Arthas使用文档

**版本记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 修订日期 | 修订人员 | 修订纪要 |
| V1.0 | 2019-07-01 | 徐强辉 | 初稿 |

# 工具说明

Arthas 是基于 Greys 进行二次开发的全新在线诊断工具，利用Java6的Instrumentation特性，动态增强你所指定的类，获取你想要到的信息, 采用命令行交互模式，同时提供丰富的 Tab 自动补全功能，让你在定位、分析诊断问题时看每一个操作都看起来是那么的 666

# 工具作用.

* 这个类从哪个 jar 包加载的？为什么会报各种类相关的 Exception？
* 我改的代码为什么没有执行到？难道是我没 commit？分支搞错了？
* 遇到问题无法在预发 debug 一下，难道只能通过加日志再重新预发布吗？
* 线上遇到某个用户的数据处理有问题，但线上同样无法 debug，线下无法重现,怎么办?
* 是否有一个全局视角来查看系统的运行状况？
* 有什么办法可以监控到容器和中间件的实时运行状态？

# 工具使用

## 3.1 安装方式

### 3.1.1直接下载

wget https://alibaba.github.io/arthas/arthas-boot.jar

### 3.1.2 Cloud Toolkit插件

<https://alibaba.github.io/arthas/install-detail.html#cloud-toolkitarthas>

### 3.1.3 Docker

<https://alibaba.github.io/arthas/docker.html>

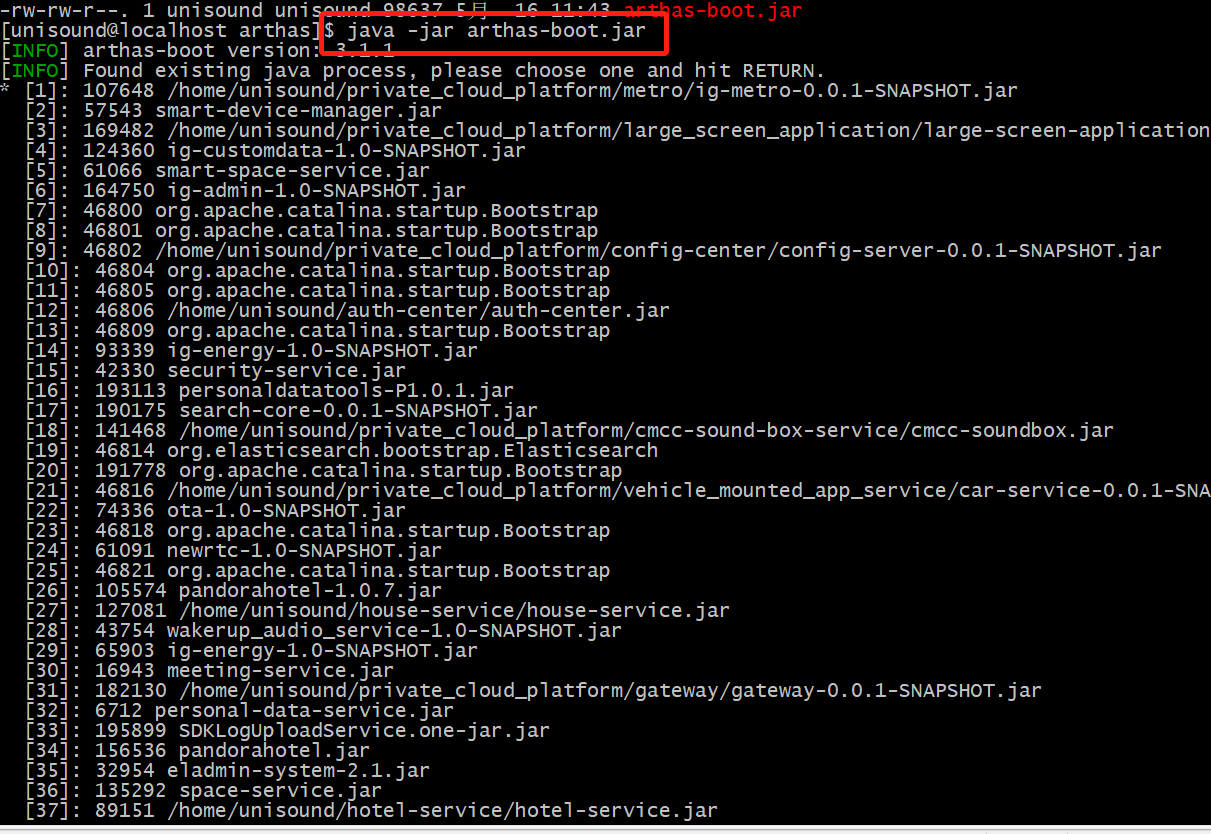
## 3.2 快速入门

### 3.2.1 启动

直接jar包启动.

