java -Xlog:logging+gc
[0.003s][warning][logging] No tag set matches selection(s): logging+gc
```

Agenda

1. Logging History
2. Unified Logging Concepts
3. Additional Configuration
4. Usage and Recommendations

Log Outputs

- Allow logging both to terminal and file(s)
  - Valid outputs are *stdout*, *stderr*, and any number of log files
    - Support for log file rotation
      - Based on target file size
- Log configuration is tied to the output
  - Can have different logging selections for different files
    - E.g. log GC messages on trace level to `gctrace.log`, while at the same time logging info level messages to the terminal

Selecting Log Output

- Second `-Xlog` option specifies output
  - E.g. `-Xlog:all=debug:debug.log` logs to file ‘debug.log’
  - Valid options are stdout, stderr, file=<filename>
    - ‘`file=`’ prefix is default
  - Omitting the output defaults to `stdout`
    - I.e. `-Xlog:all` is short for `-Xlog:all:stdout`
- New syntax: `-Xlog:[<selection>]:[<output>]`
  - Where `<output> =` ‘`stdout`’
‘`stderr`’
`[file=]<filename>`

Selecting Log Output

- Can include JVM PID and timestamp in filename
  - %t expands to timestamp for JVM start
    - E.g. ‘foo.log.%t’ → ‘foo.log.2016-09-21_09-05-17’
  - %p expands to the JVM process ID
    - E.g. ‘`foo.%p.log`’ → ‘`foo.4711.log`’

Usage Example With a Different Log Output

```
