

---

# 老男孩教育-online-Tomcat

---

老男孩教育-online-Tomcat

运维排错那点事

排错基本思维

课程概述

预报姿势

## 2. 期末架构服务-tomcat

2.1 jre-jdk-jvm

2.2 环境准备 与 部署

2.3 Tomcat服务管理

2.3.1 tomcat 相关命令

2.3.2 Tomcat目录结构

2.4 小试牛刀-部署简单Java页面

2.5 Tomcat管理端

2.6 tomcat日志与进程信息

2.6.1 进程信息

2.6.2 Tomcat日志

2.7 Tomcat通过SystemCtl管理

2.8 Tomcat配置文件

2.9 Tomcat 日志格式

2.10 部署Tomcat多虚拟主机

1) 环境准备

2) 配置与调试

3) 重看-tomcat 处理请求流程

2.11 Java应用部署方式

1) 部署应用方式

2) jar包运行

3) java开源软件-war包

3) 部署应用-zrlog

4) 部署 jpress应用(作业)

2.8 Tomcat 3种工作模式: bio , nio , apr

1) io模型区别

2) 查看当前使用的io模型

3) 修改io模型

2.8 tomcat 多实例

2.10 Tomcat 监控功能

2.10.1 命令

2.10.2 开启tomcat监控功能

2.10.3 tomcat远程监控功能小结

2.11 Tomcat相关故障及排错

2.10 https

1) 单台tomcat

2) tomcat集群部署 https

2.11 tomcat 与 nginx

2.12 Tomcat优化-优化部分-架构优化项目进行讲解

2.12.1 Tomcat安全优化

2.13 Tomcat性能优化

2.13.1 调优思路与工具

2.13.2 Jmeter使用指南

2.13.3 Tomcat准备

2.13.4 进行基准测试

2.13.5 Tomcat 配置参数优化 ✨ ✨ ✨ ✨ ✨

**2.14 Tomcat总结**

## 运维排错那点事

---

☒ 防火墙:

☒ 开源: iptables, firewallld

☒ 云: 安全组

☒ 硬件: ....

☒ 负载均衡:

☒ 4层

☒ 7层

☒ 负载均衡无法访问,直接检查后端节点web服务

☒ 配合抓包: wireshark, tcpdump

☒ Web服务:

☒ Inmp

☒ Inmt

☒ 检查环境是否ok nginx? nginx+php? php+mysql?

☒ 检查应用配置(数据库,权限....)

☒ nginx处理用户请求的流程

☒ 配合: 主要看日志 辅助抓包

☒ 数据服务

☒ nfs

☒ 数据库

☒ 备份服务

☒ rsyncd 日志,命令行提示....

☒ 运维服务:

☒ 跳板机/堡垒机

☒ teleport

☒ jumpserver

☒ 日志分析

☒ goaccess

☒ OpenVPN

☒ 抓包和日志

## 排错基本思维

- 目标:定位是哪一个服务有问题,进一步排查
- [1] 通过各种监控工具
- [2] 通过监控定位到某台或某个集群
- [3] 找出一台机器,进行精确的定位
- [4] 排查出指定的服务(ps top,.....)
- [5] 定位到服务,进一步的排查
  - [A] 服务状态
  - [B] 服务的各种日志
  - [AB] 抓住关键报错提示: [OK] info(正常信息)    [!] 异常 **error err failed failure stderr ....**
  - [abc] 根据错误提示进一步排查,指定的问题 .....
- [6] 解决故障后,进行总结.
- [!] 没有日志生成日志,日志不详细开启debug级别日志.

## 课程概述

☒ tomcat jvm jre jdk    \*\*\*\*\*

☒ tomcat 环境部署

☒ jdk 1.8

☒ tomcat 8.5.50 9.0.52

☒ tomcat目录结构    \*\*\*\*\*

☒ bin/ startup.sh shutdown.sh catalina.sh (配置tomcat监控功能    配置java(tomcat) 启动参数 jvm优化)

☒ conf/ **server.xml**    web.xml    tomcat-user.xml

☒ logs / catalina.out (每日切割 切割后文件内容不会被清空)    catalina.2019-12-12.log    access.log

☒ Deploying web application

☒ error

- ☐ 那些年你看过的日志?
- ☐ tomcat: catalina.out    localhost\_access\_log.2019-12-17.txt
- ☒ webapps ❸❸❸❸❸❸
- ☒ 配置tomcat管理端
- ☒ tomcat的配置文件: **tomcat默认端口 虚拟主机** ❸❸❸❸❸
- ☒ tomcat 部署应用及方式 ❸❸❸
  - ☒ war
  - ☒ jar java -jar xxxx.jar
- ☐ **tomcat3种工作模式: bio , nio , apr** ❸❸❸
- ☒ 通过systemctl 管理tomcat❸❸❸❸
- ☐ **tomcat 多实例:** 端口不同 路径 ❸❸❸
- ☐ **tomcat 监控功能** ❸❸❸
- ☐ tomcat 与 nginx https ❸❸❸❸❸
- ☐ **tomcat 故障及排查 jstack jmap** ❸❸❸❸
- ☐ **tomcat 优化**
- ☐ 安全优化
  - ☐ 性能优化
- ☐ tomcat 性能及压力测试

## 预报姿势

---

- web服务器: lnmt (nginx+java)
  - 中间件:nginx+动态
  - 数据库中间件: redis/读写分离MyCAT.....
- tomcat 处理java程序代码
- ◦ **tomcat** ☒
- resin
  - JBoss
- weblogic (Oracle,配合oracle数据库,甲骨文)
  - 其他。 . . .
- 

## 2. 期末架构服务-tomcat

---

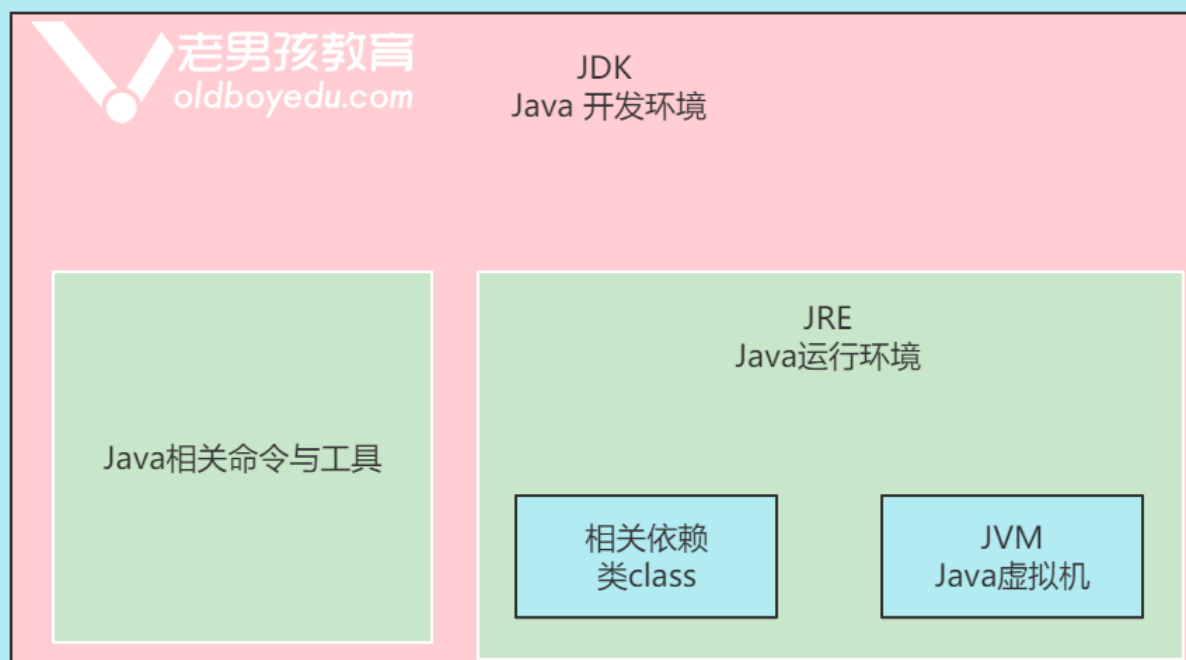
### 2.1 jre-jdk-jvm

---

- 正常书写的程序代码,只能在某个平台使用

- linux--->windows
  - window QQ.exe 上传 ----> linux
- java 只要书写一份java代码,把代码放在 java虚拟机中运行. 只要java虚拟机可以运行,java代码就可以正常运行.
- jvm java virtual machine java虚拟机
- **Write Once ,Run Anywhere.** 一次书写(编译),处处运行.
- java jvm 解决代码可移植性问题(跨平台)
- **jre java runtime enviroment java运行环境(提供jvm)**
- **jdk java delelopment kit java开发环境(很多内容)** = jre + 额外java工具
- jvm ,jre,jdk
- JVM java virtual machine java虚拟机
  - 1份代码 想在不同的系统使用
  - java 程序代码 运行在java虚拟机中 只要系统能有java环境(java虚拟机) 就可以运行代码
  - 1份代码 处处使用问题 代码可移植性
  - 对于 java虚拟机 一般关注 **内存使用情况**
- JDK
  - java runtime environment java运行环境
- JRE
  - java development kit java开发环境

## 老男孩教育1202年-最新架构-JVM-JRE-JDK



## 2.2 环境准备 与 部署

| Tomcat |            |          |            |
|--------|------------|----------|------------|
| web03  | jdk tomcat | 10.0.0.9 | 172.16.1.9 |

- jdk tomcat版本选择
- jdk 版本 一般根据开发使用的版本为准 这里我们选择:1.8.0
  - jdk(Oracle官方) [最新版jdk下载地址](#) [jdk各种版本旧版本下载地址](#)
    - rpm包安装
    - 二进制包 (软件绿色版,解压即用)
    - 源码 (编译安装)
  - openJDK Linux
    - yum `yum list |grep -i openjdk`
- tomcat 9.0 8.5 8.0 7.x
  - apache-tomcat

Apache Tomcat®

Tomcat 9 Software Downloads

Welcome to the Apache Tomcat® 9.x software download page. This page provides download links for obtaining the latest version of Tomcat 9.0.x software, as well as links to the archives of Unsure which version you need? Specification versions implemented, minimum java version required and lots more useful information may be found on the ["which version?"](#) page.

Quick Navigation

[KEYS](#) | [9.0.52](#) | [Browse](#) | [Archives](#)

Release Integrity

You **must** [verify](#) the integrity of the downloaded files. We provide OpenPGP signatures for every release file. This signature should be matched against the [KEYS](#) file. After you download the file, you should calculate a checksum for your download, and make sure it is the same as ours.

Mirrors

You are currently using <https://mirrors.tuna.tsinghua.edu.cn/apache/>. If you encounter a problem with this mirror, please select another mirror. If all mirrors are unavailable, please contact [the Apache Tomcat Project](#).

Other mirrors: <https://mirrors.bfsu.edu.cn/apache/> [Change](#)

9.0.52

Please see the [README](#) file for packaging information. It explains what every distribution contains.

Binary Distributions

#web03

##jdk

#各种版本: <https://mirrors.tuna.tsinghua.edu.cn/apache/tomcat/>

####01 部署 jdk

```
[root@oldboy-tomcat ~]# tar xf /app/tools/jdk-8u241-linux-x64.tar.gz -C /app/
[root@oldboy-tomcat ~]# ln -s /app/jdk1.8.0_241/ /app/jdk
[root@oldboy-tomcat ~]# ll /app
total 0
lrwxrwxrwx 1 root root 18 Feb 7 10:26 jdk -> /app/jdk1.8.0_241/
drwxr-xr-x 7 10143 10143 245 Dec 11 18:39 jdk1.8.0_241
drwxr-xr-x 2 root root 75 Feb 7 10:13 tools
```

####02 java jdk 环境变量

```
cat >>/etc/profile<<'EOF'
export JAVA_HOME=/app/jdk
export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib:$JAVA_HOME/lib/tools.jar
export TOMCAT_HOME=/app/tomcat
EOF
```

#说明部分

##export JAVA\_HOME=/app/jdk

#jdk目录

```
##export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH          #配置命令路径
##export CLASSPATH=.$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib:$JAVA_HOME/lib/tools.jar
##export TOMCAT_HOME=/app/tomcat                                #tomcat 目录
```

```
[root@oldboy-tomcat ~]# . /etc/profile #或者source /etc/profile
```

####03. 检查 jdk是否部署完成

```
[root@oldboy-tomcat ~]# java -version
```

```
java version "1.8.0_241"
```

```
Java(TM) SE Runtime Environment (build 1.8.0_241-b07)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 25.241-b07, mixed mode)
```

####04. 部署 tomcat

```
[root@oldboy-tomcat ~]# tar xf /app/tools/apache-tomcat-8.5.50.tar.gz -C /app/
```

```
[root@oldboy-tomcat ~]# ln -s /app/apache-tomcat-8.5.50/ /app/tomcat
```

####05 检查 jdk+tomcat

```
[root@oldboy-tomcat ~]# /app/tomcat/bin/version.sh
```

```
Using CATALINA_BASE: /app/tomcat
```

```
Using CATALINA_HOME: /app/tomcat
```

```
Using CATALINA_TMPDIR: /app/tomcat/temp
```

```
Using JRE_HOME: /app/jdk
```

```
Using CLASSPATH: /app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar
```

```
Server version: Apache Tomcat/8.5.50
```

```
Server built: Dec 7 2019 19:19:46 UTC
```

```
Server number: 8.5.50.0
```

```
OS Name: Linux
```

```
OS Version: 4.18.0-80.11.2.el8_0.x86_64
```

```
Architecture: amd64
```

```
JVM Version: 1.8.0_241-b07
```

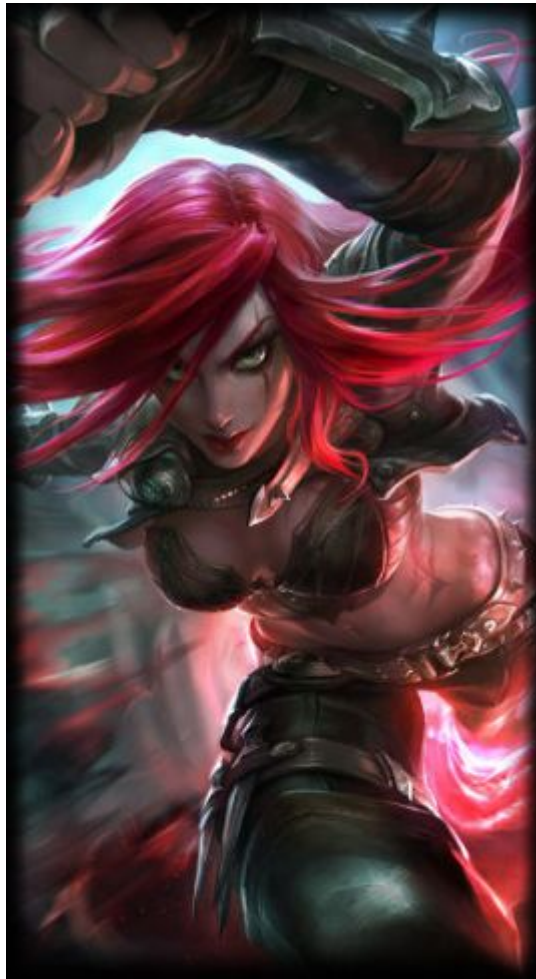
```
JVM Vendor: Oracle Corporation
```

## 2.3 Tomcat服务管理

### 2.3.1 tomcat 相关命令

- tomcat 管理命令

| bin目录            | 含义           | 主要命令                                                                                      |
|------------------|--------------|-------------------------------------------------------------------------------------------|
| /app/tomcat/bin/ | tomcat管理命令目录 | startup.sh 启动tomcat                                                                       |
|                  |              | shutdown.sh 关闭tomcat                                                                      |
|                  |              | version.sh 部署完成后 检查 jdk 与tomcat                                                           |
|                  |              | catalina.sh 卡特琳娜#tomcat核心脚本 startup shutdown 都会调用这个脚本<br>#配置 java启动参数 ,tomcat远程管理 配置jvm参数 |



```
[root@web01 tools]# #/application/tomcat/bin/startup.sh
[root@web01 tools]# #/application/tomcat/bin/catalina.sh start
```

```
[root@web01 tools]# /application/tomcat/bin/startup.sh
Using CATALINA_BASE:   /application/tomcat
Using CATALINA_HOME:   /application/tomcat
Using CATALINA_TMPDIR: /application/tomcat/temp
Using JRE_HOME:        /application/jdk
Using CLASSPATH:       /application/tomcat/bin/bootstrap.jar:/application/tomcat/bin/tomcat-juli.jar
Tomcat started.

[root@web01 tools]# ss -ltnup|grep tomcat
[root@web01 tools]# ss -ltnup|grep java
tcp        LISTEN     0          100        :::8009           :::*              users:
(("java",pid=58497,fd=51))
tcp        LISTEN     0          100        :::8080           :::*              users:
(("java",pid=58497,fd=46))
tcp        LISTEN     0          1        ::ffff:127.0.0.1:8005  :::*              users:
(("java",pid=58497,fd=66))
[root@web01 tools]# ps -ef |grep java
root      58497      1   2   12:08 pts/0    00:00:02 /application/jdk/bin/java -
Djava.util.logging.config.file=/application/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -
Djava.endorsed.dirs=/application/tomcat/endorsed -classpath
/application/tomcat/bin/bootstrap.jar:/application/tomcat/bin/tomcat-juli.jar -
Dcatalina.base=/application/tomcat -Dcatalina.home=/application/tomcat -
Djava.io.tmpdir=/application/tomcat/temp org.apache.catalina.startup.Bootstrap start
```



```
root      58547  58190  0 12:09 pts/0    00:00:00 grep --color=auto java
[root@web01 tools]#


/app/jdk/bin/java
-Djava.util.logging.config.file=/app/tomcat/conf/logging.properties
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djdk.tls.ephemeralDHKeySize=2048
-Djava.protocol.handler.pkgs=org.apache.catalina.webresources
-Dorg.apache.catalina.security.SecurityListener.UMASK=0027
-Dignore.endorsed.dirs=
-classpath /app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat
-juli.jar
-Dcatalina.base=/app/tomcat
-Dcatalina.home=/app/tomcat
-Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap
start
```

- web访问 10.0.0.7:8080


← → 🔒 不安全 | 10.0.0.9:8080 🔍 ☆ ▼

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

## Apache Tomcat/9.0.52

 **APACHE**™ SOFTWARE FOUNDATION  
<http://www.apache.org/>

If you're seeing this, you've successfully installed Tomcat. Congratulations!