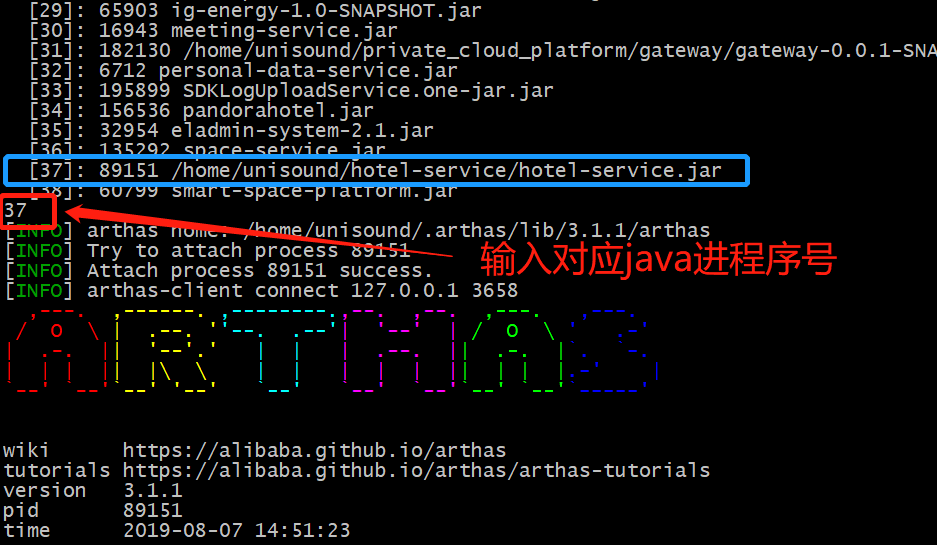
java -jar arthas-boot.jar

启动之后会 列出当前服务器所有jvm实例.



