# Ali-Tomcat

hongjiang
2014.5

# 关于

- blog：http://hongjiang.info
- 微博：http://weibo.com/woodcafe

- 经历：
  - Java : 11y+
  - Scala : 3y+
  - 曾在阿里巴巴中文站和laiwang.com担任架构师，现在中间件负责应用容器
  - Scala布道者, 业余马拉松爱好者

# 大纲

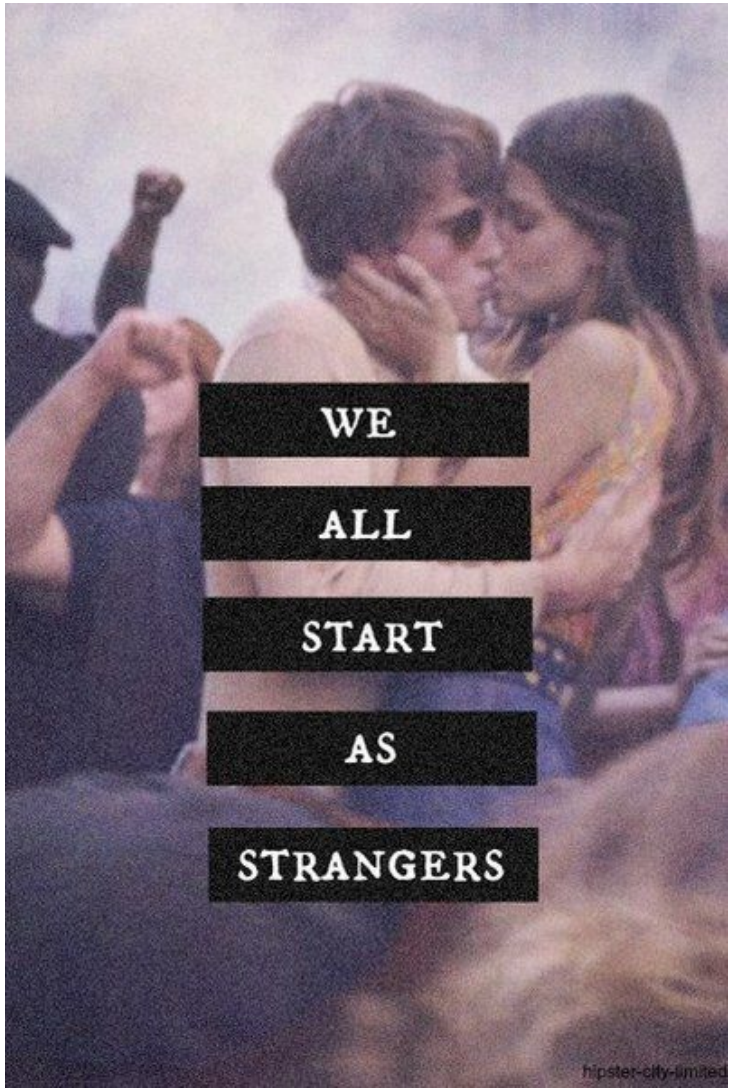
Apache Tomcat

1) 如何设计一个App Server
2) Tomcat运转流程
3) 一些干货(Tips)

Ali-Tomcat

1) How alibaba use tomcat & what we did
2) Pandora: 一个轻量级模块化系统
3) 监控与诊断
4) 日志，工具链

# 熟悉的陌生人

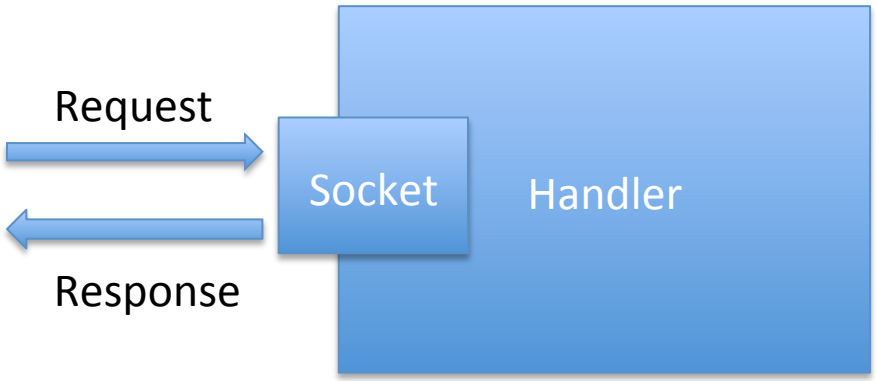


# 你会如何设计一个 app server ?

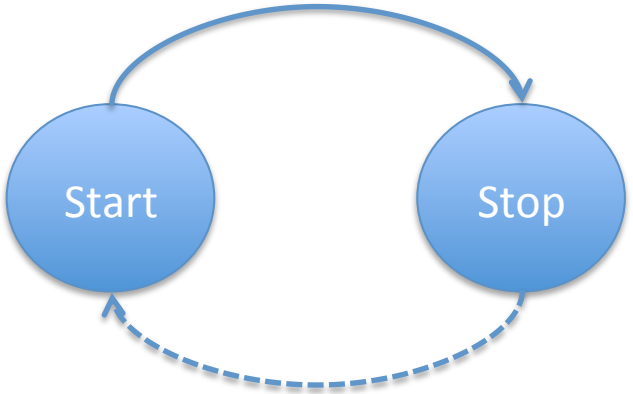
# 设计一个App Server

从组成部分和生命周期两个维度考虑

一个最简单的server：

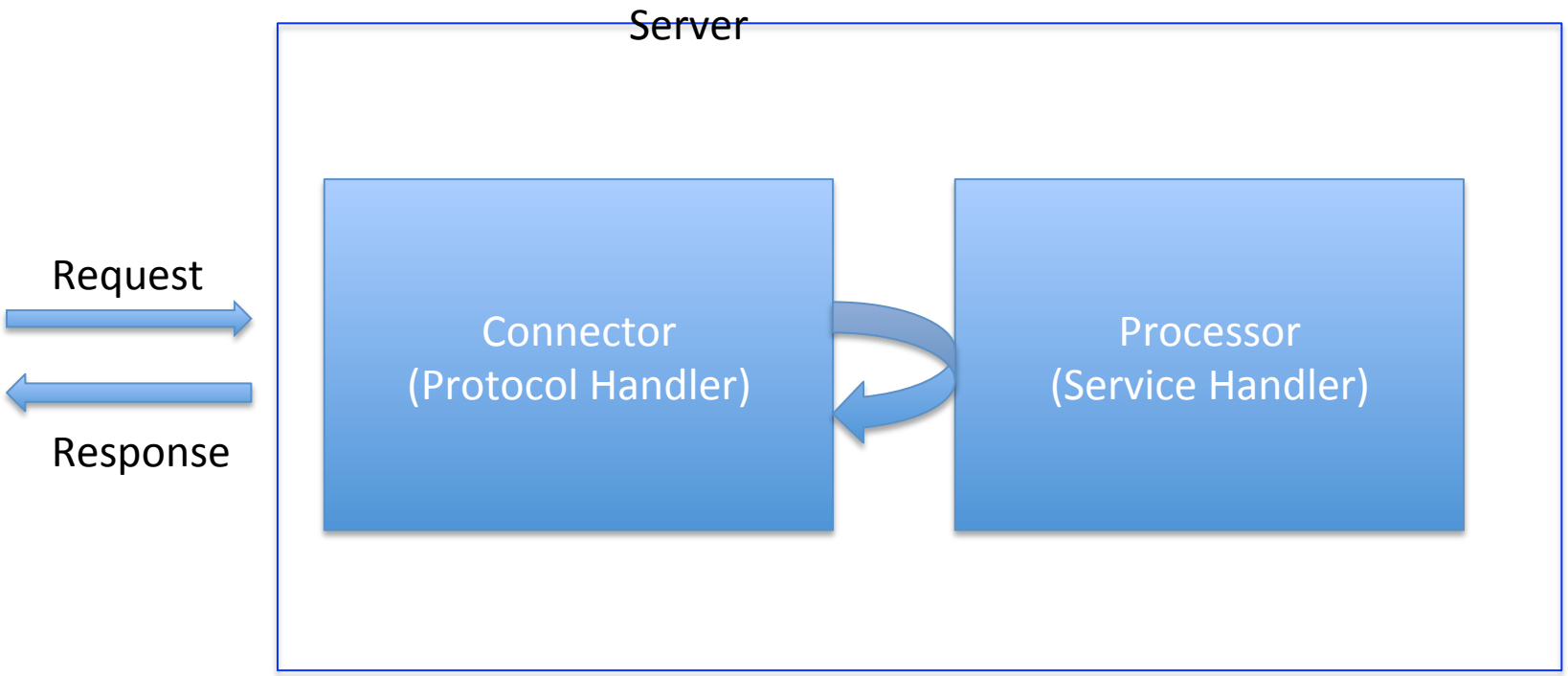



Lifecycle:



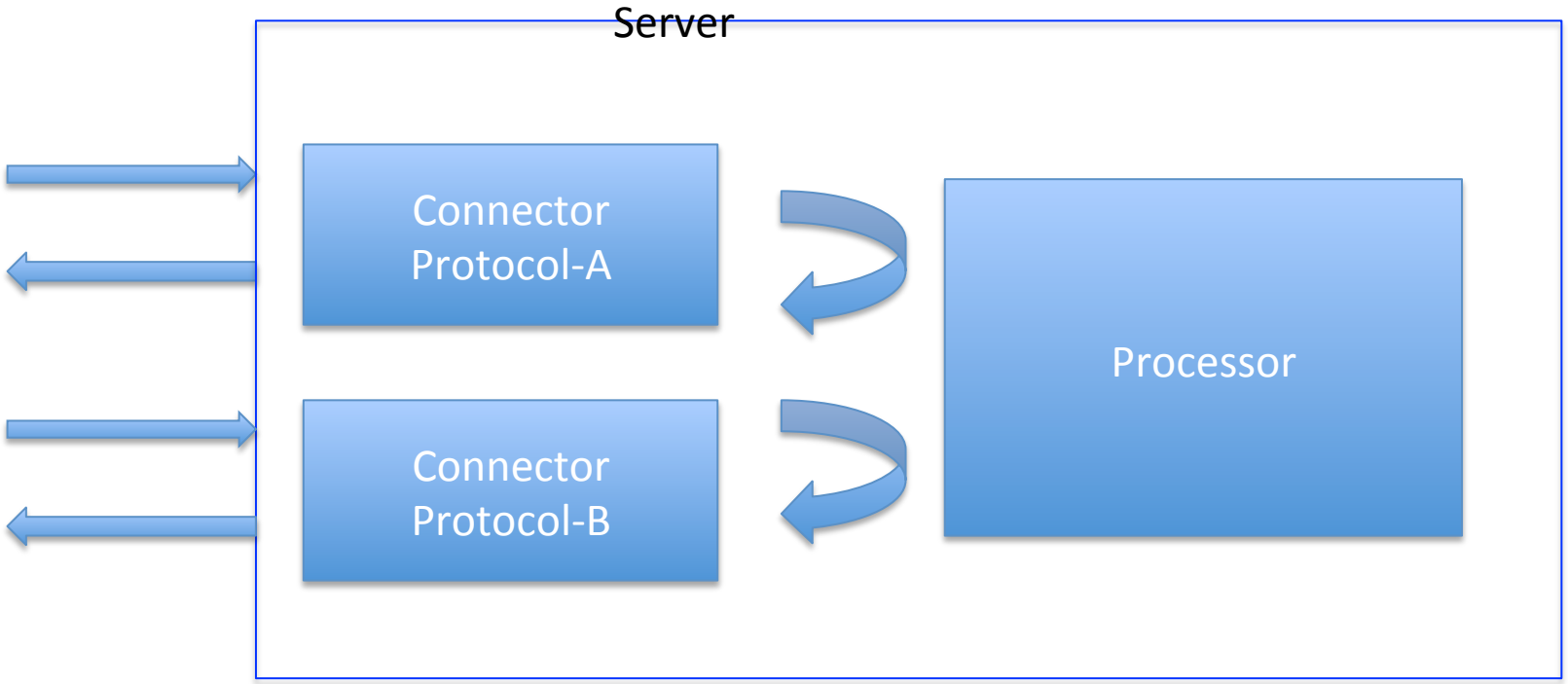
# 设计一个App Server

- 分离角色: 协议 & 逻辑



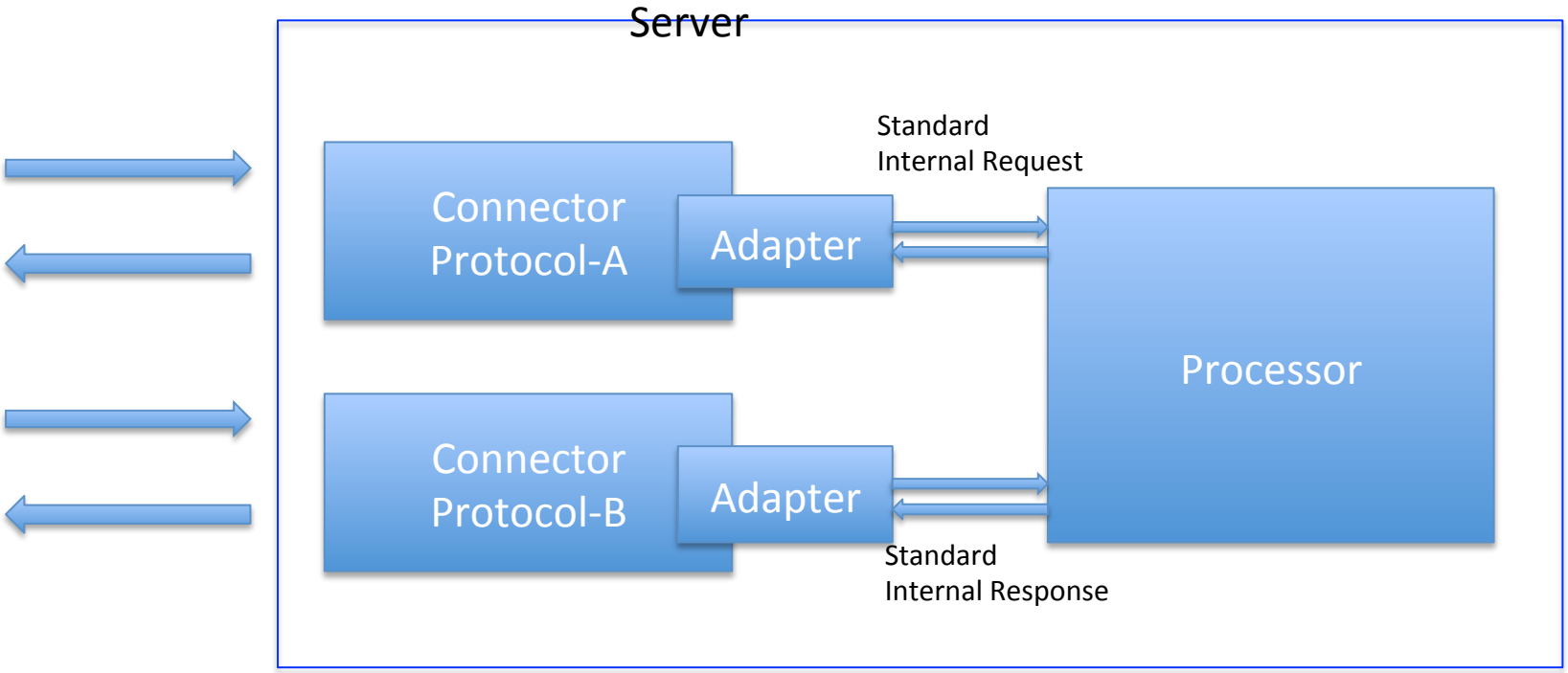
# 设计一个App Server

- 多协议的支持



# 设计一个App Server

- 适配成一致的内部Request/Response模型



# 设计一个App Server

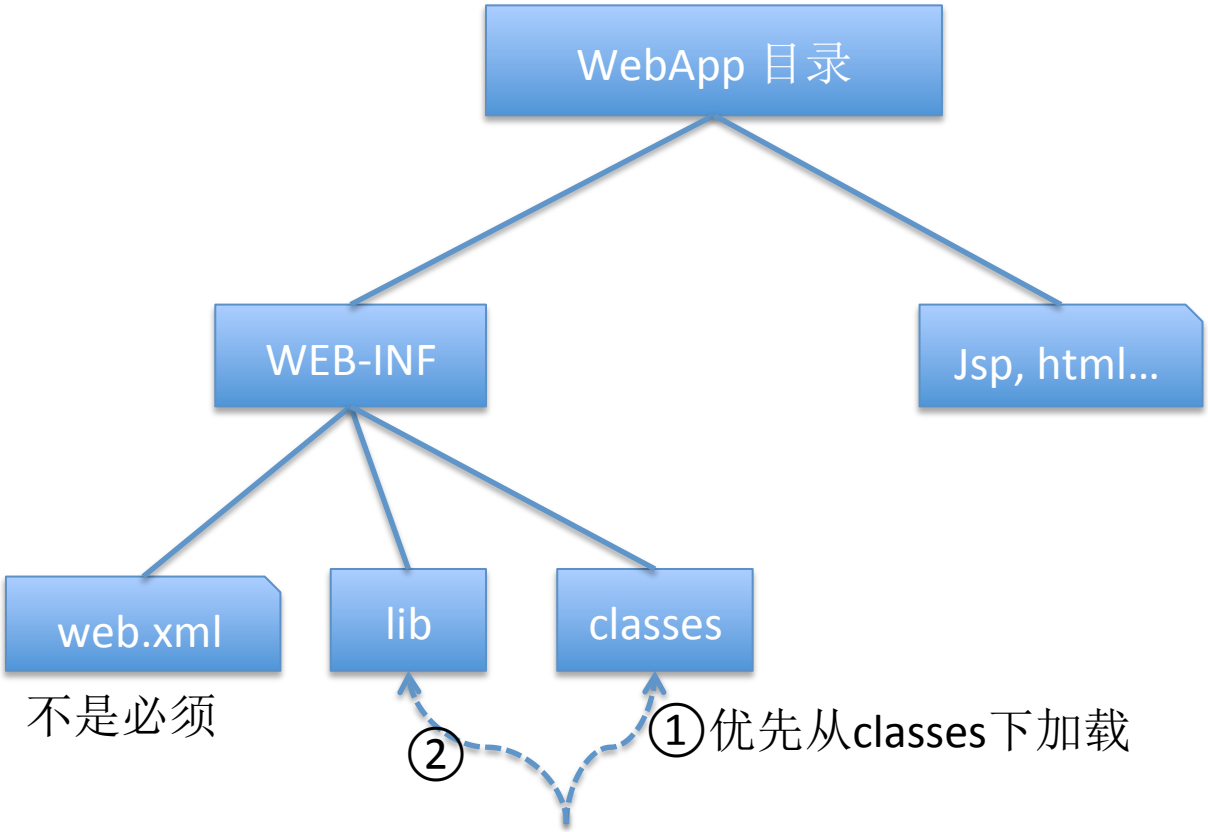
- 接下来，如何托管应用：

1) 怎么定义一个应用?
2) 怎么部署应用?
3) 怎么加载应用?
4) 怎么卸载应用?
5) …

Servlet编程模型和规范

# 设计一个App Server

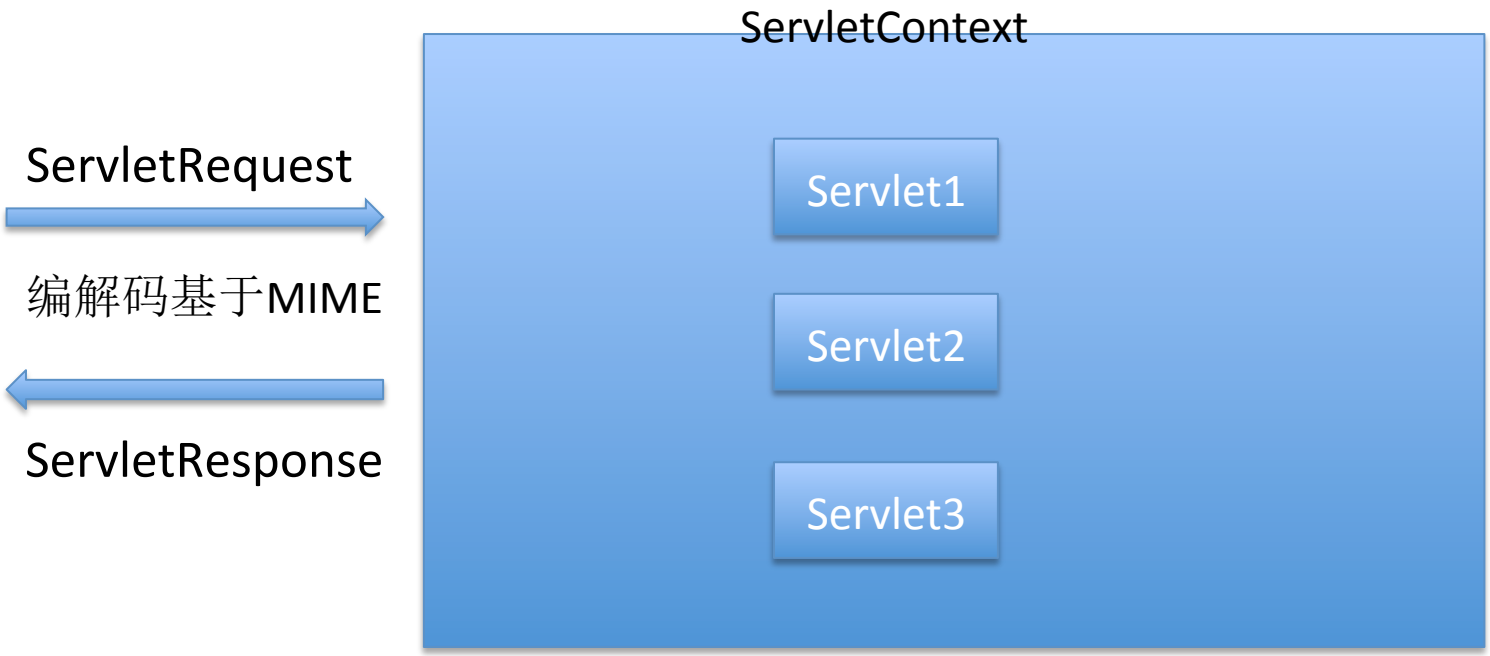