$ java -Xlog:gc*=debug:gc-debug.log -version
java version "9-ea"
Java(TM) SE Runtime Environment (build 9-ea+135)
Java HotSpot(TM) 64-Bit Server VM (build 9-ea+135, mixed mode)
$ cat gc-debug.log
[0.010s][info][gc,heap] Heap region size: 1M
[0.010s][debug][gc,heap] Minimum heap 8388608  Initial heap 262144000  Maximum heap
4181721088
[0.010s][debug][gc,ergo,refine] Initial Refinement Zones: green: 4, yellow: 12, red: 20,
min yellow size: 8
[0.011s][debug][gc,marking,start] Initialize Card Live Data (0.011s)
[0.011s][debug][gc,marking      ] Initialize Card Live Data (0.011s, 0.011s) 0.019ms
[0.011s][debug][gc              ] ConcGCThreads: 1
[0.011s][debug][gc              ] ParallelGCThreads: 4
[0.011s][debug][gc,ergo,heap    ] Expand the heap. requested expansion amount:262144000B
expansion amount:262144000B
[0.013s][debug][gc,ihop         ] Target occupancy update: old: 0B, new: 262144000B
[0.013s][info ][gc              ] Using G1
...
```

Log Message Decorations

Log Message Decorations

- Decorate log messages with additional information
  - Uptime, level and tag set by default
    - E.g. `[0.014s][info][gc] <message>`
  - Supports a set of predetermined decorations:
    - Date and time in ISO 8601
    - JVM uptime
    - Log tags
    - Log level
    - Epoch seconds
    - System.currentMillis()
    - System.nanoTime()
    - Thread ID
    - Process ID
