Logging

Configuration

Basic configuration of the standard JDK to show logging methods can be accomplished with the following procedure:

In file %JAVA_HOME%/jre/lib/logging.properties or equivalent

- set the log handler is configured to show all levels

```
java.util.logging.ConsoleHandler.level = ALL
```

- set levels for the specific logging names that you wish to log

```
com.foo.level = FINE

com.foo.MyClass.level = FINEST
```

@Log

You can annotate your classes with the @Log transformation to automatically inject a logger in your Groovy classes, under the `log` property. Four kind of loggers are actually available:

- `@Log` for java.util.logging
- `@Commons` for Commons-Logging
- `@Log4j` for Log4J
- `@Slf4j` for SLF4J

Here's a sample usage of the @Log transformation:

```
import groovy.util.logging.*

@Log
class Car {
    Car() {
        log.info 'Car constructed'
    }
}

def c = new Car()
```

You can change the name of the logger by specifying a different name, for instance with `@Log('myLoggerName')`.

Another particularity of these logger AST transformations is that they take care of wrapping and safe-guarding logger calls with the usual `isSomeLevelEnabled()` calls. So when you write `log.info 'Car constructed'`, the generated code is actually equivalent to:

```
if (log.isLoggable(Level.INFO)) {
    log.info 'Car constructed'
}
```

Tracing Method Calls

In order to enable tracing of how Groovy calls MetaMethods, use the following settings:

in file %JAVA_HOME%/jre/lib/logging.properties or equivalent

- make sure your log handler is configured to show level 'FINER' at least,
  e.g.

```
java.util.logging.ConsoleHandler.level = ALL
```

- set MetaClass logging to 'FINER' at least,
  e.g.

```
groovy.lang.MetaClass.level = FINER
```

- set the appropriate Level for the Classes and optionally method names
  that you want to trace. The name for the appropriate logger starts with
  'methodCalls' and optionally ends with the method name,
  e.g.

```
# trace all method calls
methodCalls.level = FINER

# trace method calls to the 'String' class
methodCalls.java.lang.String.level = FINER

# trace method calls to Object.println()
methodCalls.java.lang.Object.println.level = FINER
```

Example:
with tracing enabled for all method calls a Groovy command line
script appears as follows (German locale)

```
$ groovy -e "println 'hi'"
13.09.2005 14:33:05 script_from_command_line run()
FEINER: called from MetaClass.invokeMethod
13.09.2005 14:33:05 script_from_command_line println('hi')
FEINER: called from MetaClass.invokeMethod
hi
```

See Also

- groovy.util.logging package