**Recommended Reading:**

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Server Status

Manager App

Host Manager

**Developer Quick Start**

|                                       |                                  |                          |                                        |
|---------------------------------------|----------------------------------|--------------------------|----------------------------------------|
| <a href="#">Tomcat Setup</a>          | <a href="#">Realms &amp; AAA</a> | <a href="#">Examples</a> | <a href="#">Servlet Specifications</a> |
| <a href="#">First Web Application</a> | <a href="#">JDBC DataSources</a> |                          | <a href="#">Tomcat Versions</a>        |

## 2.3.2 Tomcat目录结构

| 目录      | 含义                                                                                                                                                                                                                                  |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bin     | startup.sh<br>shutdown.sh<br>catalina.sh                                                                                                                                                                                            |
| conf/   | tomcat配置文件<br><b>server.xml #nginx.conf 主配置文件</b><br>web.xml #补充 额外功能<br>tomcat-user.xml # <b>tomcat管理端</b> 配置文件 用户名 和 密码                                                                                                           |
| logs    | 日志<br><b>catalina.out #tomcat最全日志 查看 error startup 启动的时间</b> catalina.out切割之后内容不变,未来需要logrotate切割<br>catalina.2019-12-16.log #catalina.out的 <b>每天的切割日志</b> ,<br>localhost_access_log.2019-12-16.txt # <b>tomcat access.log 访问日志</b> |
| webapps | tomcat站点目录 nginx html war包                                                                                                                                                                                                          |

```
[root@web01 tomcat]# ll
total 92
drwxr-xr-x 2 root root 4096 Dec 16 12:01 bin
drwxr-xr-x 3 root root 198 Dec 16 12:08 conf
drwxr-xr-x 2 root root 4096 Dec 16 12:01 lib #库文件 tomcat插件
drwxr-xr-x 2 root root 197 Dec 16 12:08 logs
-rw-r--r-- 1 root root 1444 Sep 28 2015 NOTICE
-rw-r--r-- 1 root root 6741 Sep 28 2015 RELEASE-NOTES
-rw-r--r-- 1 root root 16204 Sep 28 2015 RUNNING.txt
drwxr-xr-x 2 root root 30 Dec 16 12:01 temp
drwxr-xr-x 7 root root 81 Sep 28 2015 webapps
drwxr-xr-x 3 root root 22 Dec 16 12:08 work
```

## 2.4 小试牛刀-部署简单Java页面

```
#应用war包 部署到 /app/tomcat/webapps/
#tomcat 自动解压
#tomcat 自动部署
```



JVM memory detail info :  
 Max memory:235MB  
 Total memory:40MB  
 Free memory:12MB  
 Available memory can be used is :207MB

- 日志信息

```
26-Aug-2021 10:58:13.011 信息 [Catalina-utility-2] org.apache.catalina.startup.HostConfig.deployWAR 正在部署web应用程序存档文件[/app/apache-tomcat-9.0.52/webapps/memtest.war]
26-Aug-2021 10:58:13.089 信息 [Catalina-utility-2] org.apache.catalina.startup.HostConfig.deployWAR web应用程序存档文件[/app/apache-tomcat-9.0.52/webapps/memtest.war]的部署已在[77]ms内完成
```

```
26-Aug-2021 10:58:13.011 信息 [Catalina-utility-2] org.apache.catalina.startup.HostConfig.deployWAR 正在部署web应用程序存档文件[/app/apache-tomcat-9.0.52/webapps/memtest.war]
26-Aug-2021 10:58:13.089 信息 [Catalina-utility-2] org.apache.catalina.startup.HostConfig.deployWAR web应用程序存档文件[/app/apache-tomcat-9.0.52/webapps/memtest.war]的部署已在[77]ms内完成
```

## 2.5 Tomcat管理端

| Tomcat管理端应用场景              |                  |  |
|----------------------------|------------------|--|
| 搭建与测试的时候 开启管理端 <b>进行调试</b> | 开启管理端 debug      |  |
| 生产环境中(线上环境)                | 关闭管理端 ,清除管理端相关文档 |  |
|                            |                  |  |

- tomcat 9.0
  - 注意: tomcat 8.5 对管理端限制更严格
  - 要配置tomcat-user.xml之外
  - 还限制 只能在本地使用127.0.0.1 访问管理的
  - 从tomcat8.5开始 管理端默认只能通过 本地使用 127.0.0.1 访问 (类似于nginx `allow 127.0.0.1 ; deny all;`)
  - 默认情况下, 只能从与Tomcat运行在同一台计算机上的浏览器访问管理器。如果要修改此限制, 则需要编辑管理器的context.xml文件。

```
## 01 配置 tomcat 管理端
[root@static01 tomcat]# cat conf/tomcat-users.xml
<?xml version="1.0" encoding="UTF-8"?>
<tomcat-users xmlns="http://tomcat.apache.org/xml"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
               version="1.0">
  <role rolename="manager-gui"/>
  <role rolename="admin-gui"/>
  <user username="tomcat" password="1" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

## 02 重启tomcat

## 03 命令行测试

```
curl -u tomcat:1 http://127.0.0.1:8080/manager/status
```

#注意: 使用127.0.0.1

## 04 开启tomcat默认的限制

```
/app/tomcat/webapps/host-manager/META-INF/context.xml
/app/tomcat/webapps/manager/META-INF/context.xml
```

```
[root@static01 tomcat]# sed -i.bak 's#127#\d+#g' /app/tomcat/webapps/host-manager/META-INF/context.xml
/app/tomcat/webapps/manager/META-INF/context.xml

[root@static01 tomcat]# find -name "context.xml" |xargs grep allow
./webapps/host-manager/META-INF/context.xml:         allow="\d+\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
./webapps/manager/META-INF/context.xml:         allow="\d+\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
```





服务器状态

管理器

应用程序列表

HTML 管理器帮助

管理器帮助

完整的服务器状态

服务器信息

| Tomcat 版本            | JVM 版本       | JVM 提供商            | OS 名称 | 操作系统版本                      | 操作系统架构 | 主机名      | IP 地址    |
|----------------------|--------------|--------------------|-------|-----------------------------|--------|----------|----------|
| Apache Tomcat/9.0.52 | 1.8.0_60-b27 | Oracle Corporation | Linux | 3.10.0-1160.31.1.el7.x86_64 | amd64  | static01 | 10.0.0.9 |

JVM

剩余内存: 12.66 MB 总内存: 43.23 MB 最大内存 235.87 MB

| 内存池                    | 类型              | 初始化      | 总共       | 最大值        | 已用             |
|------------------------|-----------------|----------|----------|------------|----------------|
| Eden Space             | Heap memory     | 4.31 MB  | 12.06 MB | 65.06 MB   | 3.91 MB (6%)   |
| Survivor Space         | Heap memory     | 0.50 MB  | 1.43 MB  | 8.12 MB    | 1.43 MB (17%)  |
| Tenured Gen            | Heap memory     | 10.68 MB | 29.73 MB | 162.68 MB  | 25.01 MB (15%) |
| Code Cache             | Non-heap memory | 2.43 MB  | 8.93 MB  | 240.00 MB  | 8.80 MB (3%)   |
| Compressed Class Space | Non-heap memory | 0.00 MB  | 3.00 MB  | 1024.00 MB | 2.74 MB (0%)   |
| Metaspace              | Non-heap memory | 0.00 MB  | 27.75 MB | -0.00 MB   | 26.92 MB       |

"http-nio-8080"

最大线程: 200 当前线程数: 10 当前线程繁忙: 1 存活连接子总数: 1  
最大处理时间: 1463 ms 处理时间: 1.986 s 请求总数: 74 错误数: 17 收到字节: 0.00 MB 发送字节: 0.63 MB

| 阶段 | 时间    | 发送字节 | 接收字节 | 客户端 (转发) | 客户端 (实际) | 虚拟主机     | 请求                           |
|----|-------|------|------|----------|----------|----------|------------------------------|
| R  | ?     | ?    | ?    | ?        | ?        | ?        |                              |
| R  | ?     | ?    | ?    | ?        | ?        | ?        |                              |
| S  | 15 ms | 0 KB | 0 KB | 10.0.0.1 | 10.0.0.1 | 10.0.0.9 | GET /manager/status HTTP/1.1 |
| R  | ?     | ?    | ?    | ?        | ?        | ?        |                              |

P: 解析和设置 request S: 服务 F: 结果 R: 就绪 K: 存活





Tomcat Web应用程序管理者

消息: OK

管理器

应用程序列表

HTML 管理器帮助

管理器帮助

服务

应用程序

| 路径            | 版本号 | 显示名称                            | 运行中  | 会话 | 命令                               |
|---------------|-----|---------------------------------|------|----|----------------------------------|
| /             | 未指定 | Welcome to Tomcat               | true | 0  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |
| /docs         | 未指定 | Tomcat Documentation            | true | 0  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |
| /examples     | 未指定 | Servlet and JSP Examples        | true | 0  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |
| /host-manager | 未指定 | Tomcat Host Manager Application | true | 1  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |
| /manager      | 未指定 | Tomcat Manager Application      | true | 1  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |
| /memtest      | 未指定 |                                 | true | 1  | 启动 停止 重新加载 卸载<br>过期会话 闲置 ≥ 30 分钟 |



## 2.6 tomcat日志与进程信息

### 2.6.1 进程信息

```
#ps命令查询 java进程信息
[root@oldboy-tomcat ~]# ps -ef |grep java
root      4112    1  1 11:43 pts/1    00:00:15 /app/jdk/bin/java -
Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHkeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -classpath
/app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat -
Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap start
root      4199  4175  0 11:59 pts/0    00:00:00 grep --color=auto java
[root@oldboy-tomcat ~]#

/app/tomcat/bin/startup.sh

/app/jdk/bin/java
-Djava.util.logging.config.file=/app/tomcat/conf/logging.properties
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djdk.tls.ephemeralDHkeySize=2048
-Djava.protocol.handler.pkgs=org.apache.catalina.webresources
-Dorg.apache.catalina.security.SecurityListener.UMASK=0027
-Dignore.endorsed.dirs=
-classpath /app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar
-Dcatalina.base=/app/tomcat
-Dcatalina.home=/app/tomcat
-Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap
start

/app/tomcat_8081
/app/tomcat_8082
```

### 2.6.2 Tomcat日志

|                             |         |  |
|-----------------------------|---------|--|
|                             |         |  |
| conf/logging.properties     | 日志的配置文件 |  |
| catalina.out ****           | 持续增加    |  |
| catalina-年-月-日.log          | 切割日志    |  |
| xxxxxxaccess.log ****       | 访问日志    |  |
| host-manager.2021-08-26.log | 管理端日志   |  |
| localhost.2021-08-26.log    | 管理端日志   |  |
| manager.2021-08-26.log      | 管理端日志   |  |
|                             |         |  |

- catalina.out 主日志

###catalina.out

# error 错误

# failed

# exception 异常

# startup 或 finished 启动所需的时间

# deploy

```
07-Feb-2020 12:03:16.039 INFO [main] org.apache.catalina.core.StandardServer.await A valid shutdown command
was received via the shutdown port. Stopping the Server instance.
07-Feb-2020 12:03:16.039 INFO [main] org.apache.coyote.AbstractProtocol.pause Pausing ProtocolHandler
["http-nio-8080"]
07-Feb-2020 12:03:16.046 INFO [main] org.apache.coyote.AbstractProtocol.pause Pausing ProtocolHandler
["ajp-nio-8009"]
07-Feb-2020 12:03:16.051 INFO [main] org.apache.catalina.core.StandardService.stopInternal Stopping service
[Catalina]
07-Feb-2020 12:03:16.093 INFO [main] org.apache.coyote.AbstractProtocol.stop Stopping ProtocolHandler
["http-nio-8080"]
07-Feb-2020 12:03:16.098 INFO [main] org.apache.coyote.AbstractProtocol.stop Stopping ProtocolHandler
["ajp-nio-8009"]
07-Feb-2020 12:03:16.100 INFO [main] org.apache.coyote.AbstractProtocol.destroy Destroying ProtocolHandler
["http-nio-8080"]
07-Feb-2020 12:03:16.101 INFO [main] org.apache.coyote.AbstractProtocol.destroy Destroying ProtocolHandler
["ajp-nio-8009"]
```

- 访问日志

```
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /tomcat.css HTTP/1.1" 200 5581
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /tomcat.png HTTP/1.1" 200 5103
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /bg-nav.png HTTP/1.1" 200 1401
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /asf-logo-wide.svg HTTP/1.1" 200 27235
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /bg-upper.png HTTP/1.1" 200 3103
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /bg-middle.png HTTP/1.1" 200 1918
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /bg-button.png HTTP/1.1" 200 713
223.104.2.165 - - [07/Feb/2020:12:10:00 +0800] "GET /favicon.ico HTTP/1.1" 200 21630
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET / HTTP/1.1" 200 11215
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /tomcat.css HTTP/1.1" 200 5581
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /tomcat.png HTTP/1.1" 200 5103
```

```

223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /bg-nav.png HTTP/1.1" 200 1401
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /asf-logo-wide.svg HTTP/1.1" 200 27235
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /bg-upper.png HTTP/1.1" 200 3103
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /bg-button.png HTTP/1.1" 200 713
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /bg-middle.png HTTP/1.1" 200 1918
223.104.2.165 - - [07/Feb/2020:12:10:06 +0800] "GET /favicon.ico HTTP/1.1" 200 21630
223.104.2.165 - - [07/Feb/2020:12:10:11 +0800] "GET / HTTP/1.1" 200 11215
223.104.2.165 - - [07/Feb/2020:12:10:11 +0800] "GET /tomcat.css HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /tomcat.png HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /asf-logo-wide.svg HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /bg-nav.png HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /bg-upper.png HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /bg-button.png HTTP/1.1" 304 -
223.104.2.165 - - [07/Feb/2020:12:10:12 +0800] "GET /bg-middle.png HTTP/1.1" 304 -

```

- catalina.out 和 catalina.日期.log
- catalina.out在被切割后 内容不会被清空

## 2.7 Tomcat通过SystemCtl管理

- 各种服务通过yum、rpm安装自动配置systemctl
- 通过编译安装或二进制安装，手动书写systemctl配置，达到管理服务目标。

```

#找个例子对比书写 sshd nginx
[root@static01 tomcat]# systemctl cat sshd
# /usr/lib/systemd/system/sshd.service
[Unit]
Description=OpenSSH server daemon          #描述 注释
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.service    #依赖 在xxxx服务之后运行。
Wants=sshd-keygen.service

[Service]
Type=notify                                #服务类型
EnvironmentFile=/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D $OPTIONS        #启动sshd 服务的命令
ExecReload=/bin/kill -HUP $MAINPID          #重新加载sshd配置文件 优雅的重启
KillMode=process                           #指定如何关闭
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target

[root@static01 tomcat]# systemctl cat nginx
# /usr/lib/systemd/system/nginx.service
[Unit]
Description=The nginx HTTP and reverse proxy server
After=network-online.target remote-fs.target nss-lookup.target #依赖
Wants=network-online.target

[Service]
Type=forking                                #服务的另一种模式

```

```

PIDFile=/run/nginx.pid

ExecStartPre=/usr/bin/rm -f /run/nginx.pid    #pre xxx之前 启动nginx服务之前
ExecStartPre=/usr/sbin/nginx -t              #          检查语法
ExecStart=/usr/sbin/nginx                    #启动的命令 systemctl start nginx --->
ExecReload=/usr/sbin/nginx -s reload          #重启 重载

#ExecStart    ExecReload/ExecRestart    ExecStop

KillSignal=SIGQUIT
TimeoutStopSec=5
KillMode=process
PrivateTmp=true

[Install]
WantedBy=multi-user.target

##补充01
#Type
## 对于常见服务推荐使用forking 模式即可。

## oneshot 适用于一次的服务

## 补充02
EnvironmentFile=/etc/sysconfig/sshhd    #服务使用的环境变量的配置文件。

```

| systemctl 配置文件格式 |                  |  |
|------------------|------------------|--|
| [Unit]           | 配置说明信息和依赖信息      |  |
| [Service] 旻旻旻旻旻  | 这个服务端开启/关闭/重启的命令 |  |
| [Install]        | 指定运行级别 target    |  |

- 管理tomcat

```

[root@static01 tomcat]# cat /etc/sysconfig/tomcat
#变量=内容
JAVA_HOME=/app/jdk
PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin
CLASSPATH=.$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib:$JAVA_HOME/lib/tools.jar
[root@static01 tomcat]# cat /usr/lib/systemd/system/tomcat.service
[Unit]
Description=Tomcat Server Manage
After=network.target remote-fs.target

[Service]
Type=forking
EnvironmentFile=/etc/sysconfig/tomcat
ExecStart=/app/tomcat/bin/startup.sh
ExecReload="/app/tomcat/bin/shutdown.sh && sleep 2 && /app/tomcat/bin/startup.sh"
ExecStop=/app/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target

```



```
#管理tomcat实例
###nginx systemctl 管理 配置
[Unit]
Description=nginx - high performance web
serverDocumentation=http://nginx.org/en/docs/
After=network-online.target remote-fs.target nss-lookup.target
Wants=network-online.target
[Service]
Type=forking
#服务类型:forking守护进程模式
#服务类型:oneshot一次性的服务 PIDFile=/var/run/nginx.pid
#非所有服务需要,指定服务的pid文件
#ExecStart ExecRreload ExecStop 开启服务,重启服务,关闭服务对应的命令.
ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf
#/bin/kill -s HUP PID号 nginx -s reload
ExecReload=/bin/sh -c "/bin/kill -s HUP $(/bin/cat /var/run/nginx.pid)"
ExecStop=/bin/sh -c "/bin/kill -s TERM $(/bin/cat /var/run/nginx.pid)"
Install]
WantedBy=multi-user.target
#指定用户的运行级别
#开始 书写tomcat的
```

```
[root@web03 /usr/lib/systemd/system]# cat tomcat.service [Unit]Description=The tomcat 管理配置After=network-
online.target remote-fs.target nss-lookup.targetWants=network-
online.target[Service]Type=forkingExecStart=/app/tomcat/bin/startup.shExecRestart="/app/tomcat/bin/shutdown
.sh && sleep 2 && /app/tomcat/bin/startup.sh"ExecStop=/app/tomcat/bin/shutdown.sh[Install]WantedBy=multi-
user.target[root@web03 /usr/lib/systemd/system]# systemctl daemon-reload sed -i '2a source
/etc/profile' /app/tomcat*/bin/startup.sh /app/tomcat*/bin/shutdown.sh #systemctl 详细配置 官方地址
https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html-
single/system_administrators_guide/index
```

## 2.8 Tomcat配置文件

| Tomcat配置文件说明 |                             |                           |
|--------------|-----------------------------|---------------------------|
| 端口部分         | 8005 shutdown端口,关闭tomcat使用. | 默认只能127.0.0.1访问           |
|              | 8080                        | web页面端口http               |
|              | 8009                        | ajp协议使用的端口(用于与apache连接使用) |
|              |                             |                           |
|              |                             |                           |

```
[root@web03 /app/tomcat]# sed '/<!--.*-->/d;/^$/d;/<!--/,/-->/d' conf/server.xml
<?xml version="1.0" encoding="UTF-8"?>

#8005端口是 shutdown端口 shutdown="" #关闭tomcat暗号
<Server port="8005" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />

#tomcat 管理端配置 开始
<GlobalNamingResources>
  <Resource name="UserDatabase" auth="Container"
    type="org.apache.catalina.UserDatabase"
    description="User database that can be updated and saved"
    factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
    pathname="conf/tomcat-users.xml" />
</GlobalNamingResources>
#tomcat 管理端配置 结束

<Service name="Catalina">

#连接器 用户请求通过连接器 进入tomcat,然后经过tomcat处理.
#8080是tomcat默认的web服务的端口
#8443tomcat+https

  <Connector port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" /> #配置tomcat https

# defaultHost tomcat默认的主机(网站)

  <Engine name="Catalina" defaultHost="localhost">

    <Realm className="org.apache.catalina.realm.LockOutRealm">
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
        resourceName="UserDatabase"/>
    </Realm>

# Host name 域名 lidao.oldboylinux.com zrlog.oldboylinux.com ..
# appBase=webapps 网站默认的站点目录(网站的代码)
# Host name="zrlog.oldboylinux.com" appBase="/code/zrlog"

# unpackWARs 自动解压war包
# autoDeploy 自动部署

  <Host name="localhost" appBase="webapps"
    unpackWARs="true" autoDeploy="true">

# 配置tomcat的访问日志.
    <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

# 配置访问日志格式
#prefix 和suffix 日志文件名的格式

    prefix="localhost_access_log" suffix=".txt"

#日志里面的内容
    pattern="%h %l %u %t &quot;%r&quot; %s %b %D " />
  </Host>
</Engine>
</Service>
```