    - Hostname
- Configured per log output

Selecting Log Decorations

- Third `-Xlog` option specifies which decorators to use
  - Comma separated list of desired decorators
    - E.g. `-Xlog:all=debug:debug.log:uptime,tags,level,tid`
  - See -Xlog:help for the list of decorators
  - Can be set to ‘none’ for no decoration
- New syntax: `-Xlog:[<selection>]:[<output>]:[<decorators>]`
  - Where `<decorators> = <decorator>[,…]`
    `‘none’`

Log File Rotation

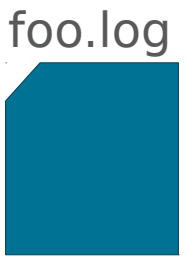
Log File Rotation

- Rotation based on target size with set number of files
- By default, files rotated using 5 files at 20MB each
- Target size and number of files configurable
  - Setting target size to 0 disables automatic rotation
    - Can still trigger rotations manually
  - Setting number of files to 0 disables rotation
    - Existing log files overwritten

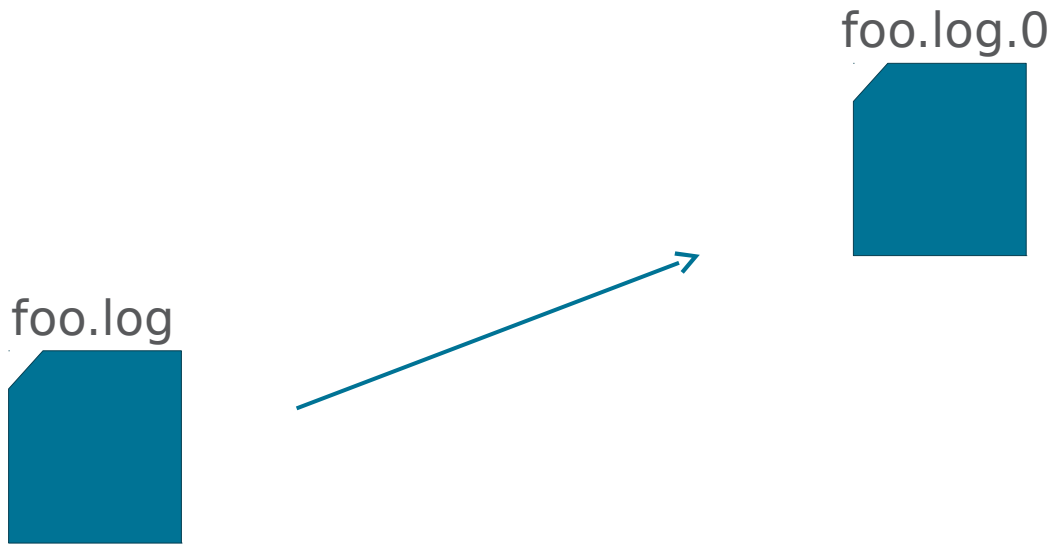
Rotation Mechanics

- Active log file uses given filename
  - E.g. `-Xlog:all:foo.log` logs to ‘foo.log’
- Rotated files are suffixed with archive number
  - E.g. ‘foo.log’ → ‘foo.log.0’
  - Round robin number assignment
    - First rotation adds `.0`, next rotation `.1`, etc
- Replaces oldest file when `filecount` reached
    - E.g. with `filecount=5`, sixth rotation will replace ‘foo.log.0’

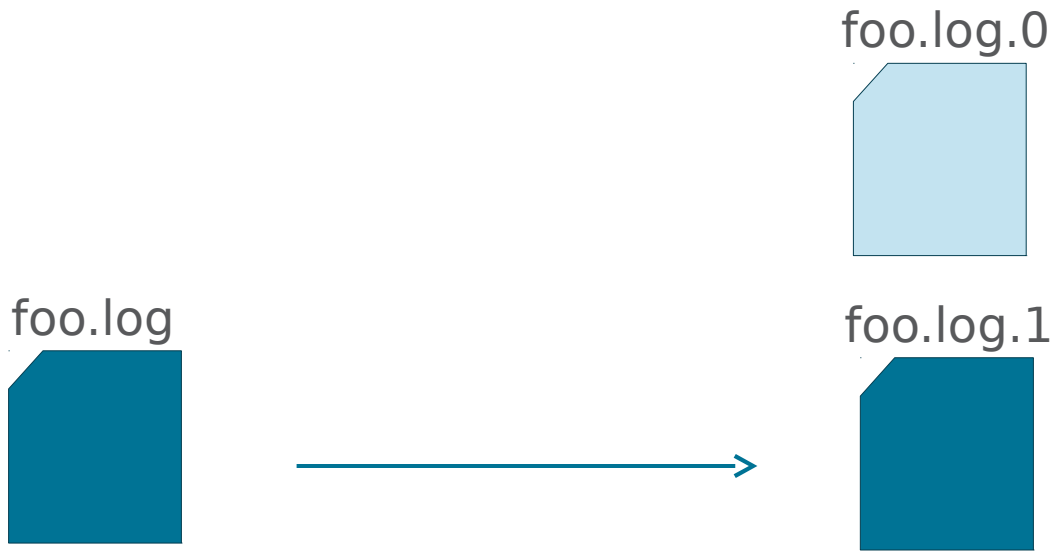
Rotation Mechanics Example