- Servlet: 应用目录结构




Classloader: 每个App有自己单独的classloader实例
线程在进入和退出webapp逻辑时要进行classloader的切换

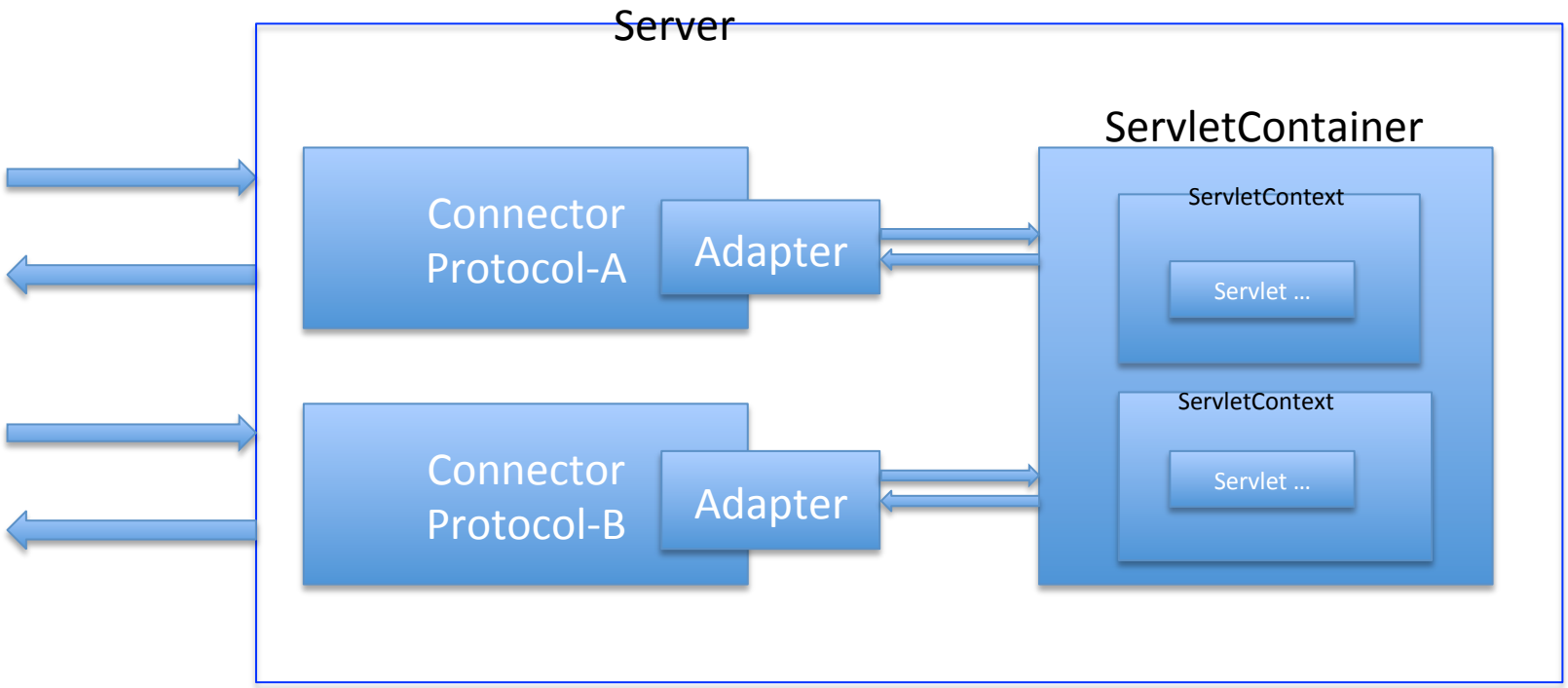
# 设计一个App Server

- Servlet: 一个WebApp对应一个ServletContext



# 设计一个App Server

- 套用servlet模型



# 设计一个App Server

- 接下来还有很多要考虑的：

- 1) 全局层面：
  - **如何以**Daemon进程的方式运行
  - 如果优雅的关闭Server进程
  - **是否提供**JMX
  - 日志如何处理
  - 是否可配置？如何配置
  - 是否可扩展？扩展点，扩展方式