```
</Server>
```

|          | tomcat host       | nginx server                 |
|----------|-------------------|------------------------------|
| 域名       | name 域名           | server_name                  |
| 站点目录     | appBase="webapps" | root /usr/share/nginx/html ; |
| 自动解压war包 | unpackWARs="true" |                              |
| 自动部署     | autoDeploy="true" |                              |

## 2.9 Tomcat 日志格式

| 日志内容                     | tomcat日志格式     | nginx日志格式              |
|--------------------------|----------------|------------------------|
| 客户端ip地址                  | %h             | \$remote_add           |
| 远程用户名                    | %l             | \$remote_user          |
| 远程用户名                    | %u             | \$remote_user          |
| 时间和日期                    | %t             | \$time_local           |
|                          |                |                        |
| http请求报文的起始行 (请求方法 和uri) | %r             | \$request              |
| status http状态码           | %s             | \$status               |
| 文件/页面/资源大小(字节)           | %b             | \$body_bytes_sent      |
| tomcat处理请求耗时(ms)         | %D             | \$request_time         |
| 记录用户从哪里跳转过来的             | %{Referer}i    | \$http_referer         |
| 客户端浏览器                   | %{User-Agent}i | \$http_user_agent      |
| 记录用户真实ip地址               |                | \$http_x_forwarded_for |
| ⚠ 双引号&quot;              |                |                        |

tomcat日志格式 [传送门](#)

```
#修改tomcat访问日志格式

pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;%{Referer}i&quot; &quot;%{User-Agent}i&quot;"; />

<?xml version="1.0" encoding="UTF-8"?>
<Server port="8005" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
```

```

<Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
<Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
<Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />
<GlobalNamingResources>
  <Resource name="UserDatabase" auth="Container"
    type="org.apache.catalina.UserDatabase"
    description="User database that can be updated and saved"
    factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
    pathname="conf/tomcat-users.xml" />
</GlobalNamingResources>
<Service name="Catalina">
  <Connector port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" />
  <Engine name="Catalina" defaultHost="localhost">
    <Realm className="org.apache.catalina.realm.LockOutRealm">
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
        resourceName="UserDatabase"/>
    </Realm>
    <Host name="localhost" appBase="webapps"
      unpackWARs="true" autoDeploy="true">
      <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
        prefix="localhost_access_log" suffix=".txt"
        pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;" />
    </Host>
  </Engine>
</Service>

```

#server.xml

#端口部分

###8005 shutdown端口

22 行: <Server port="8005" shutdown="SHUTDOWN">

###tomcat shutdown端口 telnet/nc 连接到这个端口 输入暗号 tomcat将会关闭

###8080 http协议端口

```

69   <Connector port="8080" protocol="HTTP/1.1"
70       connectionTimeout="20000"
71       redirectPort="8443" />

```

###8009 ajp协议端口 与apache连接使用

```

115  <!-- Define an AJP 1.3 Connector on port 8009 -->
116  <!--
117  <Connector protocol="AJP/1.3"
118       address="::1"
119       port="8009"
120       redirectPort="8443" />
121  -->

```

#8009端口 是用来给 apache与tomcat进行连接使用

#现在 tomcat+nginx 可以把 这一行8009注释 提高tomcat性能

```

116   <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

```

提问:tomcat默认有几个端口 及作用 ?

#tomcat管理端 相应的配置

##管理端 实际生产环境 关闭

```
37 <GlobalNamingResources>
38 <!-- Editable user database that can also be used by
39      UserDatabaseRealm to authenticate users
40 -->
41 <Resource name="UserDatabase" auth="Container"
42           type="org.apache.catalina.UserDatabase"
43           description="User database that can be updated and saved"
44           factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
45           #指定管理端 密码文件
46           pathname="conf/tomcat-users.xml" />
47 </GlobalNamingResources>
```

#配置 tomcat 虚拟主机的内容

|             |              |
|-------------|--------------|
| Nginx       | tomcat       |
| Server_name | Host name 域名 |
| root        | appBase 站点目录 |

#unpackWARs #自动解压war包

#autoDeploy 自动部署 把代码加载到jvm内存中

```
148 <Host name="localhost" appBase="webapps"
149      unpackWARs="true" autoDeploy="true">
150
151 <!-- SingleSignOn valve, share authentication between web applications
152      Documentation at: /docs/config/valve.html -->
153 <!--
154 <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
155 -->
156
157 <!-- Access log processes all example.
158      Documentation at: /docs/config/valve.html
159      Note: The pattern used is equivalent to using pattern="common" -->
```

#配置 日志

```
160 <Valve className="org.apache.catalina.valves.AccessLogValve"
161      directory="logs"
162      #日志文件 前缀是localhost_access_log #日志后缀
163      prefix="access" suffix=".log"
```

#日志 里面的格式 &quot; html语言中的 双引号

#日志内容 类似于 log\_format

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

```
164 </Host>
```

##%h 客户端ip地址或者是域名

##%l (小写L) 远程用户

##%u 用户 Remote user that was authenticated (if any), else '-' (escaped if required)

##%t 时间 日期和时间

##&quot; 双引号

##%r 请求起始行 \$request

##%s \$status 状态码

##%b 大小

- tomcat日志格式 [传送门](#)

| 日志内容 | tomcat日志格式 | nginx日志格式 |
|------|------------|-----------|
|      |            |           |
|      |            |           |
|      |            |           |

## 2.10 部署Tomcat多虚拟主机

### 1) 环境准备

- 虚拟主机,,,一个网站. [www.oldboylinux.com](#) img.oldboylinux.com live.oldboylinux.com

| 域名                 |    | 站点         | 日志                        |                 |
|--------------------|----|------------|---------------------------|-----------------|
| localhost          | 默认 | webapps    | localhost_access.xxxx.log | /var/log/tomcat |
| img.etiantian.org  | 图片 | /data/img  | img_access.xxxx.log       | /var/log/tomcat |
| live.etiantian.org | 直播 | /data/live | live_access.xxxx.log      | /var/log/tomcat |

#### #目标

浏览器访问 img.etiantian.org 显示 img 内容  
浏览器访问 live.etiantian.org 显示 live 内容  
浏览器访问 其他 . . . . 显示localhost默认内容

### 2) 配置与调试

```
<Host name="localhost" appBase="webapps"
  unpackWARs="true" autoDeploy="true">
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
    prefix="localhost_access" suffix=".log"
    pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;" />
</Host>
<Host name="img.etiantian.org" appBase="/data/img"
  unpackWARs="true" autoDeploy="true">
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
    prefix="img_access" suffix=".log"
    pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;" />
</Host>
<Host name="live.etiantian.org" appBase="/data/live"
  unpackWARs="true" autoDeploy="true">
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
    prefix="live_access" suffix=".log"
    pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;" />
</Host>
```

```
mkdir -p /data/{live,img}/ROOT/ /var/log/tomcat
echo live >/data/live/ROOT/index.jsp
echo img > /data/img/ROOT/index.jsp
```

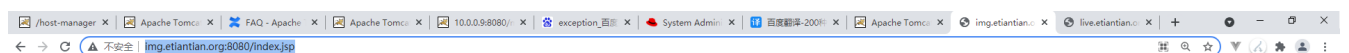
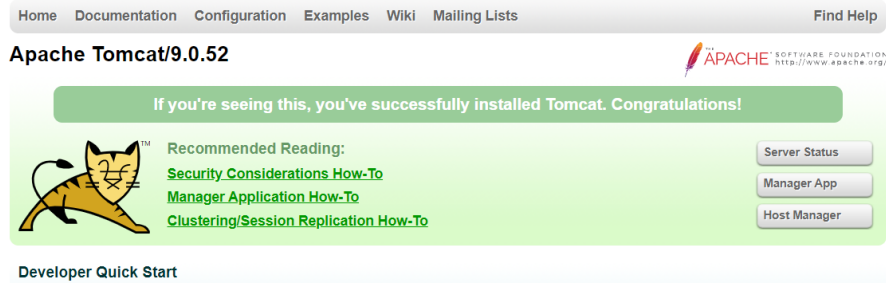
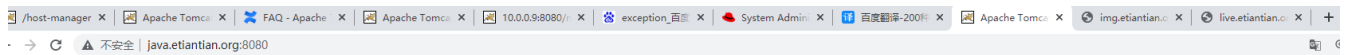
#tomcat访问的时候 uri小坑

### 用户访问的域名如果包含路径（目录） tomcat会在站点目录下面匹配指定的目录然后匹配文件

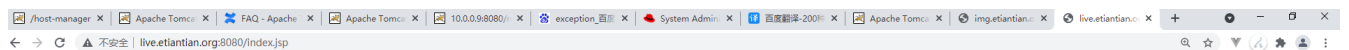
curl img.etiantian.org/oldboy/index.jsp ---> 站点目录下面的/oldboy/index.jsp

### 用户访问的url没有路径、目录 tomcat会在站点目录下面找ROOT/内容 进行匹配

curl img.etiantian.org/index.jsp ---> 站点目录下面的/ROOT/index.jsp



img



live

#tomcat虚拟主机

#01 第1部分 设置默认的虚拟主机

```
<Engine name="Catalina" defaultHost="localhost">
```

#02 第2个部分 虚拟主机的配置部分

#Host name="域名/localhost"

#appBase="站点目录" 网站程序代码存放的目录

#/code/live

#/code/img

#/code/blog

#/code/...

```
<Host name="localhost" appBase="webapps"
      unpackWARs="true" autoDeploy="true">
```

#directory 日志目录 /var/log/tomcat/

```
<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
```

#prefix="域名\_access" suffix=".log"

```
    prefix="localhost_access_log" suffix=".txt"
```

```
    pattern="%h %l %u %t "%r" %s %b %D "%{Referer}i" "%{User-Agent}i" %>
```

```
</Host>
```

```

        <Host name="live.oldboylinux.com" appBase="/code/live"
            unpackWARs="true" autoDeploy="true">
#directory 日志目录 /var/log/tomcat/
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
#prefix="域名_access" suffix=".log"
            prefix="live.oldboylinux.com_access" suffix=".log"
            pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;;" />

    </Host>

```

- 配置了2个虚拟主机的tomcat

```

<Engine name="Catalina" defaultHost="localhost">
    <Realm className="org.apache.catalina.realm.LockOutRealm">
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase"/>
    </Realm>

    <Host name="localhost" appBase="webapps"
        unpackWARs="true" autoDeploy="true">
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
            prefix="localhost_access" suffix=".log"
            pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;;" />

    </Host>
    <Host name="live.oldboylinux.com" appBase="/code/live"
        unpackWARs="true" autoDeploy="true">
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
            prefix="live_access" suffix=".log"
            pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}i&quot; &quot;{%User-Agent}i&quot;;" />

    </Host>

```

#重启后检查结果

```

[root@web03 /app/tomcat]# ll /var/log/tomcat/
total 0
-rw-r----- 1 root root 0 Aug  8 12:17 live_access.2021-08-08.log
-rw-r----- 1 root root 0 Aug  8 12:17 localhost_access.2021-08-08.log

```

#配置hosts解析

```

linux    vim /etc/hosts
10.0.0.9 live.oldboylinux.com

```

```

windows : C:\Windows\System32\drivers\etc\hosts
10.0.0.9 live.oldboylinux.com

```

#测试 live站点

```

[root@web03 /app/tomcat]# mkdir /code/live/ROOT
[root@web03 /app/tomcat]# echo live.oldboylinux.com web03 /code/live/ROOT/index.jsp
live.oldboylinux.com web03 /code/live/ROOT/index.jsp
[root@web03 /app/tomcat]# echo live.oldboylinux.com web03 > /code/live/ROOT/index.jsp
[root@web03 /app/tomcat]#
[root@web03 /app/tomcat]# curl live.oldboylinux.com:8080
live.oldboylinux.com web03

```





```
#测试 默认站点
echo this is default virtual host server > /app/tomcat/webapps/ROOT/index.jsp

[root@web03 /app/tomcat]# curl 10.0.0.9:8080

this is default virtual host server

[root@web03 /app/tomcat]# curl -H Host:lidao.oldboylinux.com 10.0.0.9:8080
this is default virtual host server
[root@web03 /app/tomcat]#
```

### 3) 重看-tomcat 处理请求流程

- ① 用户发出http请求报文: Host: 域名 /域名+端口
- ② 用请求到达tomcat, tomcat,connector 端口是否存在,不存在就拒绝,存在就继续处理
- ③ tomcat继续根据用户的请求的域名匹配站点 用户请求的域名Host与Tomcat Host name进行匹配
- ④  tomcat匹配成功,则让用户的请求到达这个站点的对应的站点目录 appBase指定的
- ⑤  tomcat匹配失败,则让用户的请求到达默认的站点进行处理.

## 2.11 Java应用部署方式

### 1) 部署应用方式

- 如果开发给你的是**war包** ==则把war包放入到 tomcat webapps 自动解压 自动部署== 开发给你源代码---maven-->war包??
- 如果开发给你的是**jar包** 相当于jar包里面已经集成了tomcat
  - `java -jar xxxx.jar` 选项

### 2) jar包运行

```
#jar运行演示
java -jar dingding-sonar-1.0-SNAPSHOT.jar --server.port=8082
```

### 3) java开源软件-war包

| 环境准备  |        |
|-------|--------|
| web01 | tomcat |
| db01  | 数据库    |

- 博客:
  - jpress java wordpress
  - **zrlog java blog**
- 功能:
  - jira
- .....

### 3) 部署应用-zrlog

| 准备内容        |  |  |
|-------------|--|--|
| tomcat(jdk) |  |  |
| 数据库         |  |  |
| 应用          |  |  |
| nginx       |  |  |

```
#01 tomcat
<Host name="zrlog.etiantian.org" appBase="/data/blog/"
      unpackWARs="true" autoDeploy="true">
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
        prefix="zrlog_access" suffix=".log"
        pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;{%Referer}%i&quot; &quot;{%User-Agent}%i&quot;"/>

</Host>
```

#### #02 数据库

```
yum install -y mariadb-server
systemctl enable mariadb
systemctl start mariadb
```

#### #初始的配置

```
mysql_secure_installation
```

```
Set root password? [Y/n] Y
New password:
Re-enter new password:
Remove anonymous users? [Y/n] Y
... Success!
Disallow root login remotely? [Y/n] Y
... Success!
```

```
Remove test database and access to it? [Y/n] Y
Reload privilege tables now? [Y/n] Y
... Success!
```

```
Thanks for using MariaDB!
```

#### #进入数据库

#### ##查看数据库

```
show databases;
```

#### ##查看数据库用户信息

```
select user,host from mysql.user;
```

#### ##创建zrlog用的数据库

```
MariaDB [(none)]> create database zrlog charset utf8;
Query OK, 1 row affected (0.00 sec)
```

```
MariaDB [(none)]> show databases;
```

```
+-----+
| Database |
+-----+
| information_schema |
| mysql |
```

```
| performance_schema |
| zrlog               |
+-----+
4 rows in set (0.00 sec)
```

##创建用户 admin 管理所有数据库

```
grant all on *.* to zrlog@'localhost' identified by '1';
grant all on *.* to zrlog@'%' identified by '1';
```

### #03 .部署应用

把代码复制到/code/zrlog

自动解压,部署

### #04. 配置hosts解析

10.0.0.9 zrlog.oldboylinux.com

- 05: 接上面, web页面部署zrlog

- 06: zrlog 连接数据库

← → ↻ ▲ 不安全 | zrlog.oldboylinux.com:8080/zrlog/install

## ZrLog - 安装向导



### 提示

- 安装过程中程序不会创建数据库, 请确保数据库是存在的
- 安装过程将会删除, 清空原有程序的数据和使用数据表

### 填写数据库信息

|         |                                        |
|---------|----------------------------------------|
| 数据库服务器: | <input type="text" value="127.0.0.1"/> |
| 数据库名:   | <input type="text" value="zrlog"/>     |
| 数据库用户名: | <input type="text" value="root"/>      |
| 数据库密码:  | <input type="password"/>               |
| 数据库端口:  | <input type="text" value="3306"/>      |

下一步 →

## ZrLog - 安装向导



### 提示

- 安装过程中程序不会创建数据库，请确保数据库是存在的
- 安装过程将会删除，清空原有程序的数据和使用数据表

### 填写数据库信息

|         |                                        |
|---------|----------------------------------------|
| 数据库服务器: | <input type="text" value="127.0.0.1"/> |
| 数据库名:   | <input type="text" value="zrlog"/>     |
| 数据库用户名: | <input type="text" value="admin"/>     |
| 数据库密码:  | <input type="password" value="....."/> |
| 数据库端口:  | <input type="text" value="3306"/>      |

下一步 →

- 07: 检查 数据库内容(了解) 与用户是上传内容(必会)

```
select * from zrlog.log ;

find /code/zrlog/ -type f -mmin -100
```

#### #部署应用后 tomcat的日志

```
[WARN] 2021-06-02 11:08:46,902 com.zrlog.web.config.ZrLogConfig configPlugin - Not found lock
file(/app/apache-tomcat-8.5.66/webapps/zrlog/WEB-INF/install.lock), Please visit the
http://yourHostName:port/zrlog/install start installation
02-Jun-2021 11:08:47.203 INFO [localhost-startStop-2] org.apache.catalina.startup.HostConfig.deployWAR
Deployment of web application archive [/app/apache-tomcat-8.5.66/webapps/zrlog.war] has finished in [2,682]
ms
```

#### #准备数据库

```
create database zrlog charset utf8;
```

```
MariaDB [(none)]> select user,host from mysql.user;
```

```
+-----+-----+
| user | host |
+-----+-----+
| all  | %    |
| root | %    |
| root | 127.0.0.1 |
| root | ::1  |
| root | db01  |
| root | localhost |
| root | web01  |
+-----+-----+
7 rows in set (0.00 sec)
```

- web页面安装zrlog
- 安装完成后
  - ip:8080/zrlog 首页
  - <http://10.0.0.9:8080/zrlog/admin> 管理页面
- 检查 数据库连接文件 和 用户上传 的目录

```
#数据库连接
[root@web03 /app/tomcat/webapps/zrlog]# cat ./WEB-INF/db.properties
#This is a database configuration file
#Wed Jun 02 11:14:44 CST 2021
driverClass=com.mysql.cj.jdbc.Driver
user=all
password=123456
jdbcUrl=jdbc:mysql://172.16.1.51:3306/zrlog?characterEncoding=UTF-8&allowPublicKeyRetrieval=true&useSSL=false&serverTimezone=GMT

#用户上传
http://10.0.0.9:8080/zrlog/attached/image/20210602/20210602111945_677.png
```

## 4) 部署 jpress应用(作业)

```
#java wordpress jpress
http://www.jpress.io/club/post/116
```

```
#数据库
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| test |
+-----+
4 rows in set (0.00 sec)

#创建数据库 jpress
MariaDB [(none)]> create database jpress charset utf8;
Query OK, 1 row affected (0.00 sec)

#添加用户
MariaDB [(none)]> grant all on jpress.* to 'jpress'@'172.16.1.%' identified by '123456';
Query OK, 0 rows affected (0.00 sec)

#grant 授权 添加用户
grant all on jpress.* to 'jpress'@'172.16.1.%' identified by '123456';
所有权限 在 jpress数据库.中所有表
```

```

172.16.1.7 #精确
172.16.1.% #局域网访问
localhost #本地访问
% #所有
MariaDB [(none)]> grant all on jpress.* to 'jpress'@'172.16.1.%' identified by '123456';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> grant all on jpress.* to 'jpress'@'localhost' identified by '123456';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)

```

#进行测试 web01

```
mysql -ujpress -p123456 -h 172.16.1.61
```

#排坑 删除MySQL中空用户

```
MariaDB [(none)]> select user,host from mysql.user;
```

```

+-----+-----+
| user  | host      |
+-----+-----+
| root  | 127.0.0.1 |
| jpress | 172.16.1.% |
| root  | ::1       |
|       | localhost |
| jpress | localhost |
| root  | localhost |
|       | m01       |
| root  | m01       |
+-----+-----+
8 rows in set (0.00 sec)

```

```

MariaDB [(none)]> drop user ''@'localhost';
Query OK, 0 rows affected (0.00 sec)

```