### 3.2.2 选择java进程

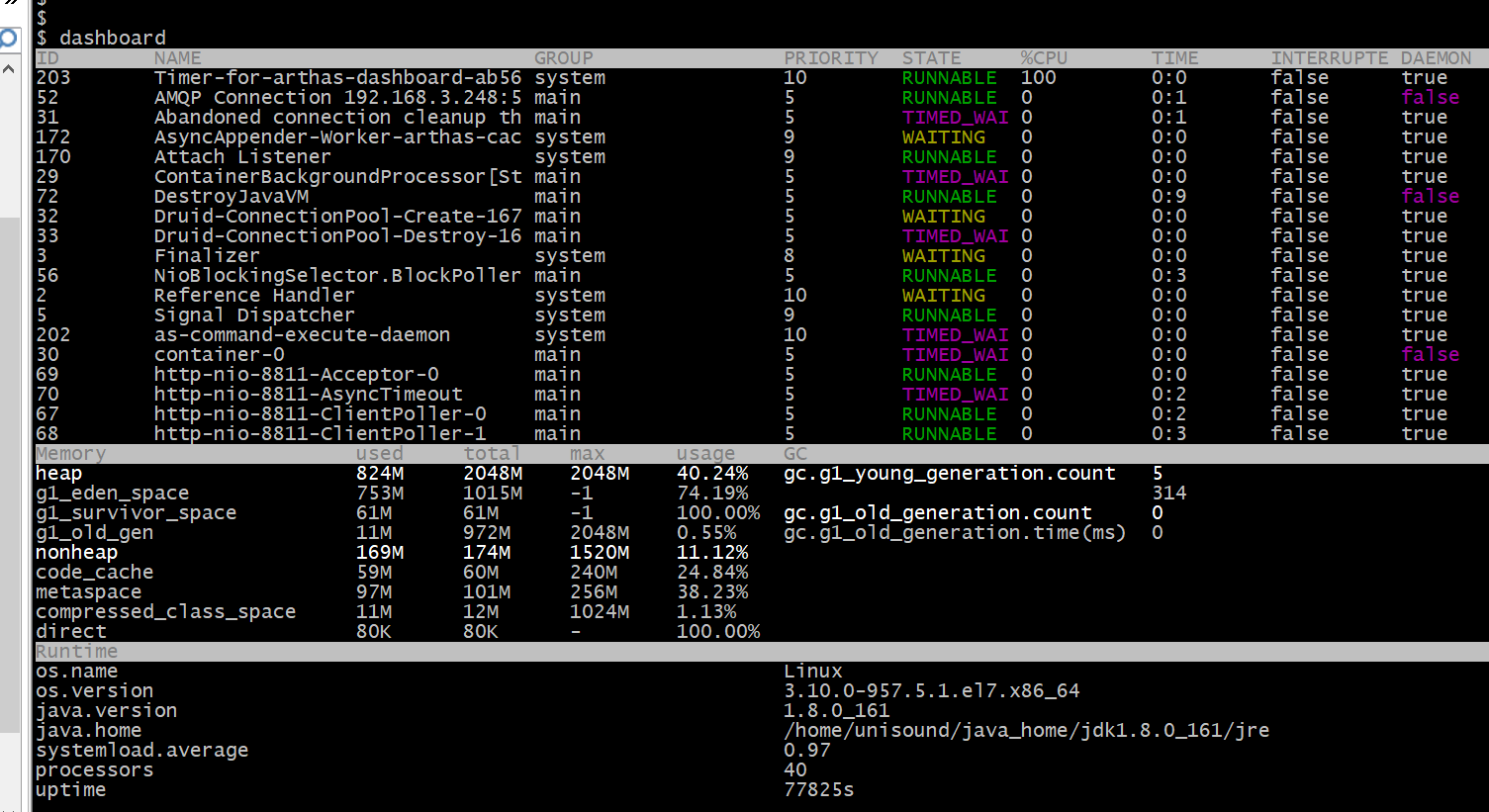
hotel-service 进程是 第37个, 输入37 按回车, Arthas 会attach到目标进程上.



### 3.2.3 基础命令

#### dashboard

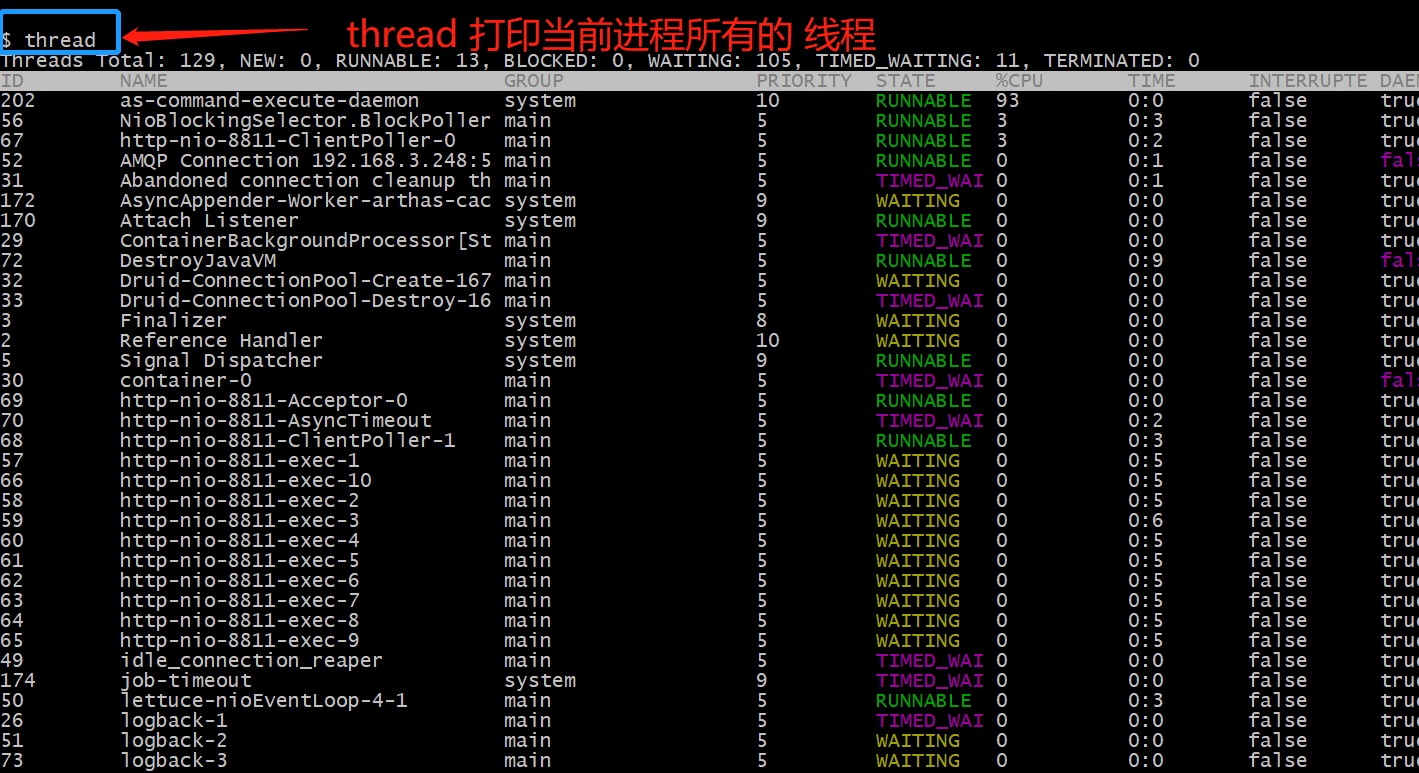
展示当前进程的信息.