- 2) Connector部分：
  - 具体支持哪些协议(只支持http就够吗)
  - IO**模型**：同步/异步？阻塞/非阻塞？

- 3) Servlet Container 部分：
  - 实现servlet 规范里的要求，大量的细节

# Apache Tomcat 的设计者是如何考虑这些问题的？

我们重点关注：

1) Connector的实现(NIO)
2) Container 的设计
3) Server的启动和关闭

# Apache-Tomcat

James Duncan Davidson

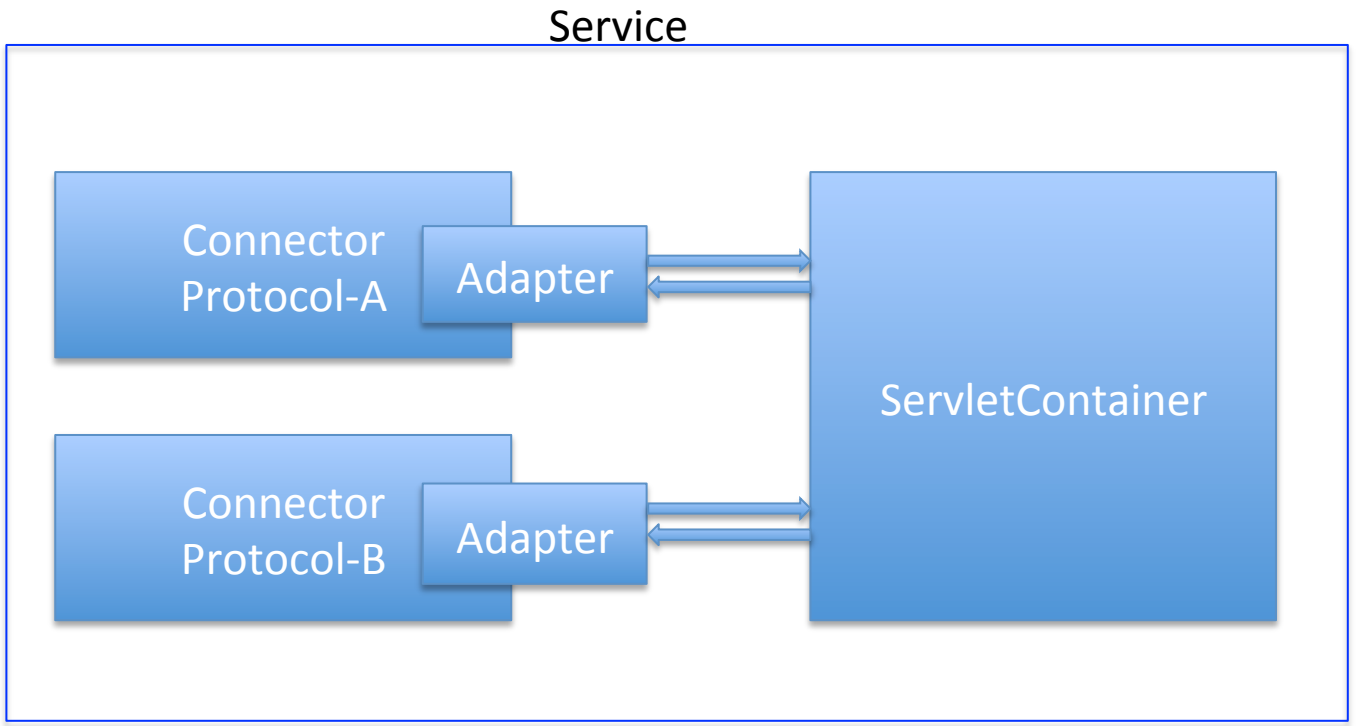
Tomcat & Ant 创始人

Craig R. McClanahan

Tomcat 架构师
Struts 创始人
Jive首席工程师

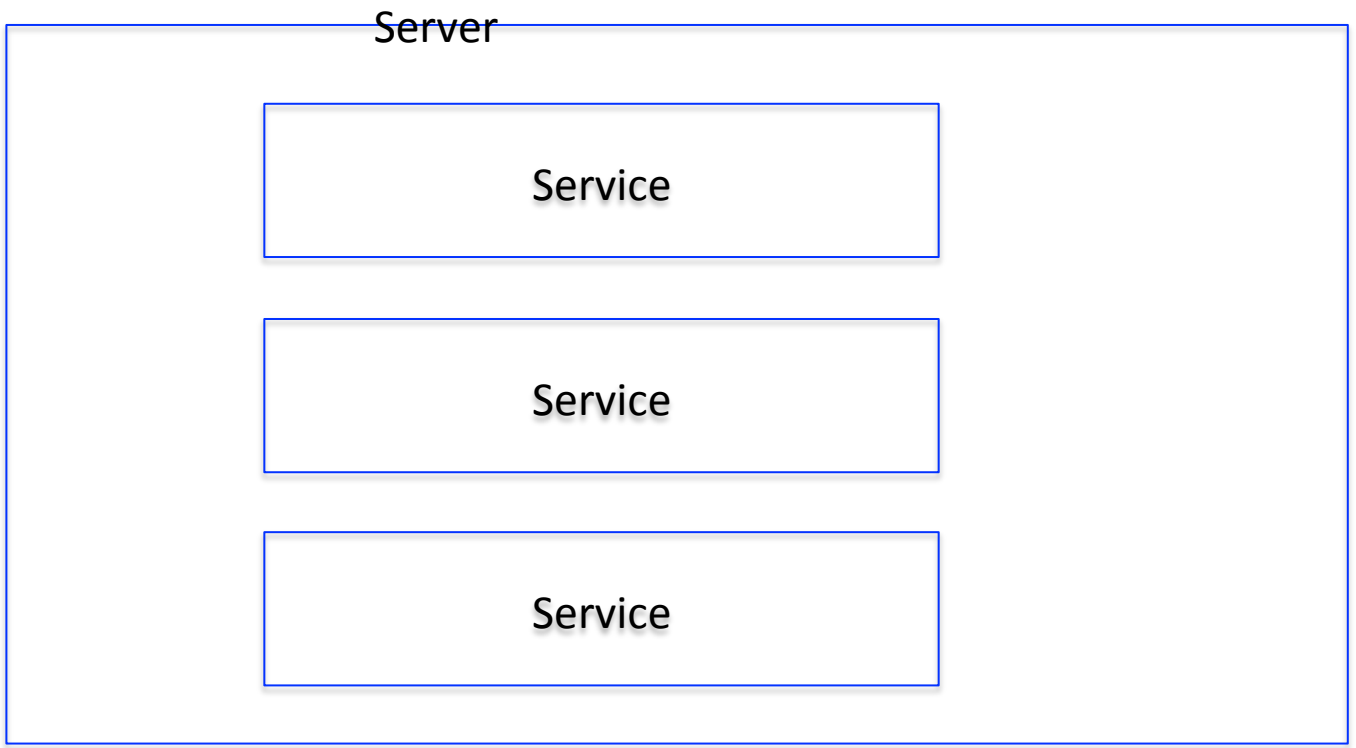
# Apache-Tomcat

- 从组件的角度，tomcat也分成了connector和container，两个组件又封装在service组件里:

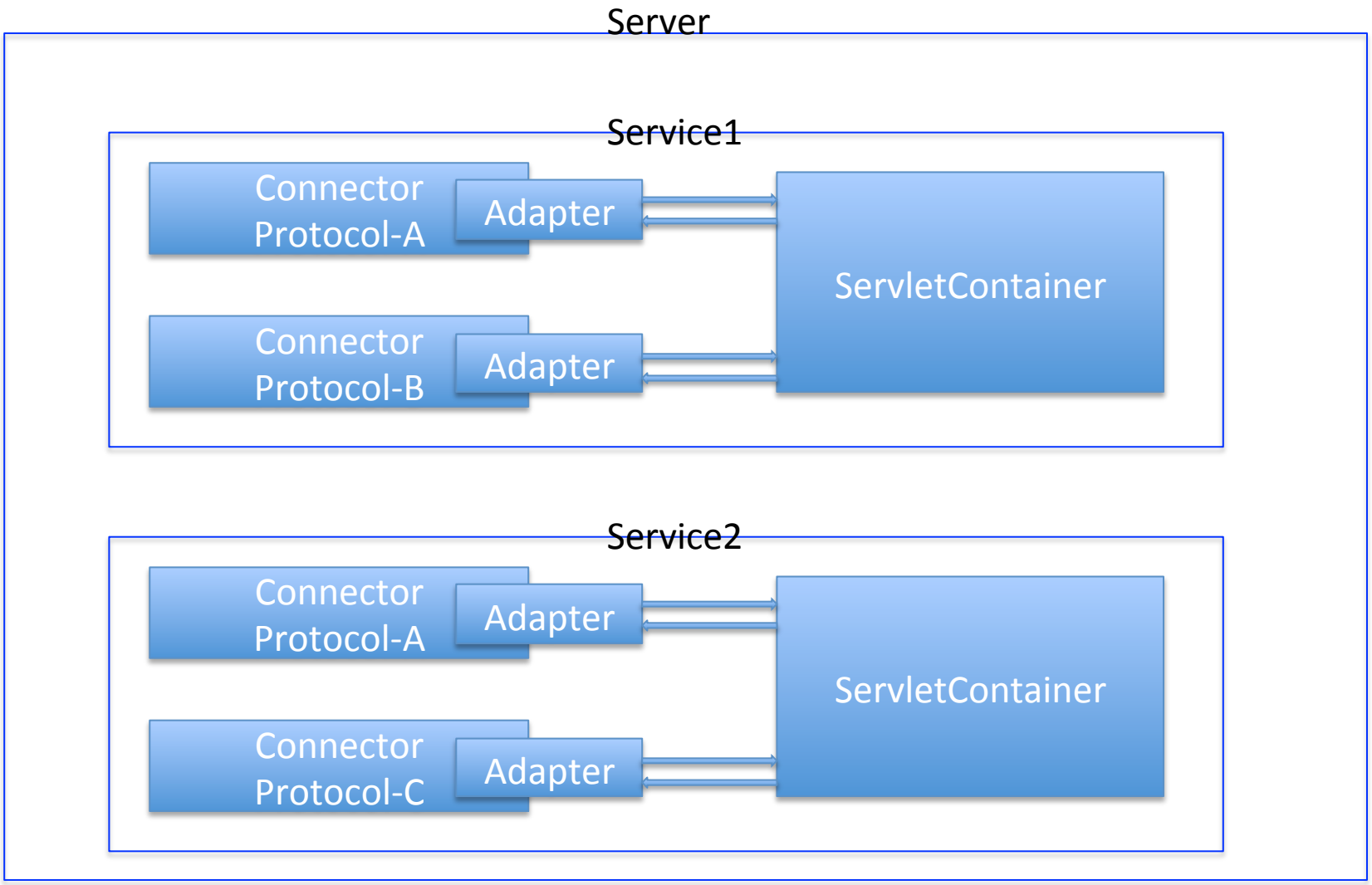


# Apache-Tomcat

- Tomcat里Server可以包含一组Service:

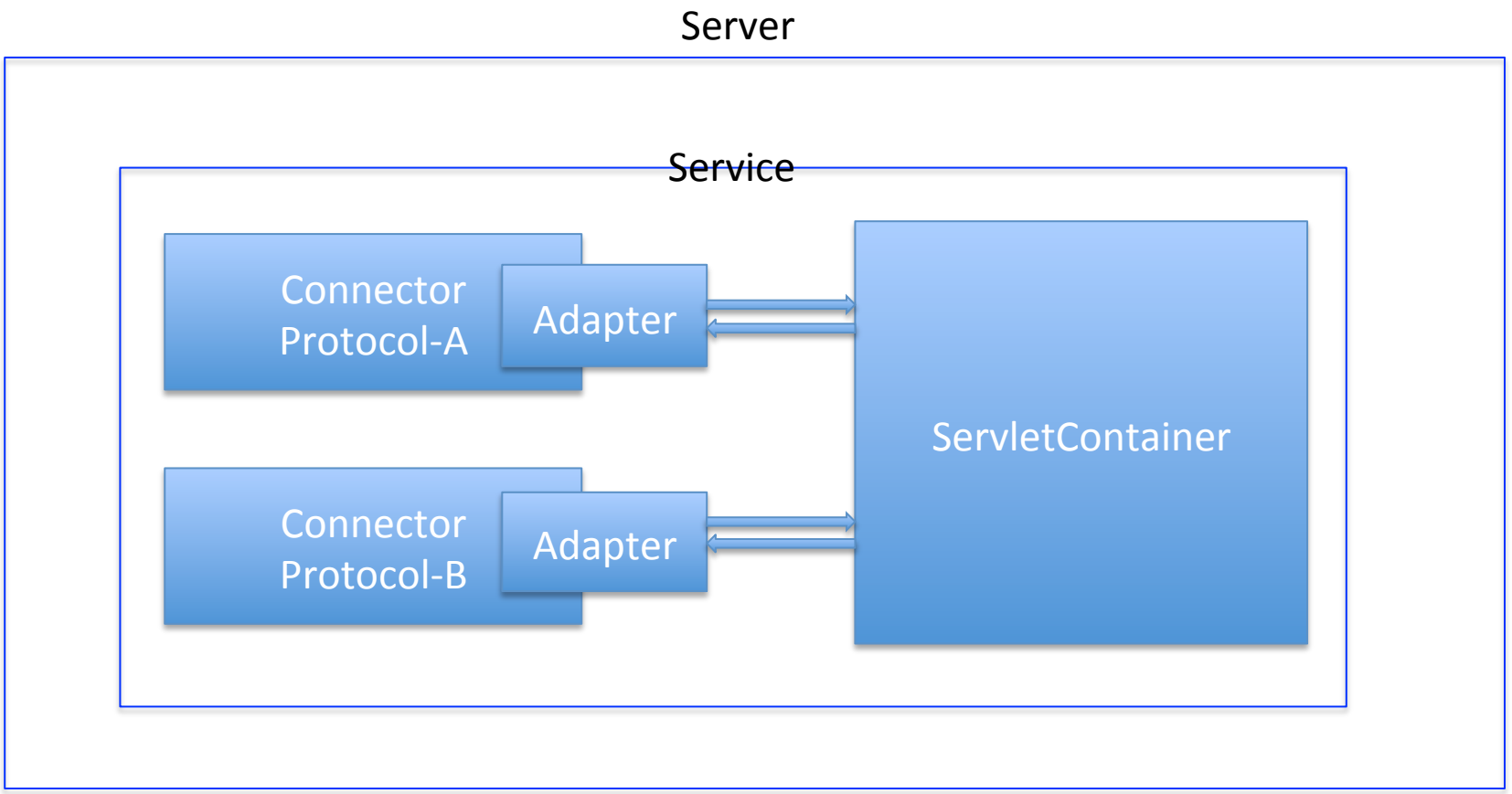


# Apache-Tomcat



# Apache-Tomcat

- 通常只有1个service，有1-2个connector

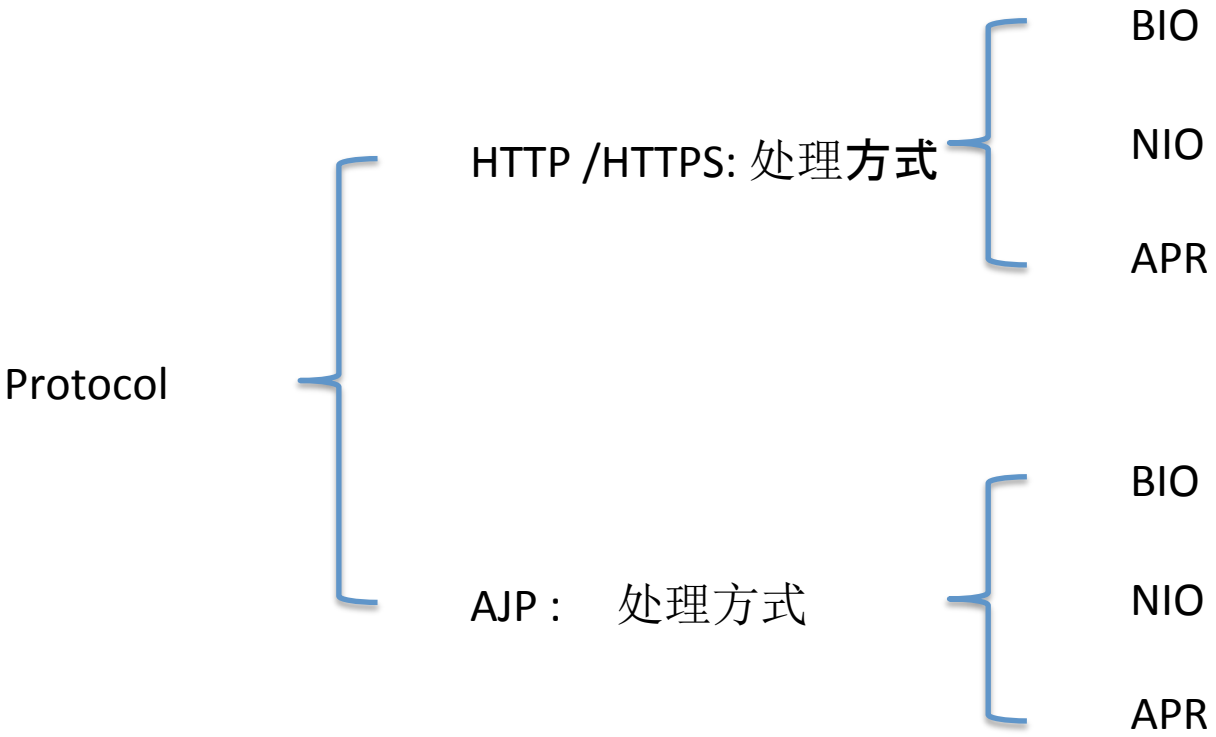


# Apache-Tomcat：Connector

- 代号 Coyote: 一种北美大草原的小狼

# Apache-Tomcat：Connector

- 协议及实现方式：



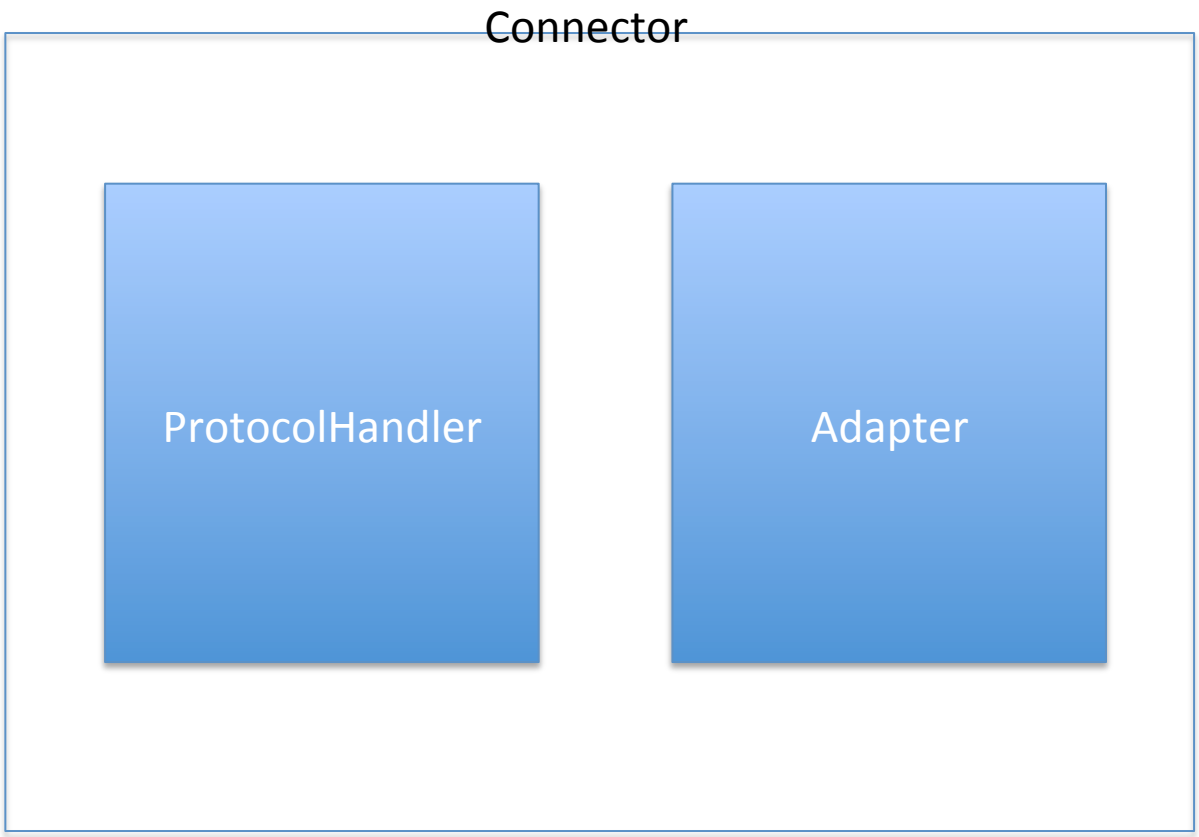

AJP: Apache JServ Protocol
APR: Apache Portable Runtime

# Apache-Tomcat：Connector

| | BIO | NIO | APR |
|---|---|---|---|
| Classname | Http11Protocol | Http11NioProtocol | Http11AprProtocol |
| Tomcat Version | 3.x onwards | 6.x onwards | 5.5.x onwards |
| Support Polling | NO | YES | YES |
| Polling Size | N/A | maxConnections | maxConnections |
| Read HTTP Request | Blocking | Non Blocking | Blocking |
| Read HTTP Body | Blocking | Sim Blocking | Blocking |
| Write HTTP Response | Blocking | Sim Blocking | Blocking |
| Wait for next Request | Blocking | Non Blocking | Non Blocking |
| SSL Support | Java SSL | Java SSL | OpenSSL |
| SSL Handshake | Blocking | Non blocking | Blocking |
| Max Connections | maxConnections | maxConnections | maxConnections |