```

MariaDB [(none)]> drop user ''@'m01';
Query OK, 0 rows affected (0.00 sec)

```

```

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)

```

```
MariaDB [(none)]> select user,host from mysql.user;
```

```

+-----+-----+
| user  | host      |
+-----+-----+
| root  | 127.0.0.1 |
| jpress | 172.16.1.% |
| root  | ::1       |
| jpress | localhost |
| root  | localhost |
| root  | m01       |
+-----+-----+
6 rows in set (0.00 sec)

```

#扩展

```
MariaDB [(none)]> show create database jpress ;
```

```

+-----+-----+
| Database | Create Database |
+-----+-----+
| jpress   | CREATE DATABASE `jpress` /*!40100 DEFAULT CHARACTER SET utf8 */ |
+-----+-----+
1 row in set (0.00 sec)

```

```
MariaDB [(none)]> show create database test ;
```

```

+-----+-----+

```

```
| Database | Create Database |
+-----+-----+
| test | CREATE DATABASE `test` /*!40100 DEFAULT CHARACTER SET latin1 */ |
+-----+-----+
1 row in set (0.00 sec)
```

- MySQL语句 SQL

- 查：

- `show databases;`
- `show tables from db;`
- `select user,host from mysql.user;`

- 增加

- 建库: `create database jpress charset utf8;`
- 添加用户: `grant all on jpress.* to 'jpress'@'172.16.1.%' identified by '123456';`

- 删除

- `drop database db;`
- `drop user jpress@localhost ;`

- 备份

- `mysqldump`

- 安装 jpress 过程



- 手动重启 tomcat



- 登录管理



- web 页面访问

- `10.0.0.7:8080/jpress/` 首页页面
- `10.0.0.7:8080/jpress/admin/` 后台管理页面

- jpress 连接哪里个数据库

```
[root@web01 classes]# cat /app/tomcat/webapps/jpress/WEB-INF/classes/db.properties
#Auto create by JPress
#Tue Dec 17 09:42:09 CST 2019
db_name=jpress
db_host_port=3306
db_tablePrefix=jpress_
db_host=172.16.1.61
db_password=123456
db_user=jpress
```

- wordpress wp-config.php
- 用户上传目录及数据库内容查看

```
[root@web01 ~]# cd /application/tomcat/webapps/
[root@web01 webapps]# ll jpress/attachment/20191217/
total 372
-rw-r--r-- 1 root root 4780 Dec 17 10:14 37b4d16bb21148f6b7a077420d28b735_240x140.jpg
-rw-r--r-- 1 root root 9366 Dec 17 10:14 37b4d16bb21148f6b7a077420d28b735_300x300.jpg
-rw-r--r-- 1 root root 13882 Dec 17 10:14 37b4d16bb21148f6b7a077420d28b735_600x300.jpg
-rw-r--r-- 1 root root 13644 Dec 17 10:14 37b4d16bb21148f6b7a077420d28b735_780x240.jpg
-rw-r--r-- 1 root root 114957 Dec 17 10:14 37b4d16bb21148f6b7a077420d28b735.jpg
-rw-r--r-- 1 root root 9578 Dec 17 10:17 418a1d7bfb9543168f748491bae39ef9_240x140.jpg
-rw-r--r-- 1 root root 18464 Dec 17 10:17 418a1d7bfb9543168f748491bae39ef9_300x300.jpg
-rw-r--r-- 1 root root 32573 Dec 17 10:17 418a1d7bfb9543168f748491bae39ef9_600x300.jpg
-rw-r--r-- 1 root root 31151 Dec 17 10:17 418a1d7bfb9543168f748491bae39ef9_780x240.jpg
-rw-r--r-- 1 root root 108261 Dec 17 10:17 418a1d7bfb9543168f748491bae39ef9.jpg

select * from jpress.jpress_content limit 1 \G
```

## 2.8 Tomcat 3种工作模式: bio , nio , apr

- 面试题: bio , nio

### 1) io模型区别

| 模式  | 英文                      |                                |                                                                     |
|-----|-------------------------|--------------------------------|---------------------------------------------------------------------|
| bio | blocking io             | tomcat 7及之前, <b>同步模型 阻塞</b>    | 一个线程处理一个请求, 缺点: 并发量高时, 线程数较多, 浪费资源。                                 |
| nio | new io                  | tomcat 8及以后的工作模式 <b>异步 非阻塞</b> | nio1(默认的) nio2 <b>可以通过少量的线程处理大量的请求</b>                              |
| apr | Apache Portable Runtime | 应对高并发场景                        | <b>Tomcat对静态文件的处理性能。</b><br><b>Tomcat apr也是在Tomcat上运行高并发应用的首选模式</b> |

### 2) 查看当前使用的io模型

- 查看日志



- 查看管理端



### 3) 修改io模型



- nio

```
protocol="org.apache.coyote.http11.Http11Nio2Protocol"

#server.xml 中 修改 8080
<Connector port="8080" protocol="org.apache.coyote.http11.Http11Nio2Protocol"
           connectionTimeout="20000"
           redirectPort="8443" />

[root@web03 /app/tomcat]# grep -ni nio2 conf/server.xml
69: <Connector port="8080" protocol="org.apache.coyote.http11.Http11Nio2Protocol"
```

- 查看修改结果-catalina.out日志

```
17-Dec-2019 10:46:33.106 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler
["http-nio2-8080"]
```



- 查看修改结果-tomcat管理端

← → ↻ ⚠ 不安全 | 10.0.0.9:8080/manager/status

管理器

应用程序列表

HTML管理器帮助

管理者帮助

服务器信息

| Tomcat.版本            | JVM.版本       | JVM提供商             | OS.名称 | 操作系统版本                 | 操作系 |
|----------------------|--------------|--------------------|-------|------------------------|-----|
| Apache Tomcat/8.5.66 | 1.8.0_60-b27 | Oracle Corporation | Linux | 3.10.0-1160.el7.x86_64 | amc |

JVM

剩余内存: 15.47 MB 总内存: 60.06 MB 最大内存 951.25 MB

| 内存池                    | 类型              | 初始化      | 总共       | 最大值       |
|------------------------|-----------------|----------|----------|-----------|
| Eden Space             | Heap memory     | 16.50 MB | 16.62 MB | 262.50 M  |
| Survivor Space         | Heap memory     | 2.06 MB  | 2.06 MB  | 32.75 M   |
| Tenured Gen            | Heap memory     | 41.37 MB | 41.37 MB | 656.00 M  |
| Code Cache             | Non-heap memory | 2.43 MB  | 7.56 MB  | 240.00 M  |
| Compressed Class Space | Non-heap memory | 0.00 MB  | 3.75 MB  | 1024.00 M |
| Metaspace              | Non-heap memory | 0.00 MB  | 30.75 MB | -0.00 M   |

"http-nio2-8080"

最大线程: 200 当前线程数: 10 当前线程繁忙: 1 存活套接字总数: -1  
最大处理时间: 128 ms 处理时间: 0.191 s 请求总数: 4 错误数: 1 收到字节: 0.00 MB 发送字节: 0.03 MB

| 阶段 | 时间    | 发送字节: | 接收字节 | 客户端 (转发) | 客户端 (实际) | 虚拟主机     |                  |
|----|-------|-------|------|----------|----------|----------|------------------|
| S  | 13 ms | 0 KB  | 0 KB | 10.0.0.1 | 10.0.0.1 | 10.0.0.9 | GET /manager/sta |
| R  | ?     | ?     | ?    | ?        | ?        | ?        |                  |

- apr

#安装apr环境

```
yum -y install apr apr-devel tomcat-native
```

#修改8080&8009端口对应的server.xml

```
protocol="org.apache.coyote.http11.Http11Nio2Protocol"
```

把Nio2 修改为Apr

```
protocol="org.apache.coyote.http11.Http11AprProtocol"
```

浏览器地址: http://10.0.9:8080/manager/status

| 服务器信息                |              |                    |       |                        |        |       |            |
|----------------------|--------------|--------------------|-------|------------------------|--------|-------|------------|
| Tomcat.版本            | JVM.版本       | JVM提供商             | OS.名称 | 操作系统版本                 | 操作系统架构 | 主机名   | IP地址       |
| Apache Tomcat/8.5.66 | 1.8.0_60-b27 | Oracle Corporation | Linux | 3.10.0-1160.el7.x86_64 | amd64  | web03 | 172.16.1.9 |

**OS**

物理内存: 3931.76 MB 可用内存: 2181.91 MB 最大页文件: 3967.99 MB 可用页文件: 3967.99 MB 加载内存: 45  
内核处理时间: 0.56 s 用户态处理实际: 12.77 s

**JVM**

剩余内存: 23.25 MB 总内存: 60.06 MB 最大内存 951.25 MB

| 内存池                    | 类型              | 初始化      | 总共       | 最大值        | 已用            |
|------------------------|-----------------|----------|----------|------------|---------------|
| Eden Space             | Heap memory     | 16.50 MB | 16.62 MB | 262.50 MB  | 6.54 MB (2%)  |
| Survivor Space         | Heap memory     | 2.06 MB  | 2.06 MB  | 32.75 MB   | 2.06 MB (6%)  |
| Tenured Gen            | Heap memory     | 41.37 MB | 41.37 MB | 656.00 MB  | 28.19 MB (4%) |
| Code Cache             | Non-heap memory | 2.43 MB  | 7.56 MB  | 240.00 MB  | 7.28 MB (3%)  |
| Compressed Class Space | Non-heap memory | 0.00 MB  | 3.62 MB  | 1024.00 MB | 3.39 MB (0%)  |
| Metaspace              | Non-heap memory | 0.00 MB  | 30.37 MB | -0.00 MB   | 29.41 MB      |

**"http-apr-8080"**

最大线程: 200 当前线程数: 10 当前线程繁忙: 1 存活连接数总数: 0  
最大处理时间: 0 ms 处理时间: 0.0 s 请求总数: 0 错误数: 0 收到字节: 0.00 MB 发送字节: 0.00 MB

| 阶段 | 时间     | 发送字节 | 接收字节 | 客户端 (转发) | 客户端 (实际) | 虚拟主机     | 请求                           |
|----|--------|------|------|----------|----------|----------|------------------------------|
| S  | 184 ms | 0 KB | 0 KB | 10.0.0.1 | 10.0.0.1 | 10.0.0.9 | GET /manager/status HTTP/1.1 |

P: 解析和准备request S: 服务 F: 结束 R: 就绪 K: 存活

14.4

```
17-Dec-2019 10:59:41.605 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-apr-8080"]
```

```
17-Dec-2019 10:59:41.621 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 3982 ms
```

## 2.8 tomcat 多实例

- 多实例:在同一台服务器上面运行多个tomcat
- 应用场景: 让服务器资源充分利用 使用多实例 E7(铂金CPU) 内存 512G 硬盘 1tb pci-e ssd \* 8 raid 10 nginx
- ⚠️ 注意事项: 端口不同与路径不同.
  - /app/tomcat 8080 8005 /code/zrlog/
  - /app/tomcat\_8081 8080 8006 /code\_8081/zrlog
  - /app/tomcat\_8082 8082 8007 /code\_8082/zrlog

#配置tomcat多实例 tomcat\_8081 tomcat\_8082

```
cp -r /app/tomcat/ /app/tomcat_8081
```

```
cp -r /app/tomcat/ /app/tomcat_8082
```

```
sed -i 's#8080#8081#g' /app/tomcat_8081/conf/server.xml
```

```

sed -i 's#8005#8006#g' /app/tomcat_8081/conf/server.xml

sed -i 's#8080#8082#g' /app/tomcat_8082/conf/server.xml
sed -i 's#8005#8007#g' /app/tomcat_8082/conf/server.xml

[root@oldboy-tomcat ~]# ss -lntup |grep java
tcp LISTEN 0 1 127.0.0.1:8005 0.0.0.0:* users:
(("java",pid=1370,fd=84))
tcp LISTEN 0 100 0.0.0.0:8009 0.0.0.0:* users:
(("java",pid=1370,fd=60))
tcp LISTEN 0 100 0.0.0.0:8080 0.0.0.0:* users:
(("java",pid=1370,fd=55))

#启动tomcat多实例与检查
[root@oldboy-tomcat ~]# /app/tomcat_8081/bin/startup.sh
Using CATALINA_BASE: /app/tomcat_8081
Using CATALINA_HOME: /app/tomcat_8081
Using CATALINA_TMPDIR: /app/tomcat_8081/temp
Using JRE_HOME: /app/jdk
Using CLASSPATH: /app/tomcat_8081/bin/bootstrap.jar:/app/tomcat_8081/bin/tomcat-juli.jar
Tomcat started.
[root@oldboy-tomcat ~]# /app/tomcat_8082/bin/startup.sh
Using CATALINA_BASE: /app/tomcat_8082
Using CATALINA_HOME: /app/tomcat_8082
Using CATALINA_TMPDIR: /app/tomcat_8082/temp
Using JRE_HOME: /app/jdk
Using CLASSPATH: /app/tomcat_8082/bin/bootstrap.jar:/app/tomcat_8082/bin/tomcat-juli.jar
Tomcat started.
[root@oldboy-tomcat ~]# ss -lntup |grep java
tcp LISTEN 0 100 0.0.0.0:8082 0.0.0.0:* users:
(("java",pid=1628,fd=55))
tcp LISTEN 0 1 127.0.0.1:8005 0.0.0.0:* users:
(("java",pid=1370,fd=84))
tcp LISTEN 0 1 127.0.0.1:8006 0.0.0.0:* users:
(("java",pid=1575,fd=81))
tcp LISTEN 0 1 127.0.0.1:8007 0.0.0.0:* users:
(("java",pid=1628,fd=81))
tcp LISTEN 0 100 0.0.0.0:8009 0.0.0.0:* users:
(("java",pid=1370,fd=60))
tcp LISTEN 0 100 0.0.0.0:8010 0.0.0.0:* users:
(("java",pid=1575,fd=60))
tcp LISTEN 0 100 0.0.0.0:8011 0.0.0.0:* users:
(("java",pid=1628,fd=60))
tcp LISTEN 0 100 0.0.0.0:8080 0.0.0.0:* users:
(("java",pid=1370,fd=55))
tcp LISTEN 0 100 0.0.0.0:8081 0.0.0.0:* users:
(("java",pid=1575,fd=55))
[root@oldboy-tomcat ~]#
[root@oldboy-tomcat ~]#
[root@oldboy-tomcat ~]#
[root@oldboy-tomcat ~]#
[root@oldboy-tomcat ~]#
[root@oldboy-tomcat ~]# egrep '808[0-2]|800|8010|8011' /app/tomcat*/conf/server.xml
/app/tomcat_8081/conf/server.xml:<Server port="8006" shutdown="SHUTDOWN">
/app/tomcat_8081/conf/server.xml: Define a non-SSL/TLS HTTP/1.1 Connector on port 8081
/app/tomcat_8081/conf/server.xml: <Connector port="8081" protocol="HTTP/1.1"
/app/tomcat_8081/conf/server.xml: port="8081" protocol="HTTP/1.1"
/app/tomcat_8081/conf/server.xml: <!-- Define an AJP 1.3 Connector on port 8010 -->
/app/tomcat_8081/conf/server.xml: <Connector port="8010" protocol="AJP/1.3" redirectPort="8443" />
/app/tomcat_8082/conf/server.xml:<Server port="8007" shutdown="SHUTDOWN">
/app/tomcat_8082/conf/server.xml: Define a non-SSL/TLS HTTP/1.1 Connector on port 8082
/app/tomcat_8082/conf/server.xml: <Connector port="8082" protocol="HTTP/1.1"
/app/tomcat_8082/conf/server.xml: port="8082" protocol="HTTP/1.1"
/app/tomcat_8082/conf/server.xml: <!-- Define an AJP 1.3 Connector on port 8011 -->
/app/tomcat_8082/conf/server.xml: <Connector port="8011" protocol="AJP/1.3" redirectPort="8443" />
/app/tomcat/conf/server.xml:<Server port="8005" shutdown="SHUTDOWN">
/app/tomcat/conf/server.xml: Define a non-SSL/TLS HTTP/1.1 Connector on port 8080
/app/tomcat/conf/server.xml: <Connector port="8080" protocol="HTTP/1.1"

```

```
/app/tomcat/conf/server.xml:         port="8080" protocol="HTTP/1.1"
/app/tomcat/conf/server.xml: <!-- Define an AJP 1.3 Connector on port 8009 -->
/app/tomcat/conf/server.xml: <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />
[root@oldboy-tomcat ~]#
```

#curl或浏览器访问tomcat多实例

###注意事项

#10.0.0.7:8080/index.jsp 直接访问首页文件 或 展示首页文件

#这个首页文件存放在webapps/ROOT下面 /app//tomcat/webapps/ROOT/index.jsp

#10.0.0.7:8080/oldboy/index.jsp

#/app//tomcat/webapps/oldboy/index.jsp

#tomcat访问首页文件 注意事项

```
[root@web03 /app/tomcat]# echo tomcat_8080 >/app/tomcat/webapps/ROOT/oldboy.jsp
[root@web03 /app/tomcat]# echo tomcat_8081 >/app/tomcat_8081/webapps/ROOT/oldboy.jsp
[root@web03 /app/tomcat]# echo tomcat_8082 >/app/tomcat_8082/webapps/ROOT/oldboy.jsp
[root@web03 /app/tomcat]#
[root@web03 /app/tomcat]# curl 10.0.0.9:8080/oldboy.jsp
tomcat_8080
[root@web03 /app/tomcat]# curl 10.0.0.9:8081/oldboy.jsp
tomcat_8081
[root@web03 /app/tomcat]# curl 10.0.0.9:8082/oldboy.jsp
tomcat_8082
[root@web03 /app/tomcat]#
[root@web03 /app/tomcat]#
[root@web03 /app/tomcat]#
[root@web03 /app/tomcat]# curl 10.0.0.9:808[0-2]/oldboy.jsp
```

```
[1/3]: 10.0.0.9:8080/oldboy.jsp --> <stdout>
--curl--10.0.0.9:8080/oldboy.jsp
tomcat_8080
```

```
[2/3]: 10.0.0.9:8081/oldboy.jsp --> <stdout>
--curl--10.0.0.9:8081/oldboy.jsp
tomcat_8081
```

```
[3/3]: 10.0.0.9:8082/oldboy.jsp --> <stdout>
--curl--10.0.0.9:8082/oldboy.jsp
tomcat_8082
```

```
[root@web03 ~]# diff /app/tomcat/conf/server.xml /app/tomcat_8081/conf/server.xml
22c22
< <Server port="8005" shutdown="SHUTDOWN">
---
> <Server port="8006" shutdown="SHUTDOWN">
69c69
< <Connector port="8080" protocol="org.apache.coyote.http11.Http11Nio2Protocol"
---
> <Connector port="8081" protocol="org.apache.coyote.http11.Http11Nio2Protocol"
141c141
<         prefix="localhost_access" suffix=".log"
---
>         prefix="localhost_access8081" suffix=".log"
145c145
< <Host name="live.oldboylinux.com" appBase="/code/live"
---
> <Host name="live.oldboylinux.com" appBase="/code_8081/live"
148c148
<         prefix="live_access" suffix=".log"
---
>         prefix="live_access8081" suffix=".log"
```

```

152c152
< <Host name="zrlog.oldboylinux.com" appBase="/code/zrlog"
---
> <Host name="zrlog.oldboylinux.com" appBase="/code_8081/zrlog"
155c155
< prefix="zrlog_access" suffix=".log"
---
> prefix="zrlog_access8081" suffix=".log"