#### thread

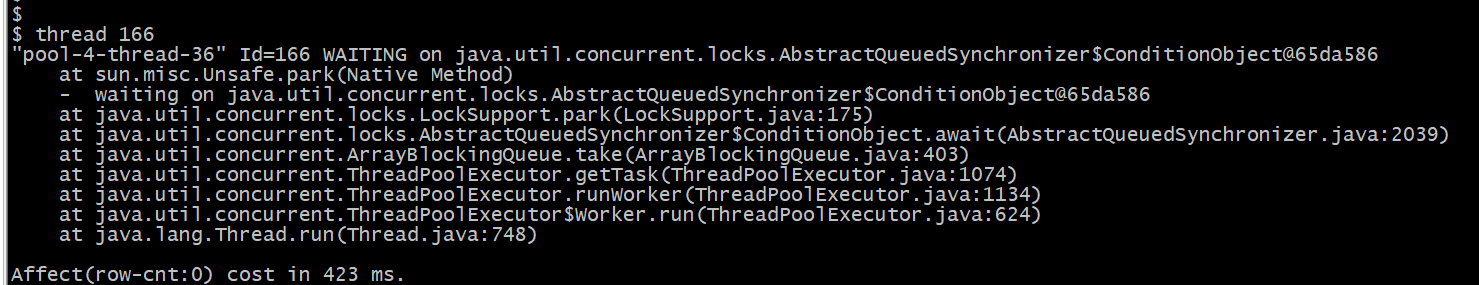
线程相关信息

thread 命令 展示所有线程信息.