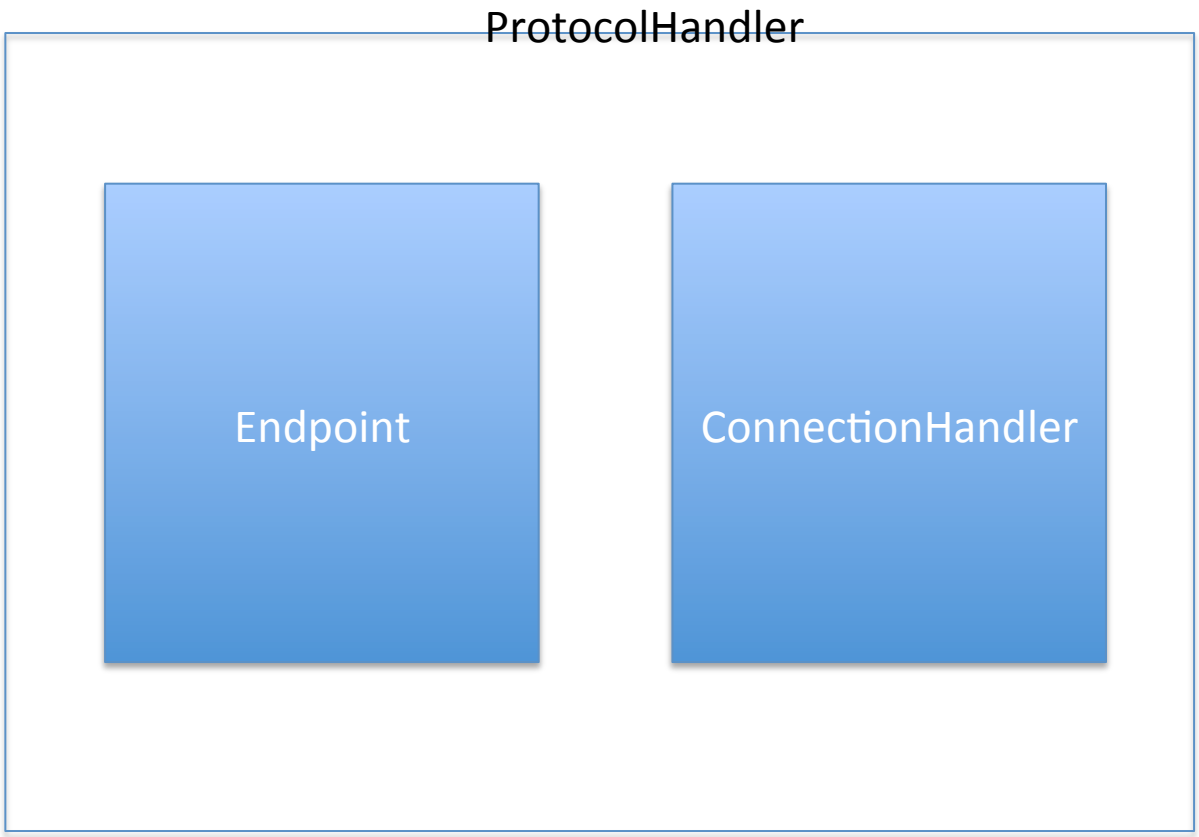
# Apache-Tomcat：Connector

- Connector 的组成：



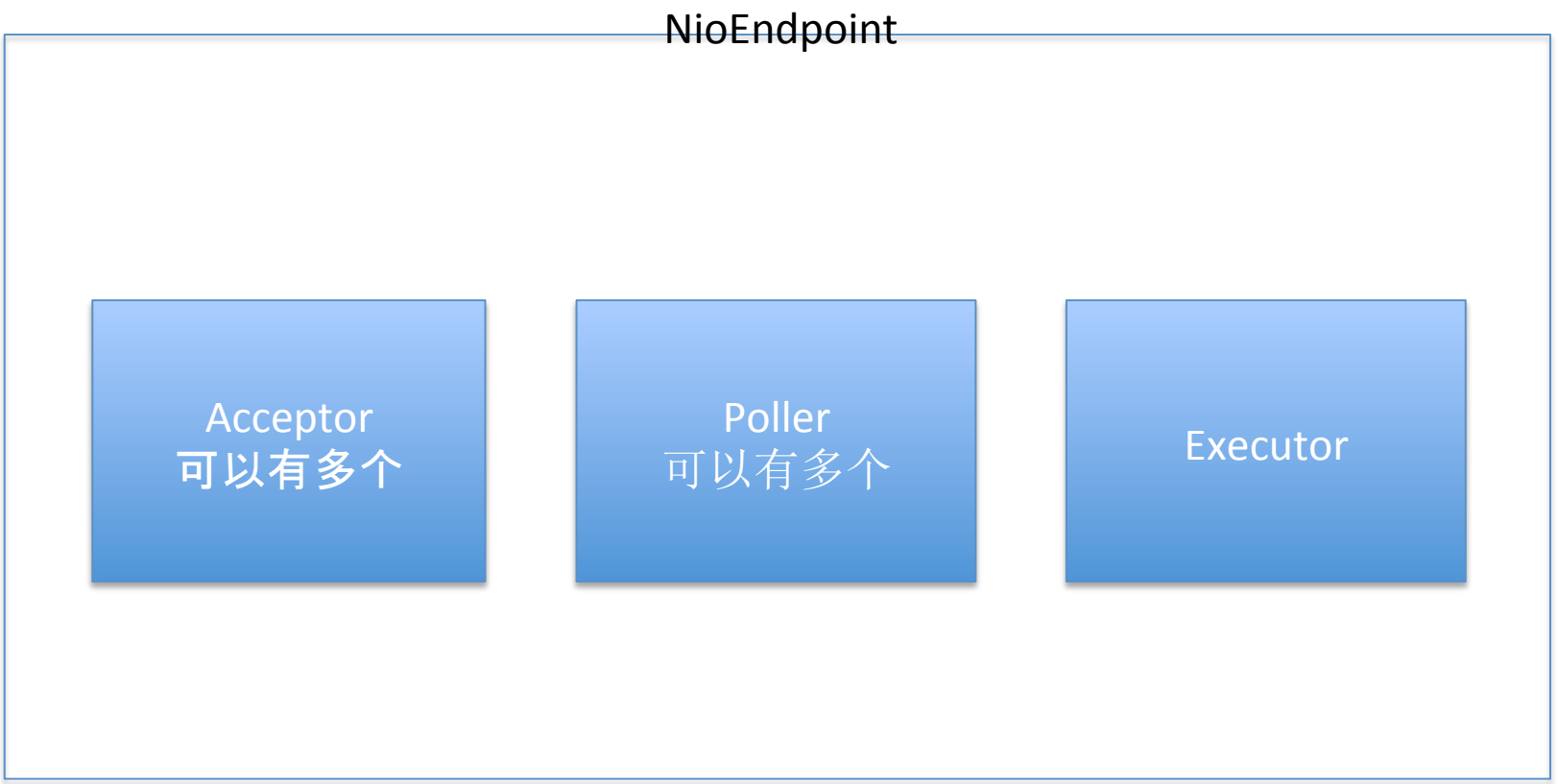
# Apache-Tomcat：Connector

- ProtocolHandler

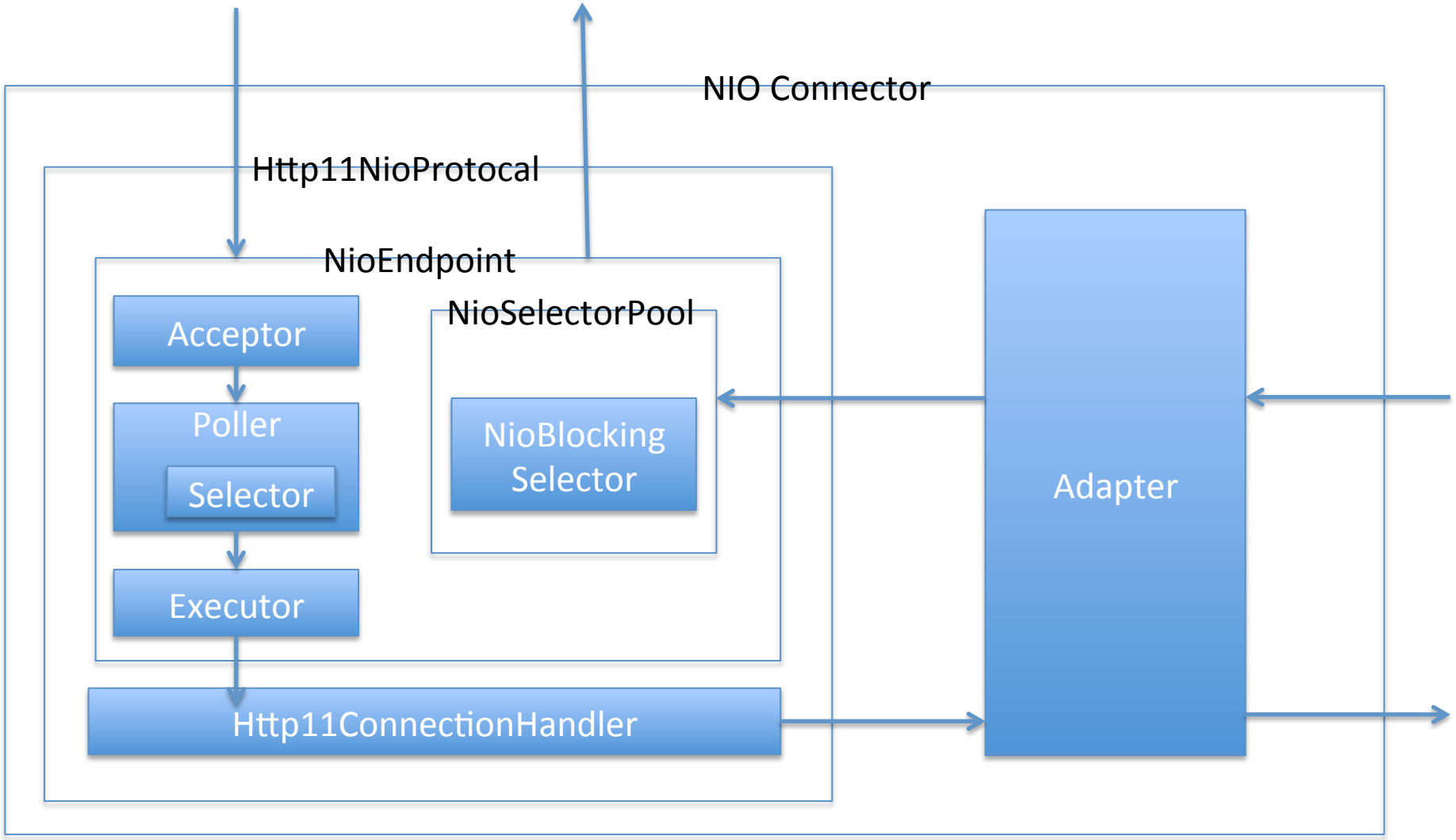


# Apache-Tomcat： Connector

- Endpoint的组成(NIO)