[root@web03 ~]# ss -lntup |grep java
tcp LISTEN 0 1 [::ffff:127.0.0.1]:8005 [::]:* users:
(("java",pid=13327,fd=62))
tcp LISTEN 0 1 [::ffff:127.0.0.1]:8006 [::]:* users:
(("java",pid=13369,fd=62))
tcp LISTEN 0 1 [::ffff:127.0.0.1]:8007 [::]:* users:
(("java",pid=13398,fd=62))
tcp LISTEN 0 100 [::]:8080 [::]:* users:
(("java",pid=13327,fd=55))
tcp LISTEN 0 100 [::]:8081 [::]:* users:
(("java",pid=13369,fd=55))
tcp LISTEN 0 100 [::]:8082 [::]:* users:
(("java",pid=13398,fd=55))
[root@web03 ~]# ps -ef|grep '/app/tomcat'
root 13312 10956 0 10:33 pts/1 00:00:00 tail -f /app/tomcat/logs/catalina.out
root 13327 1 7 10:33 ? 00:00:03 /app/jdk/bin/java -
Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHkeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -classpath
/app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat -
Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap start
root 13369 1 9 10:33 ? 00:00:03 /app/jdk/bin/java -
Djava.util.logging.config.file=/app/tomcat_8081/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHkeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -classpath
/app/tomcat_8081/bin/bootstrap.jar:/app/tomcat_8081/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat_8081 -
Dcatalina.home=/app/tomcat_8081 -Djava.io.tmpdir=/app/tomcat_8081/temp
org.apache.catalina.startup.Bootstrap start
root 13398 1 9 10:33 ? 00:00:03 /app/jdk/bin/java -
Djava.util.logging.config.file=/app/tomcat_8082/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHkeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -classpath
/app/tomcat_8082/bin/bootstrap.jar:/app/tomcat_8082/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat_8082 -
Dcatalina.home=/app/tomcat_8082 -Djava.io.tmpdir=/app/tomcat_8082/temp
org.apache.catalina.startup.Bootstrap start
root 13443 11001 0 10:34 pts/0 00:00:00 grep --color=auto /app/tomcat

[root@web03 ~]# echo 'tomcat_8080' >/app/tomcat/webapps/ROOT/index.jsp
[root@web03 ~]# echo 'tomcat_8081' >/app/tomcat_8081/webapps/ROOT/index.jsp
[root@web03 ~]# echo 'tomcat_8082' >/app/tomcat_8082/webapps/ROOT/index.jsp
[root@web03 ~]#
[root@web03 ~]#
[root@web03 ~]#
curl 10.0.0.9:8080
curl 10.0.0.9:8081
curl 10.0.0.9:8082

```

 image-20200210111157645

• 补充注意事项:

- 1. 10.0.0.9:8080/oldboy/lidao.jsp --- webapps/oldboy/lidao.jsp
- 2. 10.0.0.9:8080/lidao.jsp --- webapps/ROOT/lidao.jsp

- 多实例和多虚拟主机怎么选择?
- 兼顾性能,兼顾冗余.
- 多实例,tomcat
- 多虚拟主机,nginx

## 2.10 Tomcat 监控功能

- Tomcat远程监控(remote ) 应用:
  - 开发: 给开发留个后门 ,开发/运维 通过门查看 tomcat(jvm)
  - 运维: 通过命令检查tomcat信息 ,通过监控软件检查获取tomcat信息
- 通过监控软件 可以监控 tomcat状态(jvm状态)
- 监控方法:
  - 1. 通过命令/脚本查看
  - 2. 开启tomcat监控功能 ,jmx再让zabbix监控

### 2.10.1 命令

- 通过 命令行命令检查 tomcat状态:
- jps
- jmap
- jstack
- ps 与 jps -lvm
- pstree -a /pstree -p
- show-busy-java-threads.sh 显示java程序,使用率最高的线程.
- 使用流程
  - top/htop/ps 检查出什么进程有问题
  - 根据这个进程pid,通过jmap(导出jvm信息) jstack 显示线程信息
  - 与开发一起排查.
- 使用脚本排错流程
  - 执行脚本 获取到java进程信息(process id) java 线程信息(thread id )
  - jstack pid |grep -A5 16进制的thread id

```
# jps java ps
#显示java进程信息
[root@oldboy-tomcat ~]# jps -l
1575 org.apache.catalina.startup.Bootstrap
1370 org.apache.catalina.startup.Bootstrap
1628 org.apache.catalina.startup.Bootstrap
1759 sun.tools.jps.Jps
[root@oldboy-tomcat ~]# jps -lv
```

```

1575 org.apache.catalina.startup.Bootstrap -
Djava.util.logging.config.file=/app/tomcat_8081/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat_8081 -Dcatalina.home=/app/tomcat_8081 -Djava.io.tmpdir=/app/tomcat_8081/temp
1769 sun.tools.jps.Jps -Denv.class.path=.:/app/jdk/lib:/app/jdk/jre/lib:/app/jdk/lib/tools.jar -
Dapplication.home=/app/jdk1.8.0_241 -Xms8m
1370 org.apache.catalina.startup.Bootstrap -
Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat -Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp
1628 org.apache.catalina.startup.Bootstrap -
Djava.util.logging.config.file=/app/tomcat_8082/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat_8082 -Dcatalina.home=/app/tomcat_8082 -Djava.io.tmpdir=/app/tomcat_8082/temp
[root@oldboy-tomcat ~]# jps -lvm
1779 sun.tools.jps.Jps -lvm -Denv.class.path=.:/app/jdk/lib:/app/jdk/jre/lib:/app/jdk/lib/tools.jar -
Dapplication.home=/app/jdk1.8.0_241 -Xms8m
1575 org.apache.catalina.startup.Bootstrap start -
Djava.util.logging.config.file=/app/tomcat_8081/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat_8081 -Dcatalina.home=/app/tomcat_8081 -Djava.io.tmpdir=/app/tomcat_8081/temp
1370 org.apache.catalina.startup.Bootstrap start -
Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat -Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp
1628 org.apache.catalina.startup.Bootstrap start -
Djava.util.logging.config.file=/app/tomcat_8082/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat_8082 -Dcatalina.home=/app/tomcat_8082 -Djava.io.tmpdir=/app/tomcat_8082/temp

```

#通过 jps 或 ps 找出 java进程及pid pid随机 每次重启 tomcat pid

#脚本 show-busy-java-threads.sh

##根据java线程繁忙状态排序

##显示cpu使用率最高的 前几个 java线程

```
[root@web03 ~]# sh show-busy-java-threads.sh
```

```
[1] Busy(0.4%) thread(10562/0x2942) stack of java process(10551) under user(root):
```

process 进程

thread 线程

线程 进程

进程 在内存中创建空间

线程 实际处理用户请求

pstree

```

├─java(10522)─┬─{java}(10523)
│              └─{java}(10524)
│              └─{java}(10525)
│              └─{java}(10526)
│              └─{java}(10527)
│              └─{java}(10528)
│              └─{java}(10529)

```

```

|           |─{java}(10530)
|           |─{java}(10531)
|           |─{java}(10532)
|           |─{java}(10533)
|           |─{java}(10534)
|           |─{java}(10535)

```

```
[root@web01 ~]# sh show-busy-java-threads.sh
```

```
[1] Busy(14.8%) thread(10528/0x2920) stack of java process(10522) under user(root):
```

```
"C2 CompilerThread0" #5 daemon prio=9 os_prio=0 tid=0x00007fe9700ae000 nid=0x2920 waiting on condition
[0x0000000000000000]
```

```
java.lang.Thread.State: RUNNABLE
```

```
[2] Busy(2.0%) thread(10529/0x2921) stack of java process(10522) under user(root):
```

```
"C1 CompilerThread1" #6 daemon prio=9 os_prio=0 tid=0x00007fe9700b0800 nid=0x2921 waiting on condition
[0x0000000000000000]
```

```
java.lang.Thread.State: RUNNABLE
```

```
[3] Busy(2.0%) thread(10523/0x291b) stack of java process(10522) under user(root):
```

```
"main" #1 prio=5 os_prio=0 tid=0x00007fe970009000 nid=0x291b runnable [0x00007fe977a0d000]
```

```
java.lang.Thread.State: RUNNABLE
```

```

at java.net.PlainSocketImpl.socketAccept(Native Method)
at java.net.AbstractPlainSocketImpl.accept(AbstractPlainSocketImpl.java:409)
at java.net.ServerSocket.implAccept(ServerSocket.java:545)
at java.net.ServerSocket.accept(ServerSocket.java:513)
at org.apache.catalina.core.StandardServer.await(StandardServer.java:446)
at org.apache.catalina.startup.Catalina.await(Catalina.java:713)
at org.apache.catalina.startup.Catalina.start(Catalina.java:659)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:497)
at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:351)
at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:485)

```

```
[4] Busy(0.7%) thread(10524/0x291c) stack of java process(10522) under user(root):
```

```
"VM Thread" os_prio=0 tid=0x00007fe97006d800 nid=0x291c runnable
```

```
[5] Busy(0.0%) thread(10565/0x2945) stack of java process(10522) under user(root):
```

```
"ajp-apr-8009-AsyncTimeout" #42 daemon prio=5 os_prio=0 tid=0x00007fe97051e800 nid=0x2945 waiting on
condition [0x00007fe937ffe000]
```

```
java.lang.Thread.State: TIMED_WAITING (sleeping)
```

```

at java.lang.Thread.sleep(Native Method)
at org.apache.tomcat.util.net.AbstractEndpoint$AsyncTimeout.run(AbstractEndpoint.java:129)
at java.lang.Thread.run(Thread.java:745)

```

线程是指进程内的一个执行单元，

**#进程**

进程拥有自己独立的堆和栈，既不共享堆，亦不共享栈，进程由操作系统调度。

**#线程**

线程拥有自己独立的栈和共享的堆，共享堆，不共享栈，线程亦由操作系统调度

**#协程和线程**

协程避免了无意义的调度，由此可以提高性能；但同时协程也失去了线程使用多CPU的能力

进程与线程的区别

(1)地址空间：线程是进程内的一个执行单位，进程内至少有一个线程，他们共享进程的地址空间，而进程有自己独立的地址空间

(2)资源拥有：进程是资源分配和拥有的单位，同一个进程内线程共享进程的资源

(3)线程是处理器调度的基本单位，但进程不是

(4)二者均可并发执行

(5)每个独立的线程有一个程序运行的入口

## 2.10.2 开启tomcat监控功能

- 正常应用:开启tomcat监控功能后,通过zabbix 监控
- 安装windows版本,jdk替代zabbix监控



- 使用流程
  - 1 tomcat开启监控功能
  - 2 windows下面安装jdk并检查jdk是否正常
  - 3 windows下面通过jdk软件,连接tomcat

- 1 然后 linux下面进行配置

```
#开启tomcat远程监控 功能
#修改 tomcat/bin/catalina.sh

#CATALINA_OPTS java内置变量 修改java启动参数(tomcat)
# CATALINA_OPTS  jav

CATALINA_OPTS="$CATALINA_OPTS
-Dcom.sun.management.jmxremote           #jmx remote  开启tomcat远程监控功能
-Dcom.sun.management.jmxremote.port=12345  #指定端口 12345  还有2个随机端口
-Dcom.sun.management.jmxremote.authenticate=false #auth 认证 是否开启远程监控认证(用户名 密码)
-Dcom.sun.management.jmxremote.ssl=false      #是否开启https
-Djava.rmi.server.hostname=10.0.0.9"         #tomcat监听的ip地址
                                              #这里我们书写公网ip地址
                                              #生产环境 书写内网ip
                                              #本地ip 10.0.0.7 172.16.1.7

[root@web03 ~]# grep -A5 '^CATALINA_OPT' /app/tomcat/bin/catalina.sh
CATALINA_OPTS="$CATALINA_OPTS
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=12345
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-Djava.rmi.server.hostname=10.0.0.9"

[root@web01 ~]# cd /application/tomcat/bin/
[root@web01 bin]# vim catalina.sh

CATALINA_OPTS="$CATALINA_OPTS \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=12345 \
-Dcom.sun.management.jmxremote.authenticate=false \
-Dcom.sun.management.jmxremote.ssl=false \
-Djava.rmi.server.hostname=10.0.0.9"

#注意：

##如果是 8.5 版本
需要参数写在1行 或 使用续行符号
CATALINA_OPTS="$CATALINA_OPTS \
-Dcom.sun.management.jmxremote \
-Dcom.sun.management.jmxremote.port=12345 \
-Dcom.sun.management.jmxremote.authenticate=false \
-Dcom.sun.management.jmxremote.ssl=false \
-Djava.rmi.server.hostname=47.114.128.48"

#注意：
[root@oldboy-tomcat ~]# #jmxremote tomcat远程监控 有1个固定端口+2个随机端口
```

#开启 远程监控功能之前

```
/app/jdk/bin/java -Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dignore.endorsed.dirs= -classpath
/app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat -
Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap start
```

#开启 远程监控功能

```
/app/jdk/bin/java -Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHKeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dcom.sun.management.jmxremote -
Dcom.sun.management.jmxremote.port=12345 -Dcom.sun.management.jmxremote.authenticate=false-
Dcom.sun.management.jmxremote.ssl=false -Djava.rmi.server.hostname=10.0.0.9 -Dignore.endorsed.dirs= -
classpath /app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat -
Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap start
```

#catalina.sh

```
/app/jdk/bin/java
-Djava.util.logging.config.file=/app/tomcat/conf/logging.properties
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
-Djdk.tls.ephemeralDHKeySize=2048 -Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=12345
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-Djava.rmi.server.hostname=10.0.0.9
-Dignore.endorsed.dirs=
-classpath /app/tomcat/bin/bootstrap.jar:/app/tomcat/bin/tomcat-juli.jar -Dcatalina.base=/app/tomcat
-Dcatalina.home=/app/tomcat
-Djava.io.tmpdir=/app/tomcat/temp org.apache.catalina.startup.Bootstrap

start
```

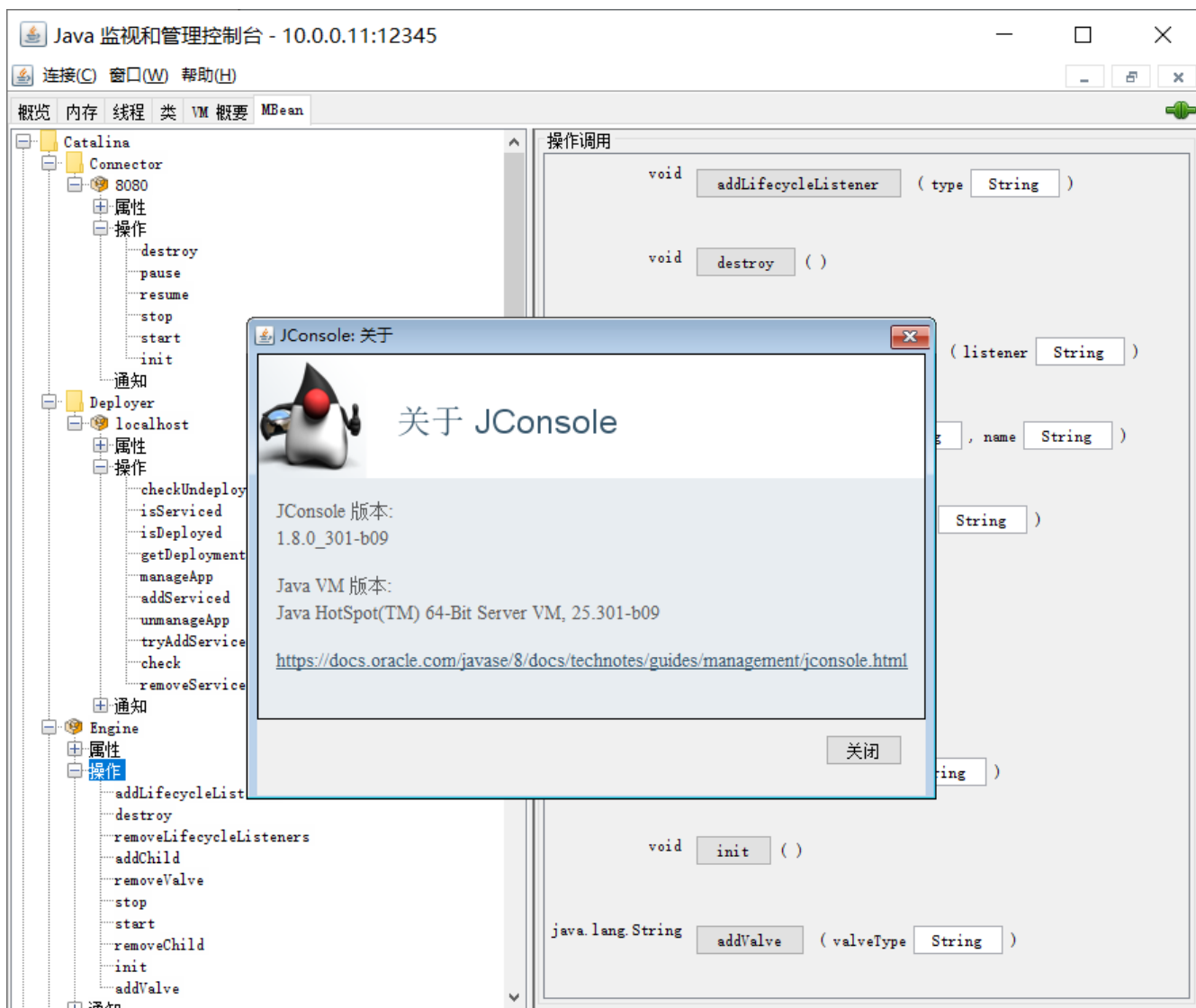
- [2]通过windows jconsole 连接(模拟zabbix连接) linux tomcat

```
C:\Users\91839>

C:\Users\91839>java -version
java version "1.8.0_31"
Java(TM) SE Runtime Environment (build 1.8.0_31-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.31-b07, mixed mode)