thread + 线程序号

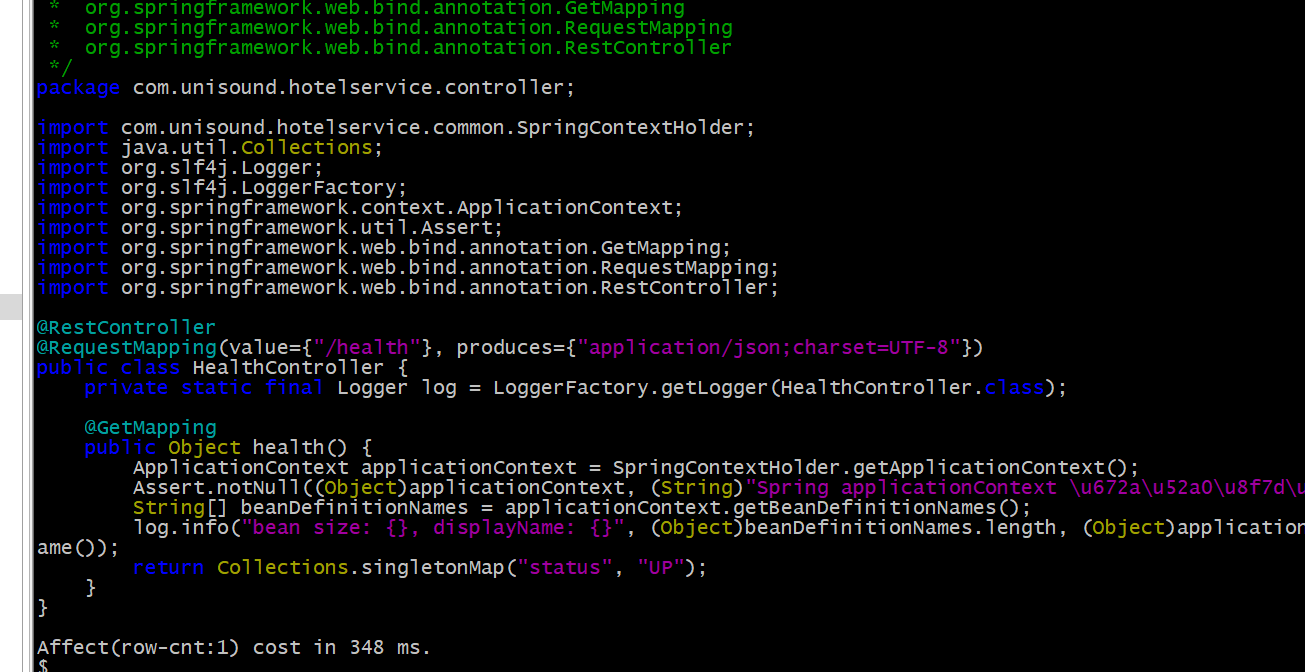
输出该线程的栈.



#### jad

反编译指令

jad com.unisound.hotelservice.controller.HealthController

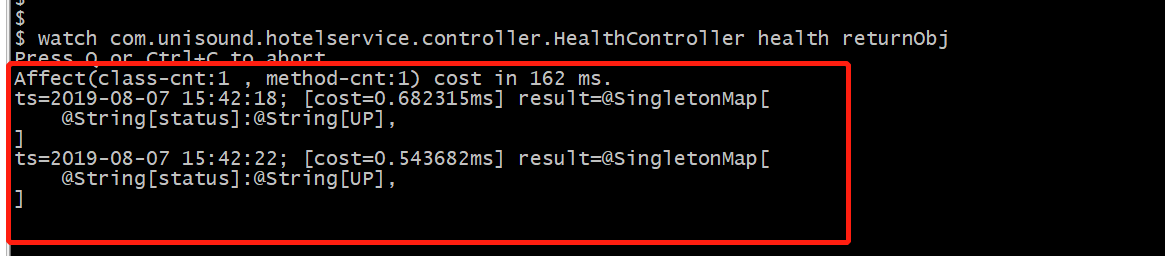


#### watch

监控输出某个类中某方法的 返回值.