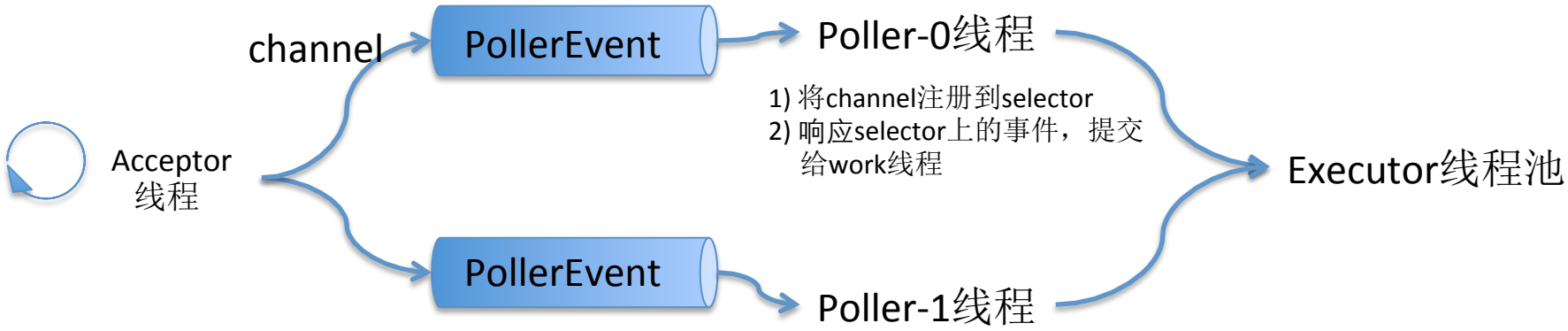


# Apache-Tomcat： Connector



# Apache-Tomcat： Connector

- NIO Connector的Request：



# Apache-Tomcat: Container

- 代号Catalina: 远程轰炸机

# Apache-Tomcat: Conatinaer

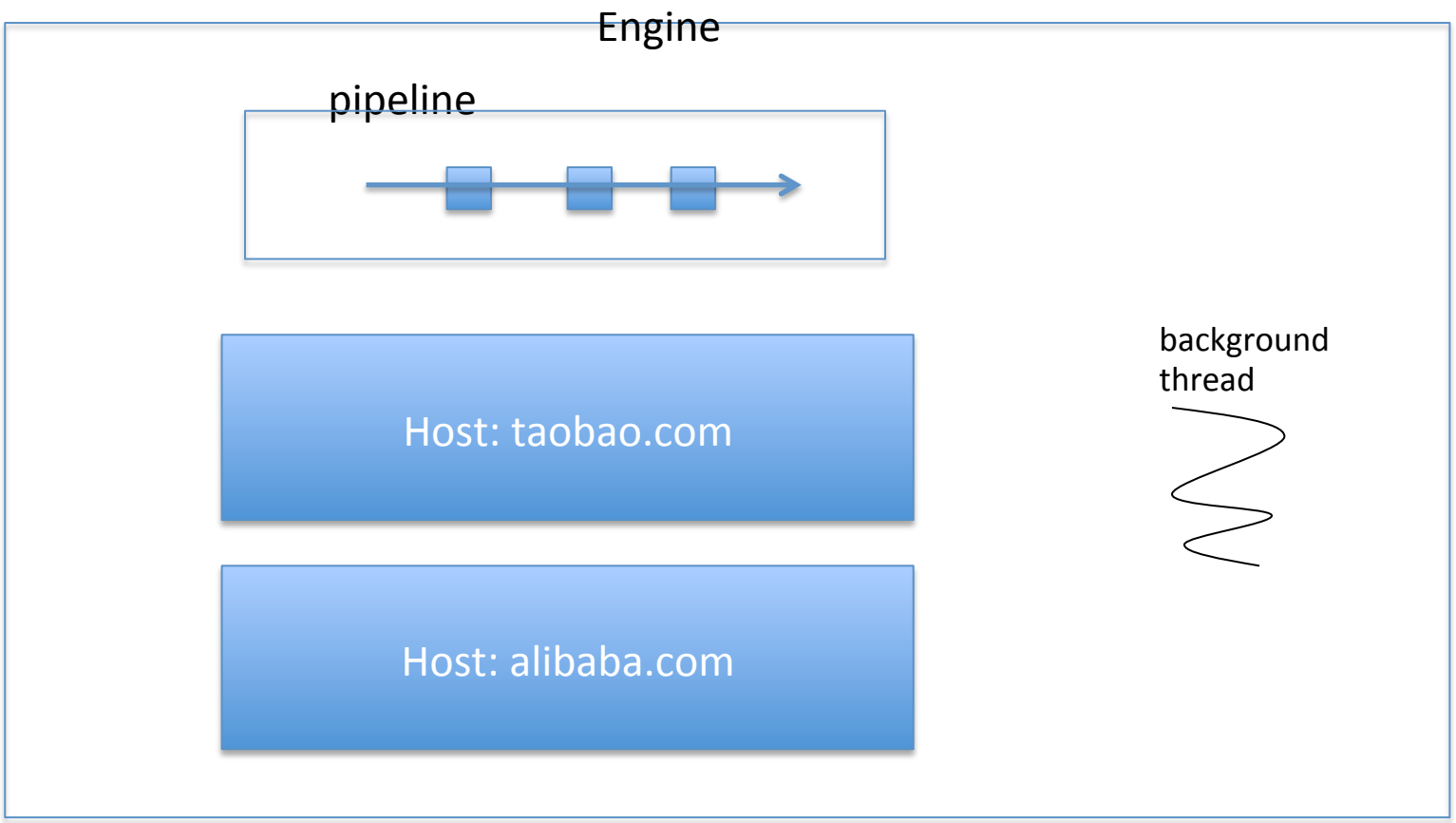
- Catalina的四重奏: Engine, Host, Context, Wrapper