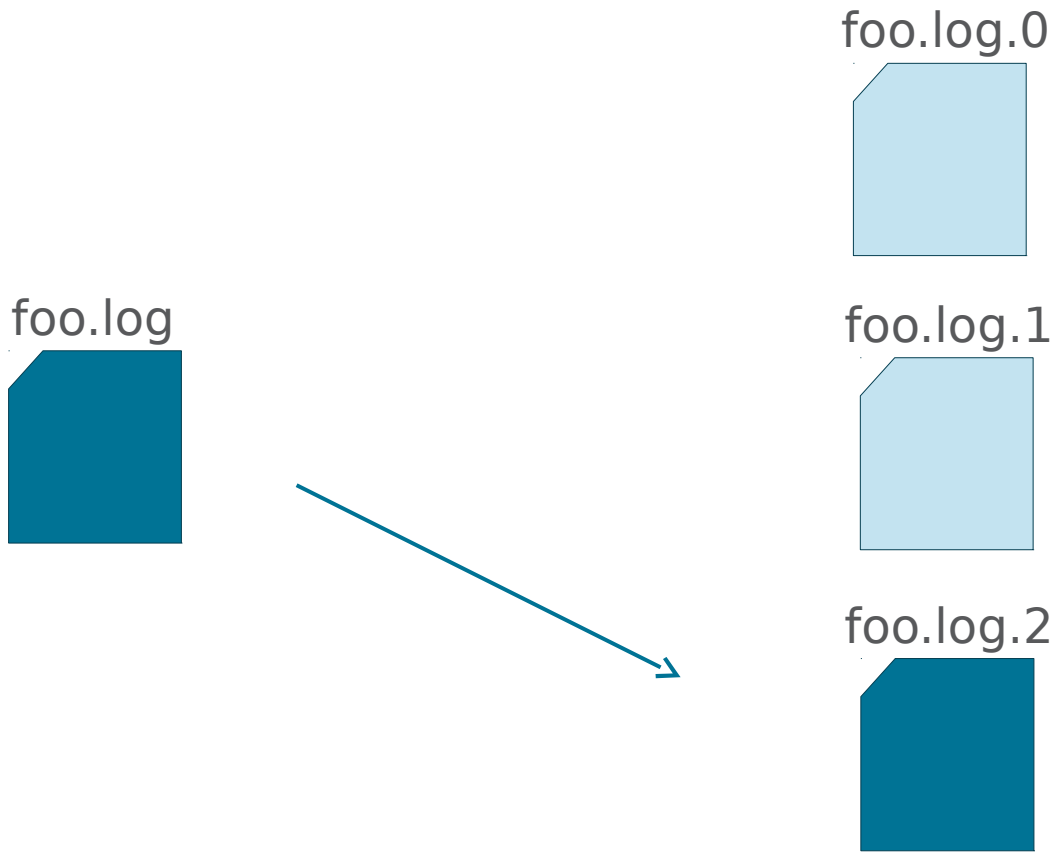
Rotation Mechanics Example



Rotation Mechanics Example

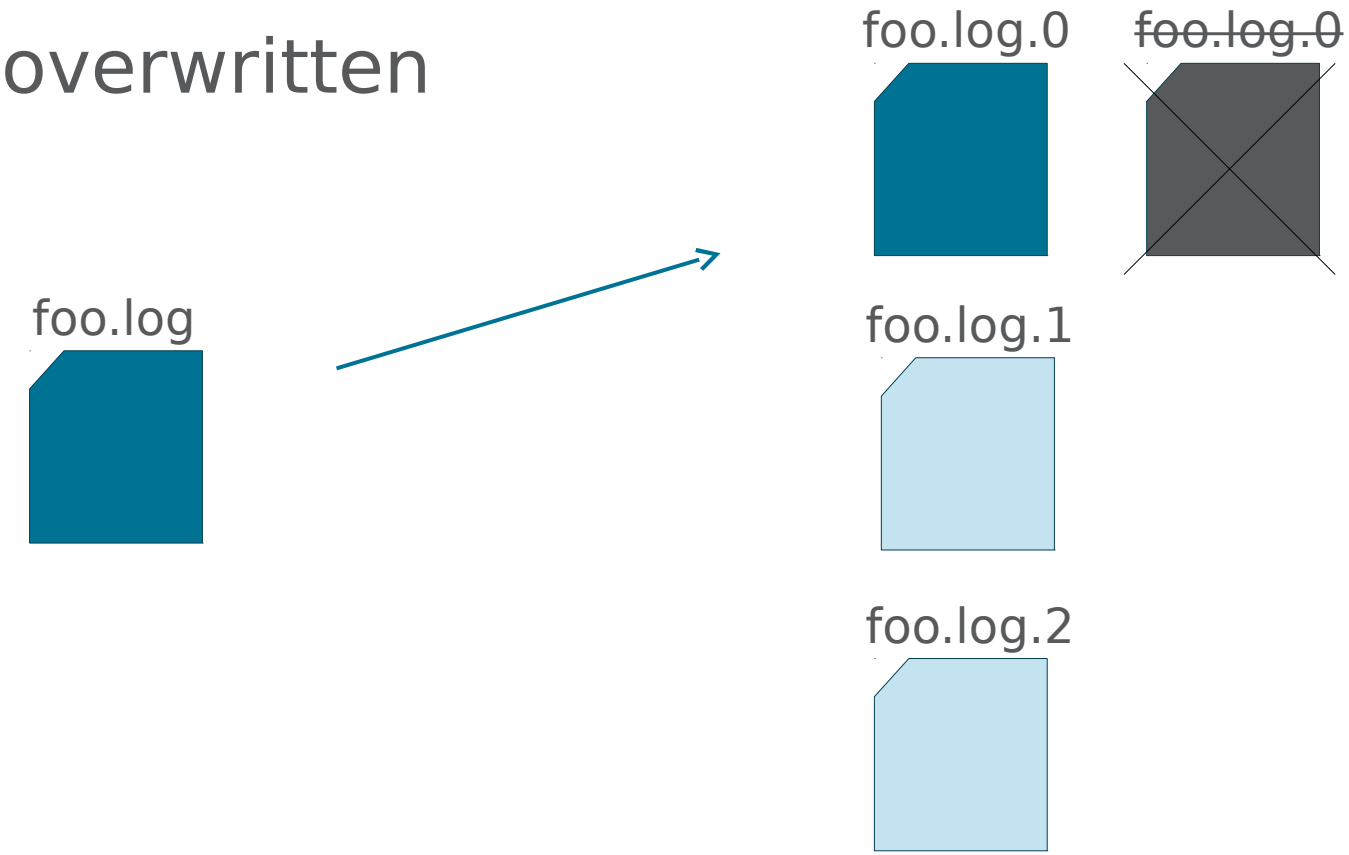


Rotation Mechanics Example



Rotation Mechanics Example

- Previous ‘foo.log.0’ overwritten



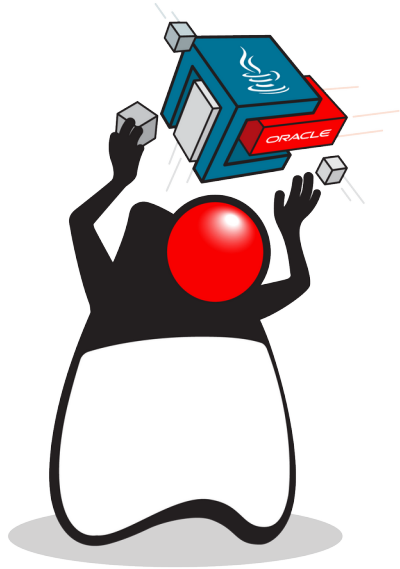
Configuring File Rotation

- Last option for `-Xlog` specifies output options
- Comma separated key=value pairs
- File outputs have `filecount` and `filesize`
  - Filesize specified in bytes (supports K, M, G suffix)
  - E.g. `-Xlog:gc:gc.log::filecount=10,filesize=200M`

Dynamic Reconfiguration

Reconfiguration During Runtime

- Log configuration can be changed at runtime
  - No need to restart the VM
  - Implemented using the diagnostic command **VM.log**
    - CLI using the **jcmd** utility
    - Exposed as MBean
      - Allows reconfiguration through JMX



VM.log Diagnostic Command

- Works similar to `-Xlog`
  - Named options
    - E.g ‘`jcmd <pid> VM.log what=gc output=stdout decorators=uptime`’
      - Equivalent to `-Xlog:gc:stdout:uptime`
- Supports listing current configuration
  - `VM.log list`
- Can trigger file rotations manually
  - `VM.log rotate`

VM.log Usage Example

```
$ jcmd <pid> VM.log list
Available log levels: off, trace, debug, info, warning, error
Available log decorators: time (t), uptime (u), timemillis (tm), uptimemillis (um), ...