watch com.unisound.hotelservice.controller.HealthController health returnObj

监控 HealthController.health() 的返回值



### 3.2.4 退出arthas

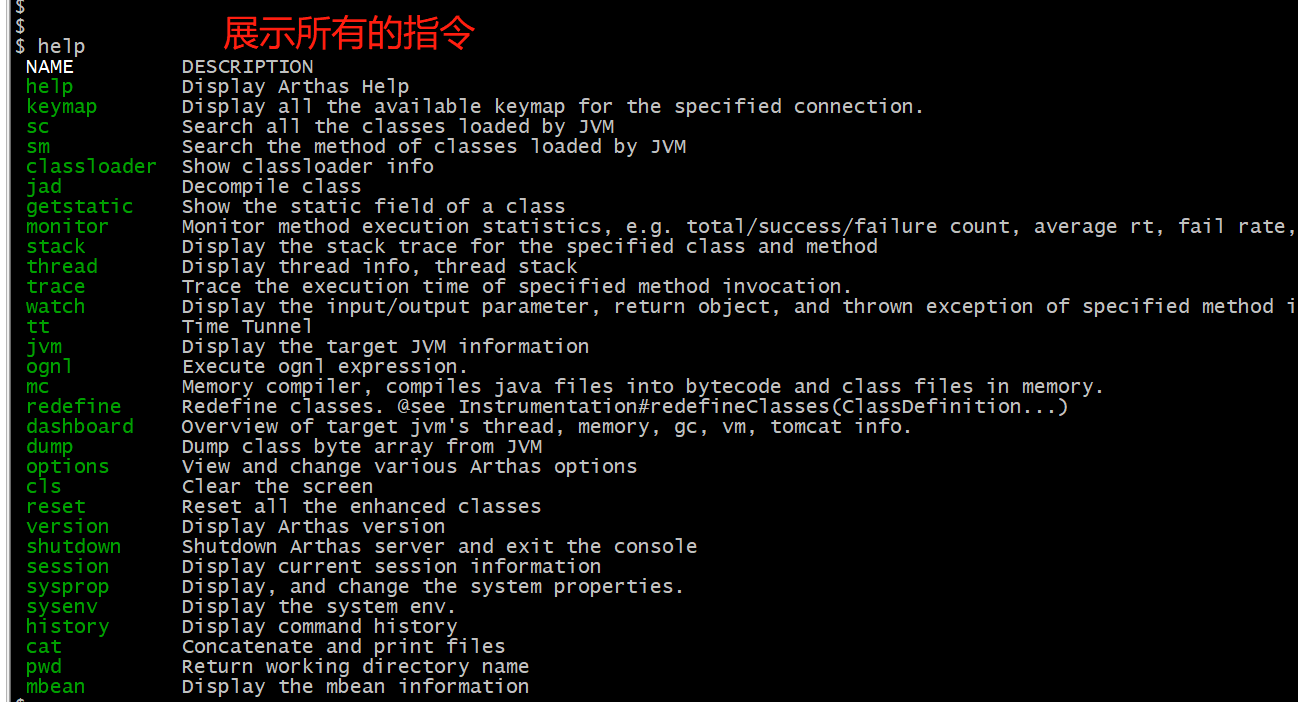
如果只是退出当前的连接，可以用quit或者exit命令。Attach到目标进程上的arthas还会继续运行，端口会保持开放，下次连接时可以直接连接上。