C:\Users\91839>
```

jconsole.exe #C:\Program Files\Java\jdk1.8.0\_31\bin\jconsole.exe  
jvisualvm.exe #C:\Program Files\Java\jdk1.8.0\_31\bin\jvisualvm.exe





JConsole: 新建连接



新建连接

☐ 本地进程 (L) :

| 名称                          | PID   |
|-----------------------------|-------|
| sun.tools.jconsole.JConsole | 16128 |

☐ 远程进程 (R) :

用法: <hostname>:<port> 或 service:jmx:<protocol>:<sap>

用户名 (U) :  口令 (P) :

连接 (C)

取消

新建连接

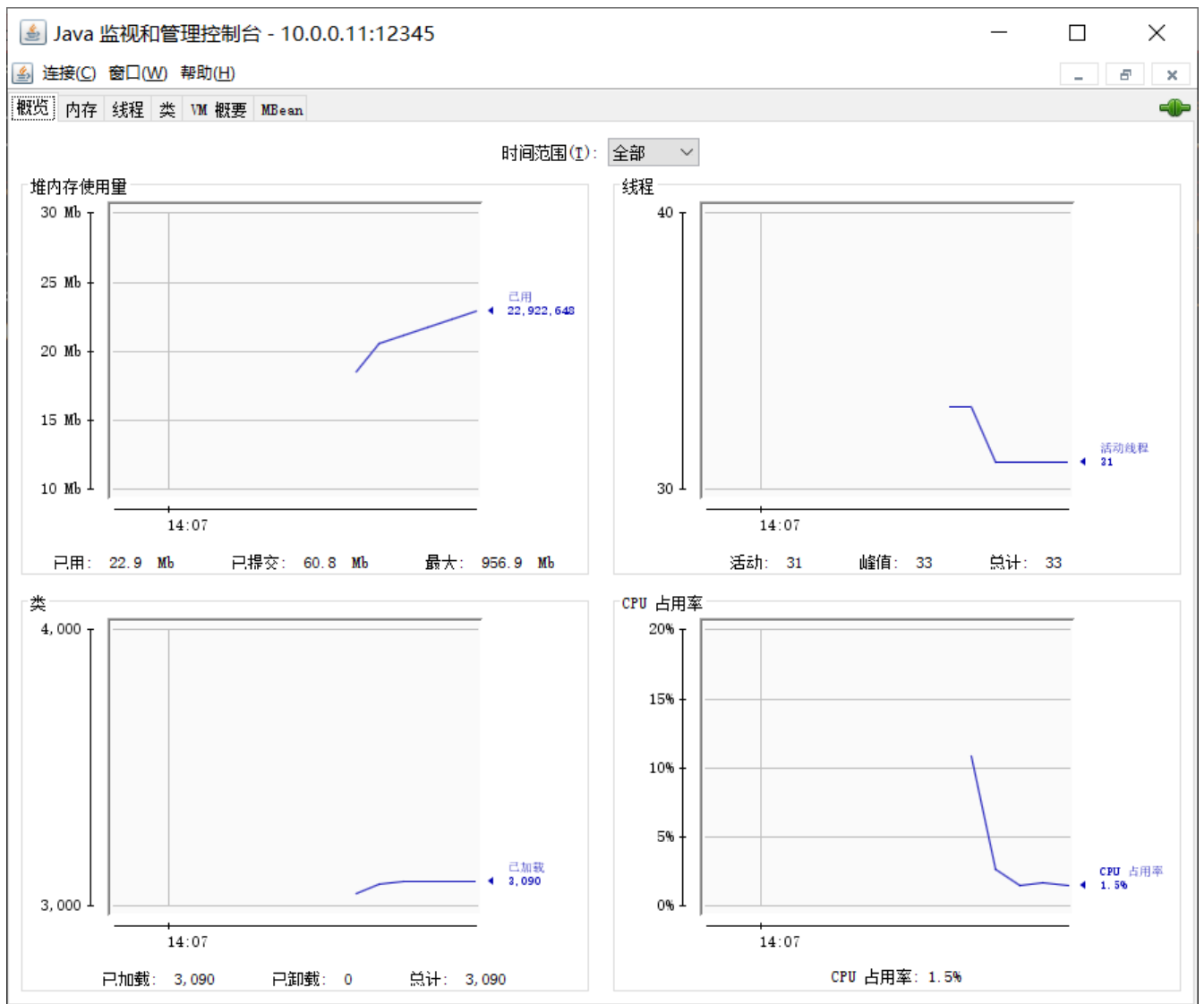
☐ 本地进程(L) :

| 名称                          | PID   |
|-----------------------------|-------|
| sun.tools.jconsole.JConsole | 16128 |

☒ 远程进程(R) :

用法: <hostname>:<port> 或 service:imx:<protocol>:<sap>

用户名(U) :  口令(P) :



- jxxxxlvm

添加远程主机

主机名(H): 10.0.0.11

☐ 显示名称(D): 10.0.0.11

高级设置(A) 确定 取消

 添加 JMX 连接

连接 (C) :

10.0.0.11:12345

用法: <主机名>:<端口> 或 service:jmx:<协议>:<sa

☐ 显示名称 (D) :

10.0.0.11:12345

☐ 使用安全凭证 (E)

用户名 (U) :

口令 (P) :

☐ 保存安全凭证 (S)

☐ 不要求SSL连接 (N)

确定

取消

### 2.10.3 tomcat远程监控功能小结

- 命令/脚本: jps -lvm / pstree / show\_busy\_java\_threads.sh
- 远程监控功能: jmxremote / jmx
  - 开发(排错)或运维使用(zabbix)
- 小坑: 在 tomcat 8.5.x 配置开启功能 修改 catalina.sh 写成一行 或加上 \ (续行)

## 2.11 Tomcat相关故障及排错

- 故障案例01: tomcat 开机自启动故障/定时任务

#背景:

##1. tomcat环境使用的jdk 是二进制

##2. 设置tomcat开机自启动

#使用二进制安装jdk的时候 环境变量不在 /bin /sbin /usr/local/bin /usr/local/sbin

#在重启就没有找到这个命令

#即使在 /etc/profile 中也是 无法识别

需要在 /etc/rc.local

. /etc/profile

##/etc/rc.local

. /etc/profile

/app/tomcat/bin/startup.sh

#二进制方式安装jdk , 手动配置环境变量 /etc/profile中

- 故障案例02: Tomcat(java) 服务器,运行占用大量swap,物理内存占用较少
- 运行着java程序服务器,发现服务的swap被占用,物理内存还有剩余.

```
#java代码问题
## 增加swap

##Linux内核参数:
/etc/sysctl.conf
vm.swappiness=0
sysctl -p #内核参数生效

vm.swappiness=0      #控制系统是否优先使用物理内存 数越小 越优先使用物理内存
swap 亲和力.
```

- 故障案例03::tomcat负载高 15k-20k
  - 排查流程
    - 1. 定位进程: 整体排查: vmstat /top/ps aux 找出哪个进程的问题
    - 2. 找出进程对应线程id :
      - 1. 通过top -Hp java进程id 找出是哪个java线程的问题
      - 2. 或者 show\_busy\_java\_threads.sh
    - 3. 转换:线程id-->16进制:问题线程的id 转换为16进制 线程id
    - 4. 找出线程的详细信息: jstack (显示java 进程信息) jstack java进程id 过滤 java线程的16进制id 与开发沟通
    - 5. 显示jvm信息: jmap (显示java jvm信息) jmap -heap java进程id 显示jvm的内存使用情况
    - 6. jvm内存内容导出:jmap (导出 jvm内存的内容) jmap -dump:format=b,file=/root/tomcat.bin pid
    - 7. 给开发分析jvm导出文件:通过 mat(Eclipse Memory Analyzer Tool )分析 windows
  - 基础排查版本使用流程

```
#01. 找出问题java进程
[root@web03 ~]# #top -Hp 1425
[root@web03 ~]# #1425 进程id

#02通过进程 找出问题线程
[root@web03 ~]# #1459 线程id

#03. 线程id转换为16进制

[root@web03 ~]# echo 'obase=16;1459' |bc
5B3

#04. 通过jstack pid 过滤 java线程id(16进制)信息

[root@web03 ~]# jstack 1425 |grep -i '5B3'
"main" #1 prio=5 os_prio=0 tid=0x00007f733c009800 nid=0x5b3 runnable [0x00007f734223c000]
[root@web03 ~]# jstack 1425 |grep -A5 -i '5B3'
"main" #1 prio=5 os_prio=0 tid=0x00007f733c009800 nid=0x5b3 runnable [0x00007f734223c000]
java.lang.Thread.State: RUNNABLE
at java.net.PlainSocketImpl.socketAccept(Native Method)
at java.net.AbstractPlainSocketImpl.accept(AbstractPlainSocketImpl.java:409)
at java.net.ServerSocket.implAccept(ServerSocket.java:545)
at java.net.ServerSocket.accept(ServerSocket.java:513)
```



- 进阶排查版本使用流程

#### #05. 把jvm内存使用情况 导出

```
[root@web03 ~]# jmap -heap 1425 #java进程pid
```

```
Attaching to process ID 1425, please wait...
```

```
Debugger attached successfully.
```

```
Server compiler detected.
```

```
JVM version is 25.60-b23
```

```
using thread-local object allocation.
```

```
Mark Sweep Compact GC
```

##### Heap Configuration:

```
MinHeapFreeRatio      = 40
MaxHeapFreeRatio      = 70
MaxHeapSize           = 1031798784 (984.0MB)
NewSize               = 21626880 (20.625MB)
MaxNewSize            = 343932928 (328.0MB)
OldSize               = 43384832 (41.375MB)
NewRatio              = 2
SurvivorRatio         = 8
MetaspaceSize         = 21807104 (20.796875MB)
CompressedClassSpaceSize = 1073741824 (1024.0MB)
MaxMetaspaceSize      = 17592186044415 MB
G1HeapRegionSize      = 0 (0.0MB)
```

##### Heap Usage:

###### New Generation (Eden + 1 Survivor Space):

```
capacity = 19464192 (18.5625MB)
used     = 7170328 (6.838157653808594MB)
free     = 12293864 (11.724342346191406MB)
36.838559751157405% used
```

###### Eden Space:

```
capacity = 17301504 (16.5MB)
used     = 5007648 (4.775665283203125MB)
free     = 12293856 (11.724334716796875MB)
28.943425958806817% used
```

###### From Space:

```
capacity = 2162688 (2.0625MB)
used     = 2162680 (2.0624923706054688MB)
free     = 8 (7.62939453125E-6MB)
99.99963008996212% used
```

###### To Space:

```
capacity = 2162688 (2.0625MB)
used     = 0 (0.0MB)
free     = 2162688 (2.0625MB)
0.0% used
```

###### tenured generation:

```
capacity = 43384832 (41.375MB)
used     = 11550936 (11.015830993652344MB)
free     = 31833896 (30.359169006347656MB)
26.624364939341014% used
```

```
15973 interned Strings occupying 1488288 bytes.
```

#### #06. 把jvm内存信息导出文件

```
jmap -dump:format=b,file=/root/tomcat.bin pid(java进程)
```

```
[root@web03 ~]# jmap -dump:format=b,file=/tmp/java-dmp.bin 1425
```

```
Dumping heap to /tmp/java-dmp.bin ...
```

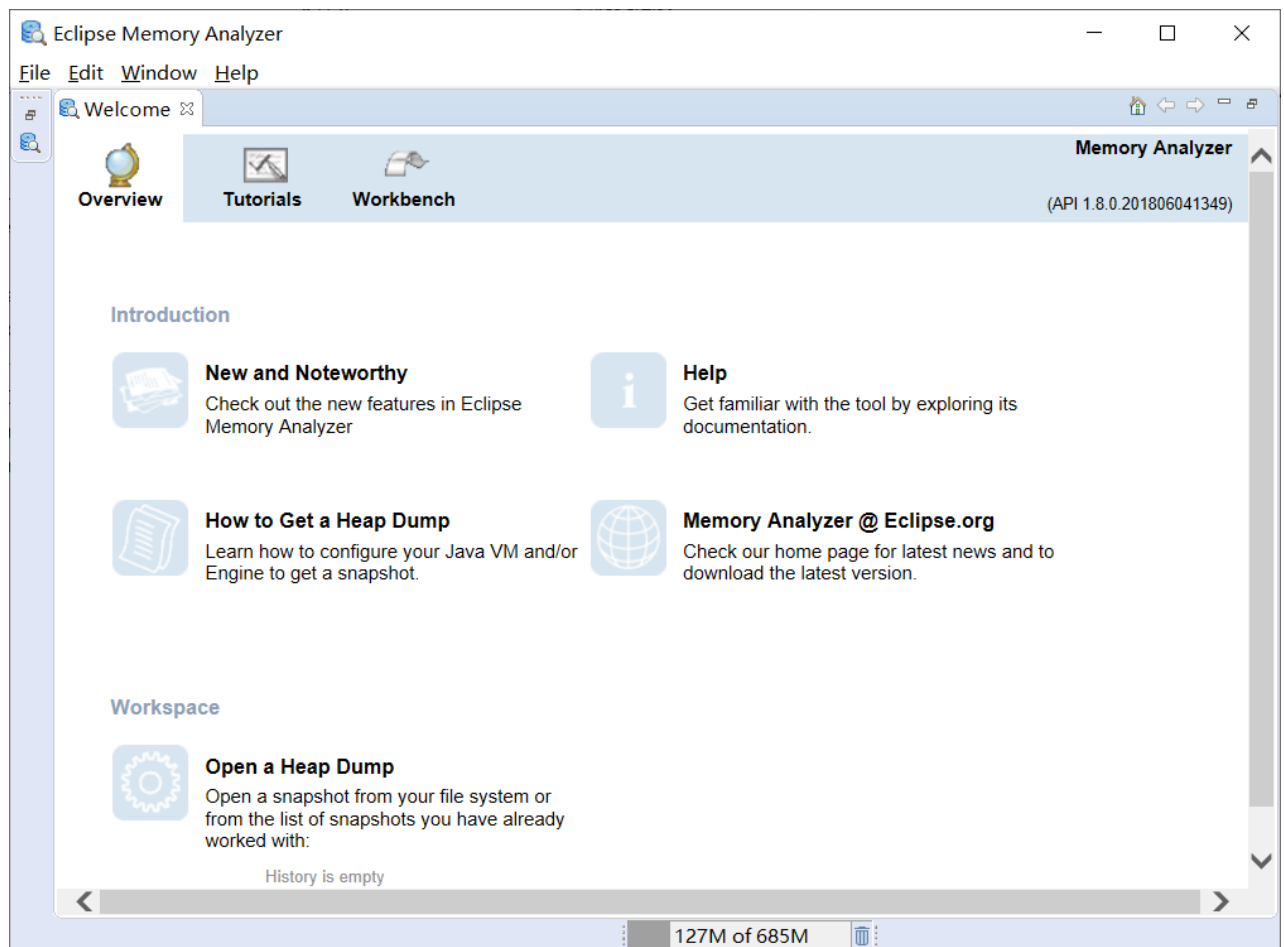
```
Heap dump file created
```

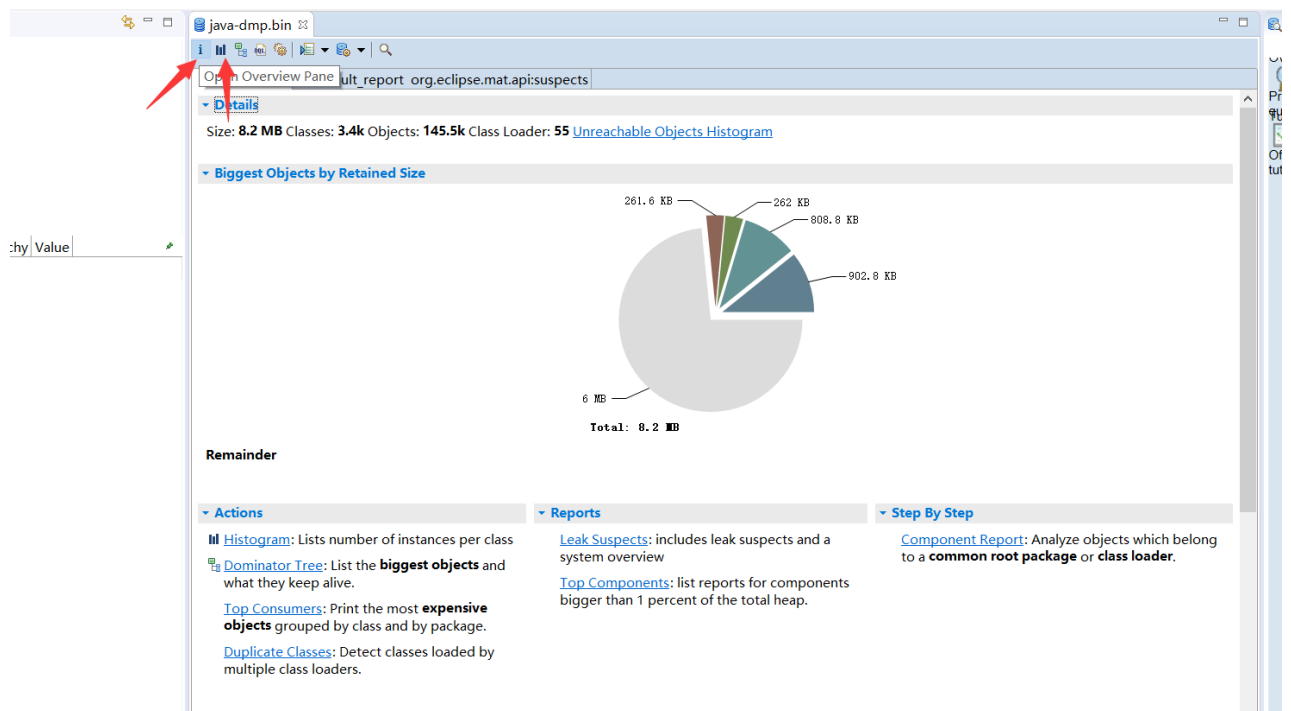
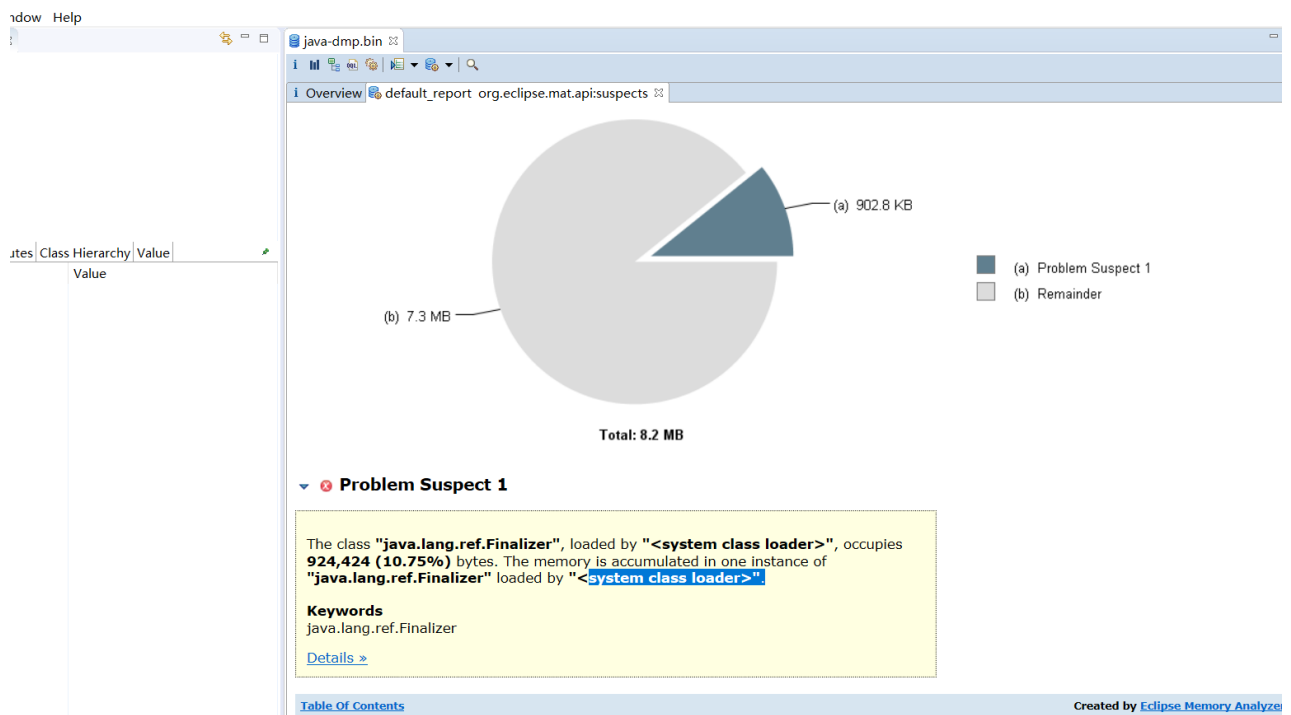
```
[root@web03 ~]# ll -h /tmp/java-dmp.bin
-rw----- 1 root root 26M Aug 15 14:46 /tmp/java-dmp.bin
[root@web03 ~]# file /tmp/java-dmp.bin
/tmp/java-dmp.bin: data
```

#07. 下载到windows 通过mat软件分析

mat eclipse Memory Analyzer (MAT)

MemoryAnalyzer.exe





- vmstat /top/ps aux
- tomcat: 找出这个tomcat的pid top -Hp 10789 精确的找出某个线程导致的
- 并且能找出线程的id 10802 (10进制)转换为16进制

```
[root@web01 jpress]# echo 'obase=16;10802' | bc
2A32
```

- jstack pid (java进程的) 然后过滤 线程16进制的id 与开发一起查看

```
[root@web01 jpress]# jstack 10789 |grep -i -C5 '2A32'
    at org.apache.catalina.startup.Bootstrap.start(Bootstrap.java:351)
    at org.apache.catalina.startup.Bootstrap.main(Bootstrap.java:485)
"VM Thread" os_prio=0 tid=0x00007fe53006d800 nid=0x2a27 runnable
"VM Periodic Task Thread" os_prio=0 tid=0x00007fe53018c000 nid=0x2a32 waiting on condition
JNI global references: 263
```

- 详细步骤

## 通用步骤

##找出 cpu或 磁盘 导致 负载高

#r 数字较大 意味着系统的 cpu使用率较高

#b 数字较大 意味着 磁盘io较高

[root@web01 jpress]# vmstat 1 10