Available log tags: add, age, alloc, annotation, ...
Described tag combinations:
 logging: Logging for the log framework itself
 ...
Log output configuration:
#0: stdout all=warning    uptime,level,tags
#1: stderr all=off        uptime,level,tags

$ jcmd <pid> VM.log output=#0 what=all=off list
...
Log output configuration:
#0: stdout all=off    uptime,level,tags
#1: stderr all=off    uptime,level,tags
```

Agenda

1. Logging History
2. Unified Logging Concepts
3. Additional Configuration
4. Usage and Recommendations

Unified Logging Adoption

- Not all logging converted
  - Ongoing effort
- Some flags removed
  - Others aliased and deprecated
- GC logging converted in full
  - The two most significant flags aliased
    - `-XX:+PrintGC` → `-Xlog:gc`
    - `-XX:+PrintGCDetails` → `-Xlog:gc*`

Deprecated and Aliased Flags

```
$ java -XX:+PrintGC -XX:+TraceClassLoading ...
[0.000s][warning][arguments] -XX:+TraceClassLoading is deprecated. Will use
-Xlog:class+load=info instead.
[0.000s][warning][gc       ] -XX:+PrintGC is deprecated. Will use -Xlog:gc instead.
[0.003s][info   ][class,load] opened: /home/mlarsson/jdk-9/lib/modules
[0.010s][info   ][gc        ] Using G1
...
```

Aliased Flags Example

Old Flag	Tag Set	Level
TraceClassLoading	`class+load`	`info`
TraceClassUnloading	`class+unload`	`info`
TraceClassResolution	class+resolve	debug
TraceExceptions	redefine+class	`info`
TraceMonitorInflation	monitorinflation	debug
TraceJVMTIObjectTagging	jvmti+objecttagging*	debug

Default and Implicit Configuration

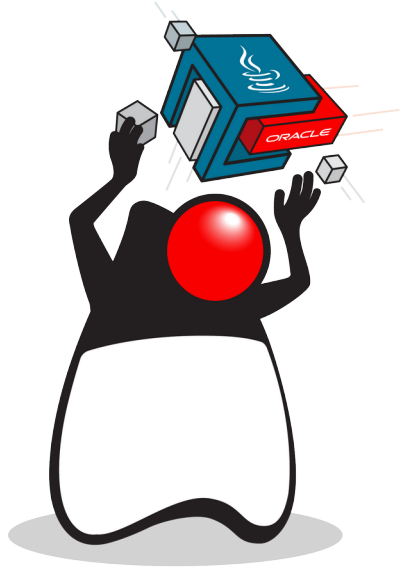
- Default configuration

  `-Xlog:all=warning:stdout:uptime,level,tags`

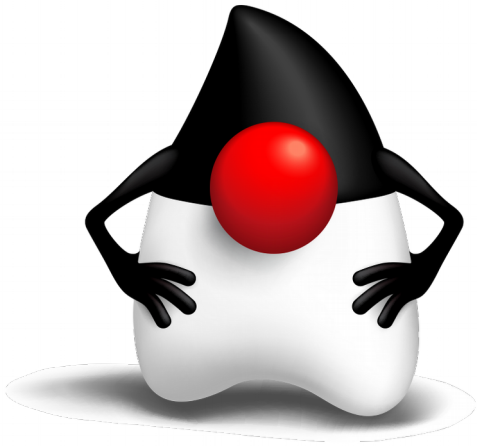
- Implicit configuration
  - When `-Xlog` is specified but options are omitted
    - `-Xlog:all=info:stdout:uptime,level,tags`
      - E.g. `-Xlog::stderr` → `-Xlog:all=info:stderr:uptime,level,tags`
    - File outputs add implicit `filecount=5,filesize=20M`

Recommendations

- Start out with broad selection
- Identify tags of interest
  - Narrow the selection
    - Optionally increase level of detail
- Use wildcard selections (*)
  - Makes configuration more robust



Questions?



JavaOne™

ORACLE®