如果想完全退出arthas，可以执行shutdown命令。

# 进阶使用

## 基础命令

### help



### cat

打印文件内容，和linux里的cat命令类似

cat /tmp/a.txt

### pwd

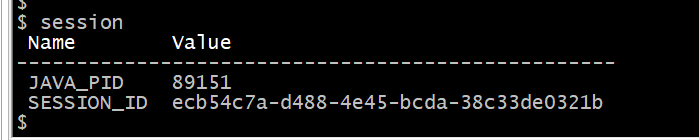
返回当前的工作目录，和linux命令类似

### cls

清空当前屏幕区域, 跟linux ctrl+l 命令类似

### session

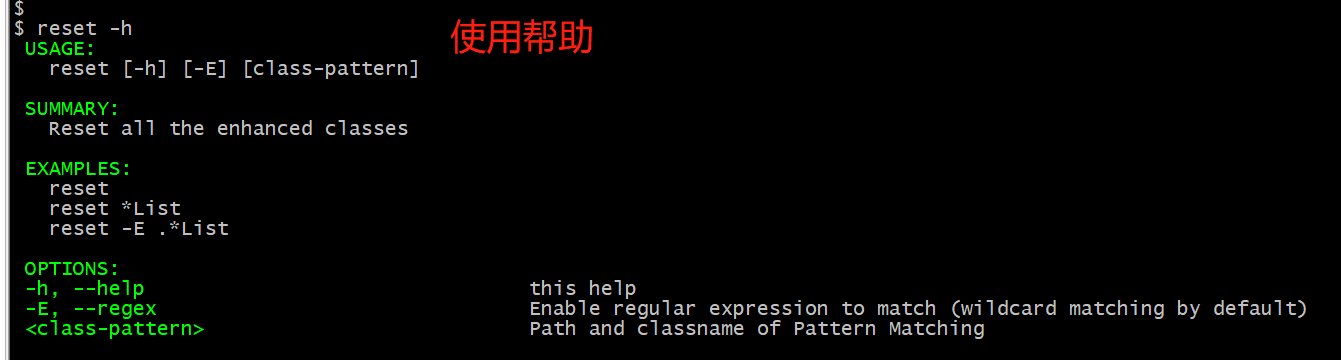
查看当前会话的信息



### reset

重置增强类，将被 Arthas 增强过的类全部还原，Arthas 服务端**shutdown**时会重置所有增强过的类

#### 使用参考



* [reset](https://alibaba.github.io/arthas/reset.html)——重置增强类，将被 Arthas 增强过的类全部还原，Arthas 服务端关闭时会重置所有增强过的类
* version——输出当前目标 Java 进程所加载的 Arthas 版本号
* history——打印命令历史
* quit——退出当前 Arthas 客户端，其他 Arthas 客户端不受影响
* shutdown——关闭 Arthas 服务端，所有 Arthas 客户端全部退出
* [keymap](https://alibaba.github.io/arthas/keymap.html)——Arthas快捷键列表及自定义快捷键

## jvm相关

* [dashboard](https://alibaba.github.io/arthas/dashboard.html)——当前系统的实时数据面板
* [thread](https://alibaba.github.io/arthas/thread.html)——查看当前 JVM 的线程堆栈信息
* [jvm](https://alibaba.github.io/arthas/jvm.html)——查看当前 JVM 的信息
* [sysprop](https://alibaba.github.io/arthas/sysprop.html)——查看和修改JVM的系统属性
* [sysenv](https://alibaba.github.io/arthas/sysenv.html)——查看JVM的环境变量
* [getstatic](https://alibaba.github.io/arthas/getstatic.html)——查看类的静态属性
* **New!** [ognl](https://alibaba.github.io/arthas/ognl.html)——执行ognl表达式
* **New!** [mbean](https://alibaba.github.io/arthas/mbean.html)——查看 Mbean 的信息

## class/classloader相关

* [sc](https://alibaba.github.io/arthas/sc.html)——查看JVM已加载的类信息
* [sm](https://alibaba.github.io/arthas/sm.html)——查看已加载类的方法信息
* [jad](https://alibaba.github.io/arthas/jad.html)——反编译指定已加载类的源码
* [mc](https://alibaba.github.io/arthas/mc.html)——内存编绎器，内存编绎.java文件为.class文件
* [redefine](https://alibaba.github.io/arthas/redefine.html)——加载外部的.class文件，redefine到JVM里
* [dump](https://alibaba.github.io/arthas/dump.html)——dump 已加载类的 byte code 到特定目录
* [classloader](https://alibaba.github.io/arthas/classloader.html)——查看classloader的继承树，urls，类加载信息，使用classloader去getResource

## monitor/watch/trace相关

请注意，这些命令，都通过字节码增强技术来实现的，会在指定类的方法中插入一些切面来实现数据统计和观测，因此在线上、预发使用时，请尽量明确需要观测的类、方法以及条件，诊断结束要执行 shutdown 或将增强过的类执行 reset 命令。

* [monitor](https://alibaba.github.io/arthas/monitor.html)——方法执行监控
* [watch](https://alibaba.github.io/arthas/watch.html)——方法执行数据观测
* [trace](https://alibaba.github.io/arthas/trace.html)——方法内部调用路径，并输出方法路径上的每个节点上耗时
* [stack](https://alibaba.github.io/arthas/stack.html)——输出当前方法被调用的调用路径
* [tt](https://alibaba.github.io/arthas/tt.html)——方法执行数据的时空隧道，记录下指定方法每次调用的入参和返回信息，并能对这些不同的时间下调用进行观测

## options

* [options](https://alibaba.github.io/arthas/options.html)——查看或设置Arthas全局开关

## 管道

Arthas支持使用管道对上述命令的结果进行进一步的处理，如sm java.lang.String \* | grep 'index'

* grep——搜索满足条件的结果
* plaintext——将命令的结果去除ANSI颜色
* wc——按行统计输出结果

## 后台异步任务

当线上出现偶发的问题，比如需要watch某个条件，而这个条件一天可能才会出现一次时，异步后台任务就派上用场了.