```
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
r  b  swpd  free  buff  cache  si  so  bi  bo  in  cs  us  sy  id  wa  st
3  0   3592  92948    0  533296  0  0   27   23  91  194  0  0  99  0  0
0  0   3592  92948    0  533296  0  0    0    0  76  136  0  0 100  0  0
0  0   3592  92948    0  533296  0  0    0   208  99  151  0  2  98  0  0
0  0   3592  92948    0  533296  0  0    0    0  72  130  0  0 100  0  0
0  0   3592  92948    0  533296  0  0    0    0  83  146  1  0  99  0  0
0  0   3592  92948    0  533296  0  0    0    0  72  129  0  0 100  0  0
0  0   3592  92948    0  533296  0  0    0    0  80  141  0  0 100  0  0
0  0   3592  92948    0  533296  0  0    0    0  99  177  0  0 100  0  0
0  0   3592  92948    0  533296  0  0    0    0  88  153  0  1  99  0  0
0  0   3592  92948    0  533296  0  0    0    0  74  133  0  0 100  0  0
```

##具体哪个进程导致的 pid

###显示 cpu和内存的使用率 每个进程

ps aux 或 top 或 htop

###显示每个进程的 io情况 pid

iotop -o #-o 只显示在进行读写的进程

# tomcat php mysql

##通过top -Hp java 进程id 找出是哪个java线程的问题

找出 进程中 哪个线程占用 cpu 内存

-H 显示进程对应线程信息

-p 指定pid

[root@oldboy-tomcat ~]# top -Hp 997

```
top - 15:54:49 up 36 min, 2 users, load average: 0.00, 0.01, 0.00
Threads: 66 total, 0 running, 66 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.3 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 1726.7 total, 834.4 free, 390.2 used, 502.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 1185.4 avail Mem
```

| PID  | USER | PR | NI | VIRT    | RES    | SHR   | S | %CPU | %MEM | TIME+   | COMMAND |
|------|------|----|----|---------|--------|-------|---|------|------|---------|---------|
| 997  | root | 20 | 0  | 2890228 | 131564 | 15308 | S | 0.0  | 7.4  | 0:00.00 | java    |
| 1018 | root | 20 | 0  | 2890228 | 131564 | 15308 | S | 0.0  | 7.4  | 0:00.55 | java    |

.....下面省略

# 问题线程的id 转换为16进制

1018 ---> 16进制

[root@oldboy-tomcat ~]# echo 'obase=16;10' |bc

A

[root@oldboy-tomcat ~]# echo 'obase=16;15' |bc

F

[root@oldboy-tomcat ~]# echo 'obase=16;1018' |bc

3FA

## tomcat

#jstack 导出java 进程 线程信息

jstack (显示java 进程信息) jstack java<font color=red>\*\*进程id\*\*</font> 过滤<font color=red> java线程</font>的16进制id 与开发沟通

#导出 java 进程信息 配合开发人员一起排查  
#jstack 997 |grep -i 3FA  
[root@oldboy-tomcat ~]# jstack 997

```
[root@oldboy-tomcat ~]# jstack 997|grep -i 3fa -A 10
"main" #1 prio=5 os_prio=0 tid=0x00007fd8c400b800 nid=0x3fa runnable [0x00007fd8c9ffe000]
java.lang.Thread.State: RUNNABLE
    at java.net.PlainSocketImpl.socketAccept(Native Method)
    at java.net.AbstractPlainSocketImpl.accept(AbstractPlainSocketImpl.java:409)
    at java.net.ServerSocket.implAccept(ServerSocket.java:545)
    at java.net.ServerSocket.accept(ServerSocket.java:513)
    at org.apache.catalina.core.StandardServer.await(StandardServer.java:447)
    at org.apache.catalina.startup.Catalina.await(Catalina.java:776)
    at org.apache.catalina.startup.Catalina.start(Catalina.java:722)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
```

- 与jvm相关的 jmap 导出或显示jvm信息
  - 显示: 显示jvm使用情况 一共多少 使用多少 剩余多少
  - 导出: jvm内存信息 数据 等等导出到磁盘
- jmap -dump 导出jvm内存内容



#5. \*\*显示jvm信息\*\*: jmap (显示java jvm信息) jmap -heap java<font color=red>\*\*进程id\*\*</font> 显示jvm的内存使用情况

jmap -heap javapid

```
[root@oldboy-tomcat ~]# jmap -heap 997 #java jvm使用率 后面用来做自定义监控
Attaching to process ID 997, please wait...
Debugger attached successfully.
Server compiler detected.
JVM version is 25.241-b07
```

using thread-local object allocation.  
Mark Sweep Compact GC

#### Heap Configuration:

|                          |                          |
|--------------------------|--------------------------|
| MinHeapFreeRatio         | = 40                     |
| MaxHeapFreeRatio         | = 70                     |
| MaxHeapSize              | = 452984832 (432.0MB)    |
| NewSize                  | = 9764864 (9.3125MB)     |
| MaxNewSize               | = 150994944 (144.0MB)    |
| OldSize                  | = 19595264 (18.6875MB)   |
| NewRatio                 | = 2                      |
| SurvivorRatio            | = 8                      |
| MetaspaceSize            | = 21807104 (20.796875MB) |
| CompressedClassspaceSize | = 1073741824 (1024.0MB)  |
| MaxMetaspaceSize         | = 17592186044415 MB      |
| GLHeapRegionSize         | = 0 (0.0MB)              |

#### Heap Usage:

New Generation (Eden + 1 Survivor Space):

```
capacity = 8847360 (8.4375MB)
used      = 3521584 (3.3584442138671875MB)
free      = 5325776 (5.0790557861328125MB)
39.80378327546296% used
Eden Space:
capacity = 7864320 (7.5MB)
used      = 3253128 (3.1024246215820312MB)
free      = 4611192 (4.397575378417969MB)
41.36566162109375% used
From Space:
capacity = 983040 (0.9375MB)
used      = 268456 (0.25601959228515625MB)
free      = 714584 (0.6814804077148438MB)
27.308756510416668% used
To Space:
capacity = 983040 (0.9375MB)
used      = 0 (0.0MB)
free      = 983040 (0.9375MB)
0.0% used
tenured generation:
capacity = 19595264 (18.6875MB)
used      = 19396856 (18.49828338623047MB)
free      = 198408 (0.18921661376953125MB)
98.98746962531355% used
```

20471 interned Strings occupying 1861416 bytes.

#6. \*\*jvm内存内容导出\*\*:jmap (导出 jvm内存的内容) `jmap -dump:format=b,file=/root/tomcat.bin pid`

jmap -dump:format=b,file=/root/tomcat.bin javapid

#下载到windows,准备 通过java mat分析

#7. 通过 java mat 分析

- 7. mat分析 java headdump(jvm内存导出文件) 熟悉

 image-20200210175242209

 image-20200210175303638

- 选择 leak suspect 检查是否内存溢出

 image-20200210175432685

- 找开发人员 一起看

 image-20200210175634815

故障:

```
root@longpeng-ni ~]# jstack 905
```

```

905: well-known file /tmp/.java_pid905 is not secure: file's group should be the current group (which is
1001) but the group is 0
[root@longpeng-ni ~]# jps -lvm
1640 sun.tools.jps.Jps -lvm -Denv.class.path=./app/jdk/lib:/app/jdk/jre/lib:/app/jdk/lib/tools.jar -
Dapplication.home=/app/tools/jdk1.8.0_241 -Xms8m
905 org.apache.catalina.startup.Bootstrap start -
Djava.util.logging.config.file=/app/tomcat/conf/logging.properties -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.ephemeralDHkeySize=2048 -
Djava.protocol.handler.pkgs=org.apache.catalina.webresources -
Dorg.apache.catalina.security.SecurityListener.UMASK=0027 -Dcom.sun.management.jmxremote -
Dcom.sun.management.jmxremote.port=12345 -Dcom.sun.management.jmxremote.authenticate=false -
Dcom.sun.management.jmxremote.ssl=false -Djava.rmi.server.hostname=123.56.119.156 -Dignore.endorsed.dirs= -
Dcatalina.base=/app/tomcat -Dcatalina.home=/app/tomcat -Djava.io.tmpdir=/app/tomcat/temp
[root@longpeng-ni ~]#
[root@longpeng-ni ~]#
[root@longpeng-ni ~]#
[root@longpeng-ni ~]#
[root@longpeng-ni ~]#
[root@longpeng-ni ~]#
[root@longpeng-ni ~]# ll /tmp/.
./          ../          .font-unix/  .ICE-unix/   .java_pid905 .Test-unix/  .X11-unix/   .XIM-
unix/
[root@longpeng-ni ~]# ll /tmp/.
./          ../          .font-unix/  .ICE-unix/   .java_pid905 .Test-unix/  .X11-unix/   .XIM-
unix/
[root@longpeng-ni ~]# ll /tmp/.java_pid905
srw----- 1 root root 0 Feb 10 16:23 /tmp/.java_pid905
[root@longpeng-ni ~]#
[root@longpeng-ni ~]# id
uid=0(root) gid=1001(oldboy) groups=1001(oldboy)
[root@longpeng-ni ~]# jstack 905
905: well-known file /tmp/.java_pid905 is not secure: file's group should be the current group (which is
1001) but the group is 0
[root@longpeng-ni ~]# usermod -g root root
[root@longpeng-ni ~]# id
uid=0(root) gid=1001(oldboy) groups=1001(oldboy)
[root@longpeng-ni ~]# grep root /etc/group
root:x:0:
[root@longpeng-ni ~]# usermod -G root root
[root@longpeng-ni ~]# id
uid=0(root) gid=1001(oldboy) groups=1001(oldboy)
[root@longpeng-ni ~]# logout

```

Connection closed by foreign host.

Disconnected from remote host(阿里云(123.56.119.156)) at 17:09:47.

Type `'help'` to learn how to use Xshell prompt.

[c:\~]\$

Connecting to 123.56.119.156:22...

Connection established.

To escape to local shell, press `'Ctrl+Alt+J'`.

Last login: Mon Feb 10 15:32:44 2020 from 45.116.153.161

我住在天朝的帝都,身边有一条叫Jack的狗.

[root@longpeng-ni ~]#

[root@longpeng-ni ~]# id

uid=0(root) gid=0(root) groups=0(root)

## 2.10 https

- 单台tomcat
- tomcat集群(多台)
- 申请https证书
  - 域名(实名认证)
  - 配置dns解析
  - 申请https证书
- 客户端--->截取数据(中间人)-->服务器

### 1) 单台tomcat

- ☑ 准备好https证书 (下载tomcat专用), 配置dns解析(/etc/hosts内部劫持)
- ☑ https证书配置
- ☑ 配置tomcat开启https功能及配置证书
- ☑ 测试https ok?
- ☑ 配置tomcat 80自动跳转443
- ☑ 测试 80--->443 ?
- ☑ 单台tomcat https部署ok

[SSL证书安装指南 - SSL证书服务 - 阿里云\(aliyun.com\)](#)

- https 证书 配置转换

```
#conf
[root@web03 /app/tomcat/conf]# ll /app/tomcat/conf/zrlog.oldboylinux.cn.*
-rw-r--r-- 1 root root 4650 Jun  4 08:56 /app/tomcat/conf/zrlog.oldboylinux.cn.pfx
-rw-r--r-- 1 root root   8 Jun  4 08:56 /app/tomcat/conf/zrlog.oldboylinux.cn.txt

#转换格式 (如果需要的话) 跳过

zrlog.oldboylinux.cn
keytool -importkeystore -srckeystore zrlog.oldboylinux.cn.pfx -destkeystore zrlog.oldboylinux.cn.jks -
srcstoretype PKCS12 -deststoretype JKS
```

- tomcat 配置 文件 server.xml

```
#修改 Connector部分 把8080 ---->80 8443 -->443

<Connector port="80" protocol="org.apache.coyote.http11.Http11NioProtocol"
           connectionTimeout="20000"
           redirectPort="443" />

#配置 https 部分

<Connector port="443"
           protocol="org.apache.coyote.http11.Http11NioProtocol"
           maxThreads="150"
           SSLEnabled="true">
  <SSLHostConfig>
    <Certificate certificateKeystoreFile="/usr/local/tomcat/cert/证书域名.pfx"
```



```

        certificateKeystorePassword="证书密码"
        certificateKeystoreType="PKCS12" />
    </SSLHostConfig>
</Connector>

```

#修改后 Connector 80和443

```

<Connector port="80" protocol="org.apache.coyote.http11.Http11NioProtocol"
    connectionTimeout="20000"
    redirectPort="443" />

<Connector port="443"
    protocol="org.apache.coyote.http11.Http11NioProtocol"
    maxThreads="150"
    SSLEnabled="true">
    <SSLHostConfig>
        <Certificate certificateKeystoreFile="/app/tomcat/conf/zrlog.oldboylinux.cn.pfx"
            certificateKeystorePassword="L122wkzz"
            certificateKeystoreType="PKCS12" />
    </SSLHostConfig>
</Connector>

```


#虚拟主机配置

```

<Engine name="Catalina" defaultHost="zrlog.oldboylinux.cn">
    <Realm className="org.apache.catalina.realm.LockOutRealm">
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase"/>
    </Realm>
    <Host name="zrlog.oldboylinux.cn" appBase="webapps"
        unpackWARs="true" autoDeploy="true">
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
            prefix="localhost_access_log" suffix=".txt"
            pattern="%h %l %u %t &quot;%r&quot; %s %b" />
    </Host>
</Engine>

```

- hosts解析
- 测试 直接访问<https://zrlog.oldboylinux.cn> 使用https访问

 image-20210604091834089

- 配置tomcat 80自动跳转443

(可选步骤) 在web.xml文件最底部添加以下内容，实现HTTP自动跳转为HTTPS。

[root@web03 /app/tomcat/conf]# tail web.xml

```

<security-constraint>
    <web-resource-collection >
        <web-resource-name >SSL</web-resource-name>
        <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
</web-app>

```

- 浏览器检查与测试



```
# zrlog    /code/zrlog    server name=zrlog.oldboylinux.cn    appBase="/code/zrlog"
# jpress   /code/jpress   server name=jpress.oldboylinu    appBase="/code/jpress"
```

server.xml

```
<Connector port="80" protocol="org.apache.coyote.http11.Http11NioProtocol"
    connectionTimeout="20000"
    redirectPort="443" />
<Connector port="443"
    protocol="org.apache.coyote.http11.Http11NioProtocol"
    maxThreads="150"
    SSLEnabled="true">
    <SSLHostConfig>
        <Certificate certificateKeystoreFile="/app/tomcat/conf/aliyun.iawk.cn.pfx"
            certificateKeystorePassword="ov6B80kP"
            storePassword="ov6B80kP"
            certificateKeystoreType="PKCS12" />
    </SSLHostConfig>
</Connector>
```

`<Connector port="443"` #将Tomcat中默认的HTTPS端口Connector port 8443修改为443。8443端口不可通过域名直接访问、需要在域名后加上端口号；443端口是HTTPS的默认端口，可通过域名直接访问，无需在域名后加端口号。

`protocol="org.apache.coyote.http11.Http11NioProtocol"` #server.xml文件中Connector port有两种运行模式（NIO和APR），请选择NIO模式（也就是`protocol="org.apache.coyote.http11.Http11NioProtocol"`）这一段进行配置。

```
maxThreads="150"
SSLEnabled="true">
```

`<SSLHostConfig>`

```
<Certificate certificateKeystoreFile="/usr/local/tomcat/cert/证书域名.pfx" #此处
```

certificateKeystoreFile代表证书文件的路径，请用您证书的路径+文件名替换证书域名.pfx，例如：

```
certificateKeystoreFile="/usr/local/tomcat/cert/abc.com.pfx"
```

```
certificateKeystorePassword="证书密码" #此处certificateKeystorePassword为SSL证书的密码，请用您证书
```

密码文件pfx-password.txt中的密码替换，例如：`certificateKeystorePassword="bMNML1Df"`

```
certificateKeystoreType="PKCS12" /> #证书类型为PFX格式时，certificateKeystoreType修改为PKCS12。
```

`</SSLHostConfig>`


`</Connector>`

web.xml

```
<security-constraint>
    <web-resource-collection >
        <web-resource-name >SSL</web-resource-name>
        <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

```
<Host name="aliyun.iawk.cn" appBase="webapps"
```

## 2) tomcat集群部署 https

image-20210603151939332

|       |                              |            |
|-------|------------------------------|------------|
|       |                              |            |
| lb01  | 10.0.0.5/172/16.1.5          |            |
| web03 | tomcat 10.0.0.9/172/16.1.9   | 8080 zrlog |
| web04 | tomcat 10.0.0.10/172/16.1.10 | 8080 zrlog |

```
#web03 web04
zrlog.oldboy.com
sed '/<!--.*-->/d;/^$/d;/<!--/,/-->/d' /app/tomcat/conf/server.xml

[root@web04 ~]# cat /app/tomcat/conf/server.xml
<?xml version="1.0" encoding="UTF-8"?>
<Server port="8006" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.startup.VersionLoggerListener" />
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />
  <GlobalNamingResources>
    <Resource name="UserDatabase" auth="Container"
      type="org.apache.catalina.UserDatabase"
      description="User database that can be updated and saved"
      factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
      pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>
  <Service name="Catalina">
    <Connector port="80" protocol="org.apache.coyote.http11.Http11NioProtocol"
      connectionTimeout="20000"
      redirectPort="443" />
    <Engine name="Catalina" defaultHost="zrlog.oldboylinux.cn">
      <Realm className="org.apache.catalina.realm.LockOutRealm">
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
          resourceName="UserDatabase"/>
      </Realm>
      <Host name="zrlog.oldboylinux.cn" appBase="/code/zrlog"
        unpackWARs="true" autoDeploy="true">
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
          prefix="zrlog.oldboylinux.cn_access_log" suffix=".log"
          pattern="%h %l %u %t &quot;%r&quot; %s %b" />
      </Host>
      <Host name="jpress.oldboylinux.cn" appBase="/code/jpress"
        unpackWARs="true" autoDeploy="true">
        <Valve className="org.apache.catalina.valves.AccessLogValve" directory="/var/log/tomcat/"
          prefix="jpress.oldboylinux.cn_access_log" suffix=".log"
          pattern="%h %l %u %t &quot;%r&quot; %s %b" />
      </Host>
    </Engine>
  </Service>
```

```
</Server>
```

#web.xml 注释掉 自动跳转

```
<!--  
<security-constraint>  
    <web-resource-collection >  
        <web-resource-name >SSL</web-resource-name>  
        <url-pattern>/*</url-pattern>  
    </web-resource-collection>  
    <user-data-constraint>  
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>  
    </user-data-constraint>  
</security-constraint>  
-->
```

- lb01

```
[root@lb01 ~]# cat /etc/nginx/conf.d/zrlog.oldboylinux.cn.conf  
upstream zrlog {  
server 10.0.0.9:80;  
server 10.0.0.10:80;  
}  
server {  
    listen 80;  
    server_name zrlog.oldboylinux.cn;  
    return 302 https://zrlog.oldboylinux.cn$request_uri;  
}  
server {  
    listen 443 ssl;  
    server_name zrlog.oldboylinux.cn;  
    ssl_certificate ssl_key/server.crt;  
    ssl_certificate_key ssl_key/server.key;  
    location / {  
        proxy_pass http://zrlog;  
        include proxy_params;  
    }  
}
```

## 2.11 tomcat 与 nginx

- Inmt :动静分离
  - tomcat可以处理动态请求 也可以处理静态请求
  - tomcat处理静态资源效率不高
  - tomcat 处理动态请求 nginx处理静态请求
  - 要求:开发要把所有静态资源存放在固定位置 /var/jpress/static
- LNMP 原理
  - 静态资源 nginx自己处理
  - 动态资源 nginx--->php处理

image-20210603164100464

|       |        |                                                      |
|-------|--------|------------------------------------------------------|
|       |        |                                                      |
| lb01  |        |                                                      |
| web03 | 动态(默认) | 只运行tomcat即可 /code/zrlog/                             |
| web04 | 静态     | 只运行nginx 指定站点目录 /code/zrlog/ /code/zrlog/static/xxxx |
|       |        |                                                      |

```
#web04(web02)
[root@web04 ~]# cat /etc/nginx/conf.d/zrlog.oldboylinux.cn.conf
server {
    listen 80;
    server_name zrlog.oldboylinux.cn;
    root /code/zrlog;
}

#lb01
upstream zrlog_static {
    server 10.0.0.8:80;
}
upstream zrlog_dong {
    server 10.0.0.9:8080;
}
server {
    listen 80;
    server_name zrlog.oldboy.com;
    return 302 https://zrlog.oldboy.com$request_uri;
}
server {
    listen 443 ssl;
    server_name zrlog.oldboy.com;
    ssl_certificate ssl_key/server.crt;
    ssl_certificate_key ssl_key/server.key;
    location / {
        proxy_pass http://zrlog_dong;
        include proxy_params;
    }
    location ~* \.(png|jpg|jpeg|bmp|gif|js|html|css) {
        proxy_pass http://zrlog_static ;
    }
}
```

#模仿LNMP ----> LNMT

```
[root@web01 jpress]# cat /etc/nginx/conf.d/tomcat.conf
server {
    listen      80;
    server_name java.oldboy.com;
    client_max_body_size 10m;
    root        /application/tomcat/webapps;
    location / {
        index index.jsp index.html index.htm;
    }
    location ~ /\.jsp$ {
        proxy_pass 127.0.0.1:8080;
    }
}
```

- nginx负载均衡

#nginx反向代理 + tomcat

```
upstream tomcat {
    server 10.0.0.7:8080;
    server 10.0.0.7:8081;
    server 10.0.0.8:8080;
}

server {
    listen 80;
    server_name tomcat.oldboy.com;
    location / {
        proxy_pass http://tomcat ;
    }
}
```

## 2.12 Tomcat优化-优化部分-架构优化项目进行讲解

- Tomcat安全优化
- Tomcat性能优化
  - server.xml
  - catalina.sh

### 2.12.1 Tomcat安全优化

#安全优化

```
##1.telnet管理端口保护（强制） tomcat shutdown端口
<Server port="8527" shutdown="dangerous">
```

## ##2. ajp连接端口保护（推荐）

如果使用的apache+tomcat 修改端口

如果没有使用apache 则把这一行注释

```
<!-- 开始 注释结束 -->
```

```
116 <!-- <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" /> -->
```

## ##3. 禁用管理端（强制）

###0. 管理端主要应用在 tomcat测试环境中

###1. 删除默认的{Tomcat安装目录}/conf/tomcat-users.xml文件，重启tomcat后将会自动生成新的文件；

###2. 删除{Tomcat安装目录}/webapps下默认的所有目录和文件；

###3. 将tomcat 应用根目录配置为tomcat安装目录以外的目录；

```
[root@oldboy-tomcat webapps]# rm manager/ host-manager/ -fr
```

###4. 注释 conf/server.xml 部分

```
<Resource name="UserDatabase" auth="Container"
          type="org.apache.catalina.UserDatabase"
          description="User database that can be updated and saved"
          factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
          pathname="conf/tomcat-users.xml" />
```

## ##4. 降权启动（强制）（监牢模式 keep in jail）

在普通用户下面,运行维护tomcat

###1) 添加用户 tomcat

###2) 修改 tomcat 所有者

###3) 通过普通用户管理

```
useradd tomcat
chown -R tomcat.tomcat /app/tomcat/ /code/ /var/log/tomcat/
su - tomcat
```

```
/app/tomcat/bin/startup.sh
```

```
systemctl #需要sudo 授权
```

###1) 添加用户 tomcat

```
useradd tomcat
```

###2) 修改 tomcat 所有者

```
chown -R tomcat.tomcat /app/tomcat/
```

###3) 通过普通用户管理

```
su - tomcat
```

###4) 开机自启动 通过tomcat 用户

```
su - tomcat -c "/app/tomcat/bin/startup.sh"
```

-c 执行命令 后面只接普通命令

####5) 注意事项:

##1. 如果给nginx配置 端口在1-1024之间 端口叫特权端口 只能root管理维护

##2. 给nginx做 降权 需要修改端口

降权启动/监牢模式 让服务通过普通用户运行 普通用户管理.

作业. tomcat nginx 监牢模式 Linux 1-1024端口 特权端口 只能root使用.

web 8099

```
upstream web_pools {
```

```
server 10.0.0.7:8099;
server 10.0.0.8:8099;
```

```
}
```

##5. 文件列表访问控制（强制）

####默认展示站点目录下所有的内容

nginx 网站文件列表功能 autoindex on;

tomcat 关闭

conf/web.xml

```
<init-param>
<param-name>listings</param-name>
<param-value>>false</param-value>
</init-param>
```

##6. 版本信息隐藏（强制）

###不同软件版本 会有些不同的故障(漏洞)

###把tomcat错误页面指定到新的页面

conf/web.xml

```
<error-page>
<error-code>403</error-code>
<location>/forbidden.jsp</location>
</error-page>
<error-page>
<error-code>404</error-code>
<location>/notfound.jsp</location>
</error-page>
<error-page>
<error-code>500</error-code>
<location>/systembusy.jsp</location>
</error-page>
```

##7. Server header重写（推荐）

###tomcat 想让用户发下 这是nginx

server.xml

```
<Connector port="8080" protocol="org.apache.coyote.http11.Http11AprProtocol"
server="nginx/1.16.1"
connectionTimeout="20000"
redirectPort="8443" />
```

```
[root@oldboy-tomcat ~]# curl -I http://47.114.128.48:8080
```

```
HTTP/1.1 200
```

```
Content-Type: text/html; charset=UTF-8
```

```
Transfer-Encoding: chunked
```

```
Date: Tue, 11 Feb 2020 02:43:19 GMT
```

```
Server: nginx/1.16.1
```

#8. 访问限制（可选）

```
<Context path="/" docBase="/home/work/tomcat" debug="0" reloadable="false" crossContext="true">
<Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="61.148.18.138,61.135.165.*"
deny="*.*.*.*/>
</Context>
```

#9. 脚本权限回收

```
chmod -R 700 tomcat/bin/*
```

#10. 配置完整访问日志格式

```
<Valve className="org.apache.catalina.valves.AccessLogValve"
directory="logs" prefix="localhost_access_log." suffix=".txt"
pattern="%h %l %u %t %r %s %b %{Referer}i %{User-Agent}i %D" resolveHosts="false"/>
```



`%{Referer}i` 用户从哪里跳转过来的  
`%{User-Agent}i` 用户客户端(浏览器)

`pattern="%h %l %u %t &quot;%r&quot; %s %b %D &quot;%{Referer}i&quot; &quot;%{User-Agent}i&quot;";`

## 2.13 Tomcat性能优化

### 2.13.1 调优思路与工具

调优 参数数值调整方式:

- 参考值:(基准) 不设置数值,通过压力测试软件 看看Tomcat性能
- 增加优化参数 ,通过压力测试软件 看看Tomcat性能

| 压力测试工具      |                     |  |
|-------------|---------------------|--|
| ab/webbench | http压力测试            |  |
| stress      | 压力测试 cpu 内存 swap .. |  |
| jmeter      | 压力测试java(tomcat)    |  |
| dd/fio      | 测试磁盘性能              |  |
| mysqlslap   | 压力测试数据库(MySQL)      |  |
| loadrunner  | 专业测试工具              |  |

### 2.13.2 Jmeter使用指南

- 准备jdk环境
- 启动jmeter
  - bin/jmeter
  - jmeter #linux启动命令
  - jmeter.bat #windows启动命令

image-20200211115215801

- jmeter启动后页面

image-20200211115327112

- 修改语言

image-20200211115652489

image-20200211115746085

- 创建测试用例(让jmeter 能压力测试tomcat)

- 修改测试用例名称



- ◦ 添加线程组



- ◦ 添加取样器 (如何进行压力测试 http ftp ...)



- ◦ 监听器:
  - 查看结果树(每一个请求信息)
  - 聚合报告(结果 统计 分析)



## 2.13.3 Tomcat准备

```
#全新Tomcat
[root@oldboy-tomcat ~]# cd /app/tools/
[root@oldboy-tomcat tools]# ll
total 220368
-rw-r--r-- 1 root root 10305939 Dec 8 03:42 apache-tomcat-8.5.50.tar.gz
-rw-r--r-- 1 root root 194545143 Feb 7 10:13 jdk-8u241-linux-x64.tar.gz
-rw-r--r-- 1 root root 20797013 Apr 9 2018 jpress.war
[root@oldboy-tomcat tools]# tar xf apache-tomcat-8.5.50.tar.gz
[root@oldboy-tomcat tools]# ll
total 220368
drwxr-xr-x 9 root root 220 Feb 11 12:19 apache-tomcat-8.5.50
-rw-r--r-- 1 root root 10305939 Dec 8 03:42 apache-tomcat-8.5.50.tar.gz
-rw-r--r-- 1 root root 194545143 Feb 7 10:13 jdk-8u241-linux-x64.tar.gz
-rw-r--r-- 1 root root 20797013 Apr 9 2018 jpress.war
[root@oldboy-tomcat tools]# mv apache-tomcat-8.5.50 tomcat-jmeter
[root@oldboy-tomcat tools]# mv tomcat-jmeter/ /app/
[root@oldboy-tomcat tools]# ll /app/tomcat
lrwxrwxrwx 1 root root 26 Feb 7 10:30 /app/tomcat -> /app/apache-tomcat-8.5.50/
[root@oldboy-tomcat tools]# rm -f /app/tomcat
[root@oldboy-tomcat tools]# ln -s /app/tomcat-jmeter/ /app/tomcat
[root@oldboy-tomcat tools]# ll /app/tomcat
lrwxrwxrwx 1 root root 19 Feb 11 12:21 /app/tomcat -> /app/tomcat-jmeter/
```

## 2.13.4 进行基准测试

测试: 1线程 测试 循环:永远



- 基准值:吞吐量 10 左右
-

```
[root@oldboy-tomcat ~]# w
14:35:15 up 4:08, 2 users, load average: 0.05, 0.01, 0.00
USER      TTY      FROM          LOGIN@      IDLE   JCPU   PCPU   WHAT
root      pts/0    117.136.94.82 10:27       2:14m 34.84s 0.18s -bash
root      pts/1    117.136.94.82 14:35       2.00s  0.01s  0.00s w
```

- 开启tomcat管理端 查看 连接信息

```
[root@oldboy-tomcat ~]# grep allow= /app/tomcat/webapps/host-manager/META-INF/context.xml
/app/tomcat/webapps/manager/META-INF/context.xml
/app/tomcat/webapps/host-manager/META-INF/context.xml:
allow="\d+\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
/app/tomcat/webapps/manager/META-INF/context.xml:          allow="\d+\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />

[root@oldboy-tomcat ~]# cat /app/tomcat/conf/tomcat-users.xml
<?xml version="1.0" encoding="UTF-8"?>
<tomcat-users xmlns="http://tomcat.apache.org/xml"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
               version="1.0">
  <role rolename="manager-gui"/>
  <role rolename="admin-gui"/>
  <user username="tomcat" password="tomcat" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

## 2.13.5 Tomcat 配置参数优化 ✨ ✨ ✨ ✨ ✨

- server.xml

```
#1. 注释 ajp 端口
116      <!--      <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" /> -->
```

```
#2. tomcat 工作模式 tomcat io模型
bio tomcat 7 同步 阻塞
nio tomcat 8 异步 非阻塞      : nio1 (默认)  nio2
apr  高并发 额外安装插件
```

```
#默认
69      <Connector port="8080" protocol="HTTP/1.1"
70          connectionTimeout="20000"
71          redirectPort="8443" />
```

修改为: `protocol="org.apache.coyote.http11.Http11Nio2Protocol"`

- 配置apr模式

```
#安装apr环境
yum -y install apr apr-devel
```

```
yum install -y tomcat-native #centos 7 直接yum安装
```

#注意:CentOS8中 yum源没有tomcat-native 需要编译安装

```
[root@oldboy-tomcat bin]# cd /app/tomcat/bin/
[root@oldboy-tomcat bin]# ll tomcat-native.tar.gz
-rw-r----- 1 root root 419428 Dec  8 03:21 tomcat-native.tar.gz
[root@oldboy-tomcat bin]# tar xf tomcat-native.tar.gz
[root@oldboy-tomcat bin]# cd tomcat-native-1.2.23-src/
docs/      examples/  java/      native/    test/      xdocs/
[root@oldboy-tomcat bin]# cd tomcat-native-1.2.23-src/native/
[root@oldboy-tomcat native]# ls
build      BUILDING      configure     libtcnative.dsp  Makefile.in      NOTICE.bin.win  src/lib
tcnative.spec
build.conf  build-outputs.mk  configure.in  libtcnative.dsw  NMAKEmakefile    os
tcnative.dsp
buildconf   config.layout    include      LICENSE.bin.win  NMAKEmakefile.inc  src
tcnative.pc.in
[root@oldboy-tomcat native]# ./configure && make && make install
```

```
cat >>/etc/profile<<'EOF'
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/apr/lib
export LD_RUN_PATH=$LD_LIBRARY_PATH:/usr/local/apr/lib
EOF
```

```
. /etc/profile
```

#修改8080&8009端口对应的server.xml

```
protocol="org.apache.coyote.http11.Http11Nio2Protocol"
```

把Nio2 修改为Apr

```
protocol="org.apache.coyote.http11.Http11AprProtocol"
```

#重启tomcat后 检查日志

```
11-Feb-2020 15:28:09.567 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler
["http-apr-8080"]
11-Feb-2020 15:28:09.580 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 464 ms
```

•

```
maxThreads="500"      #最大的线程数量 200-400之间
acceptCount=""        #当达到最大线程数量的时候 队列长度
                      #一般与maxThreads 一致
acceptorThreadCount="2" #分为几对 与cpu核心总数一致或2倍 默认是1
```

```
minSpareThreads="10"  #空闲时候最小的线程数量 不忙的时候 最少留几个服务员
```

#

maxThreads如何配置

#包括量方面: 1计算(主要消耗cpu), 2等待(io、数据库等)

##第一种极端情况, 如果我们的操作是纯粹的计算, 那么系统响应时间的主要限制就是cpu的运算能力, 此时maxThreads应该尽量设的小, 降低同一时间内争抢cpu的线程个数, 可以提高计算效率, 提高系统的整体处理能力。

##第二种极端情况，如果我们的操作纯粹是IO或者数据库，那么响应时间的主要限制就变为等待外部资源，此时maxThreads应该尽量设的大，这样 才能提高同时处理请求的个数，从而提高系统整体的处理能力。此情况下因为tomcat同时处理的请求量会比较大，所以需要关注一下tomcat的虚拟机内 存设置和linux的open file限制。

我在测试时遇到一个问题，maxThreads我设置的比较大比如3000，当服务的线程数大到一定程度时，一般是2000出头，单次请求的响应时间就会急剧的增加，

百思不得其解这是为什么，四处寻求答案无果，最后我总结的原因可能是cpu在线程切换时消耗的时间随着线程数量的增加越来越大，

#cpu处理数据 让我们感觉 同时在处理 实际上 cpu是在不同的线程/进程之间进行 快速切换

```
enableLookups="false"           #禁止DNS逆向查询    ip-->域名    route -n /arp -n

compression="on"                 #开启 tomcat压缩功能 静态文本资源 html js  css

compressionMinSize="2048"       #压缩文件 最小2048字节

compressableMimeType="text/html,text/plain,text/css,application/javascript,application/json,application/x-
font-ttf,application/x-font-otf" #压缩哪些类型的文件 文本类型 js,css,html

disableUploadTimeout="true"     #关闭上传文件超时时间

redirectPort="8443" />
```

#disableUploadTimeout="true" 故障案例：tomcat 因为用户网络问题 Read timed out

在上传文件过程中由于网速比较慢可能会屡次出现下列问题：

org.apache.commons.fileupload.FileUploadBase\$IOFileUploadException: Processing of multipart/form-data request failed. Read timed out

很明显，出现这种问题的原因是读取文件超时，解决方法是将HTTP Keep-Alive Timeout这个参数设置地尽量大。如果使用tomcat做服务器的话，我们可以通过修改服务器配置来解决该问题，具体的解决方法如下：

修改tomcat的配置文件conf/server.xml，找到下面配置信息：

#tomcat 配置文件参数优化

```
<Connector port="8080" protocol="org.apache.coyote.http11.Http11AprProtocol"
    maxThreads="500"
    acceptCount="500"
    acceptorThreadCount="2"
    minSpareThreads="10"
    enableLookups="false"
    compression="on"
    compressionMinSize="2048"

    compressableMimeType="text/html,text/plain,text/css,application/javascript,application/json,application/x-
font-ttf,application/x-font-otf"
    disableUploadTimeout="true"
    connectionTimeout="20000"
    redirectPort="8443" />
```

- java 启动参数 (catalina.sh) jvm 优化

#设置 jvm初始内存大小 (物理内存1/64) jvm最大内存大小 (物理内存的1/4 )

##修改 catalina.sh文件

JAVA\_OPTS='-Xms1024m -Xmx1024m -Xloggc:/var/log/tomcat\_gc.log'

#-Xms jvm 初始内存

#-Xmx max 最大 jvm最大内存

#一般 -Xmx 是 -Xms 2倍

#gc garbage collect 垃圾回收 定期清理 jvm内存

#-Xloggc 执行gc的时候 日志 出现故障tomcat崩溃 挂了 catalina.out gc日志

#扩展:

java程序代码方式:

#war包 扔到 tomcat webapps 目录下

#jar包 相当于里面集成了1个tomcat java -jar xxx.jar

#如果是 java命令 运行 jar包 配置参数

加在 java 命令后面即可

java -jar xxxx.jar -Xms1024m -Xmx1024m -Xloggc:/var/log/tomcat\_gc.log

## 2.14 Tomcat总结

☑ tomcat jvm jdk jre 区别❄❄❄❄❄

☑ tomcat 环境部署

☑ jdk 1.8

☑ tomcat 8.5.50

☑ tomcat目录结构 ❄❄❄❄

☑ bin/ startup.sh shutdown.sh catalina.sh (配置tomcat监控功能 配置java(tomcat) 启动参数 jvm优化)

☑ conf/ server.xml web.xml tomcat-user.xml

☑ logs / catalina.out (每日切割 切割后文件内容不会被清空) catalina.2019-12-12.log access.log

☑ 日志: /var/log/messages secure cron

☑ rsync: /var/log/rsyncd.log

☑ nginx: access.log error.log

☐ zabbix: zabbix\_server.log zabbix\_agentd.log

☐ tomcat: catalina.out localhost\_access\_log.2019-12-17.txt

☑ webapps

☑ 配置tomcat管理端 ❄

☑ tomcat的配置文件: tomcat默认端口 虚拟主机 ❄❄❄❄❄

☑ tomcat 部署应用及方式 ❄❄❄❄❄

☑ tomcat3种工作模式: bio , nio , apr ❄❄❄

☑ tomcat 多实例: 端口不同 路径 ❄❄❄

☑ tomcat 监控功能 ❄❄

☑ tomcat 与 nginx

☑ **tomcat 故障及排查 jstack jmap** ❸❸❸❸❸

☑ **tomcat 优化** ❸❸❸❸❸

☑ 安全优化

☑ 性能优化

☑ tomcat 性能及压力测试

任务：

1. 总结 排错与故障
2. Tomcat优化
3. 压力测试 Jmeter
4. 作业：研究其他测试工具：ab stress dd fio

1. linux命令

2. 三剑客

3. 服务管理:端口 ip,进程...内存,curl,wget