# Apache-Tomcat: Container

- Engine：容器有时也被称为Servlet Engine

# Apache-Tomcat: Conatinaer

- Engine: 入口，内部可以包括多个Host

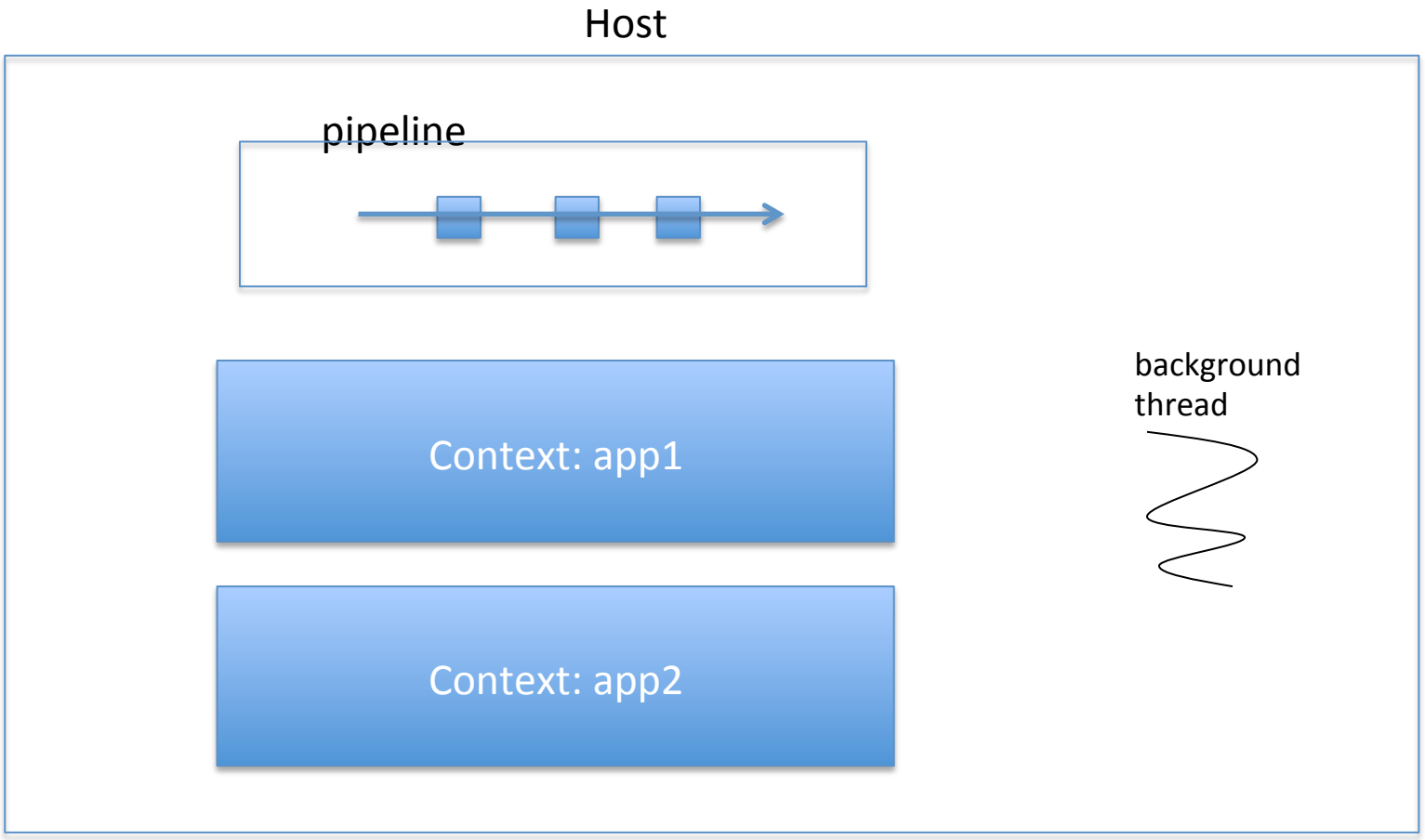


# Apache-Tomcat: Container

- Host: 虚拟主机

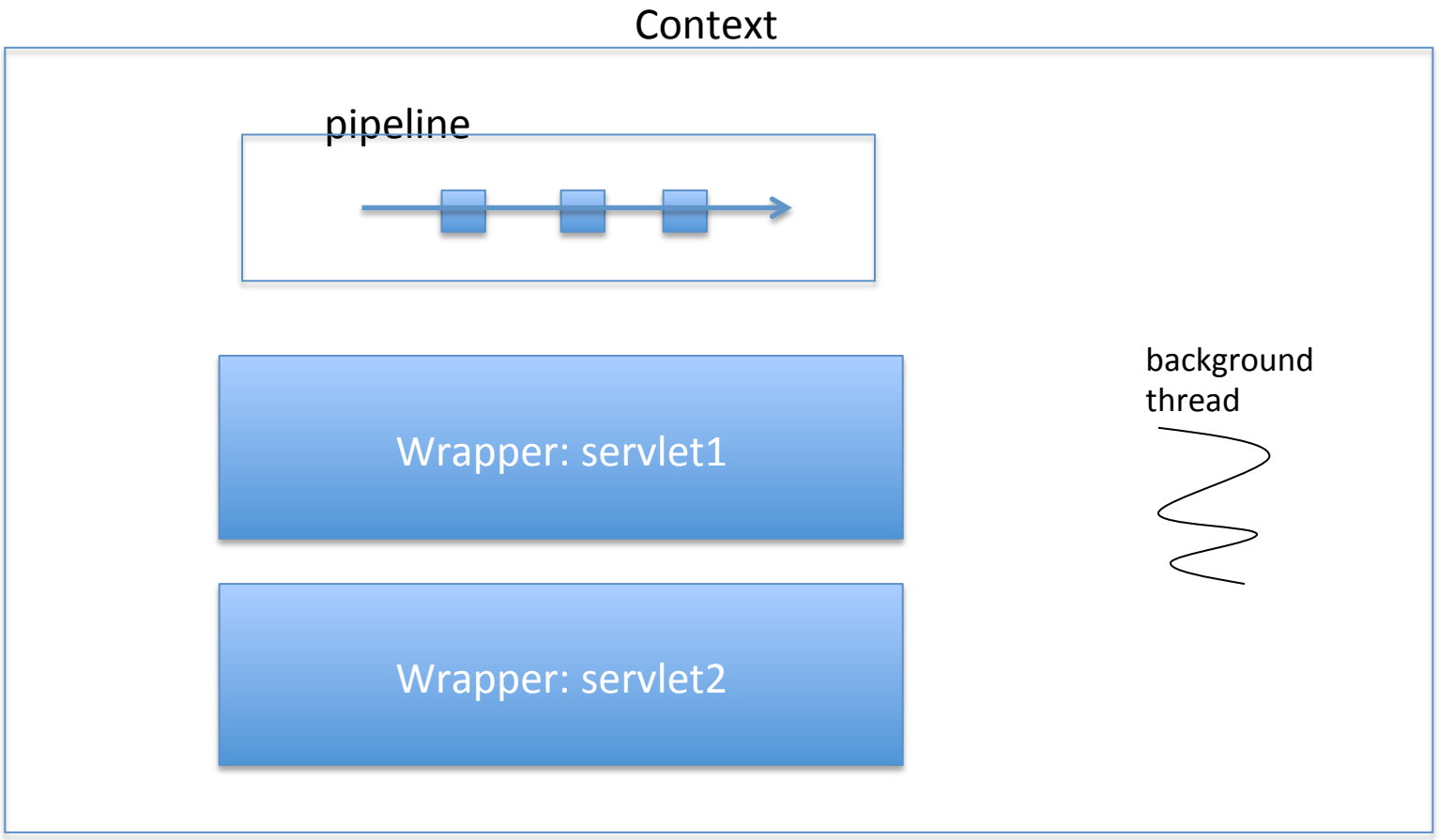
# Apache-Tomcat: Container

- Host: 部署应用



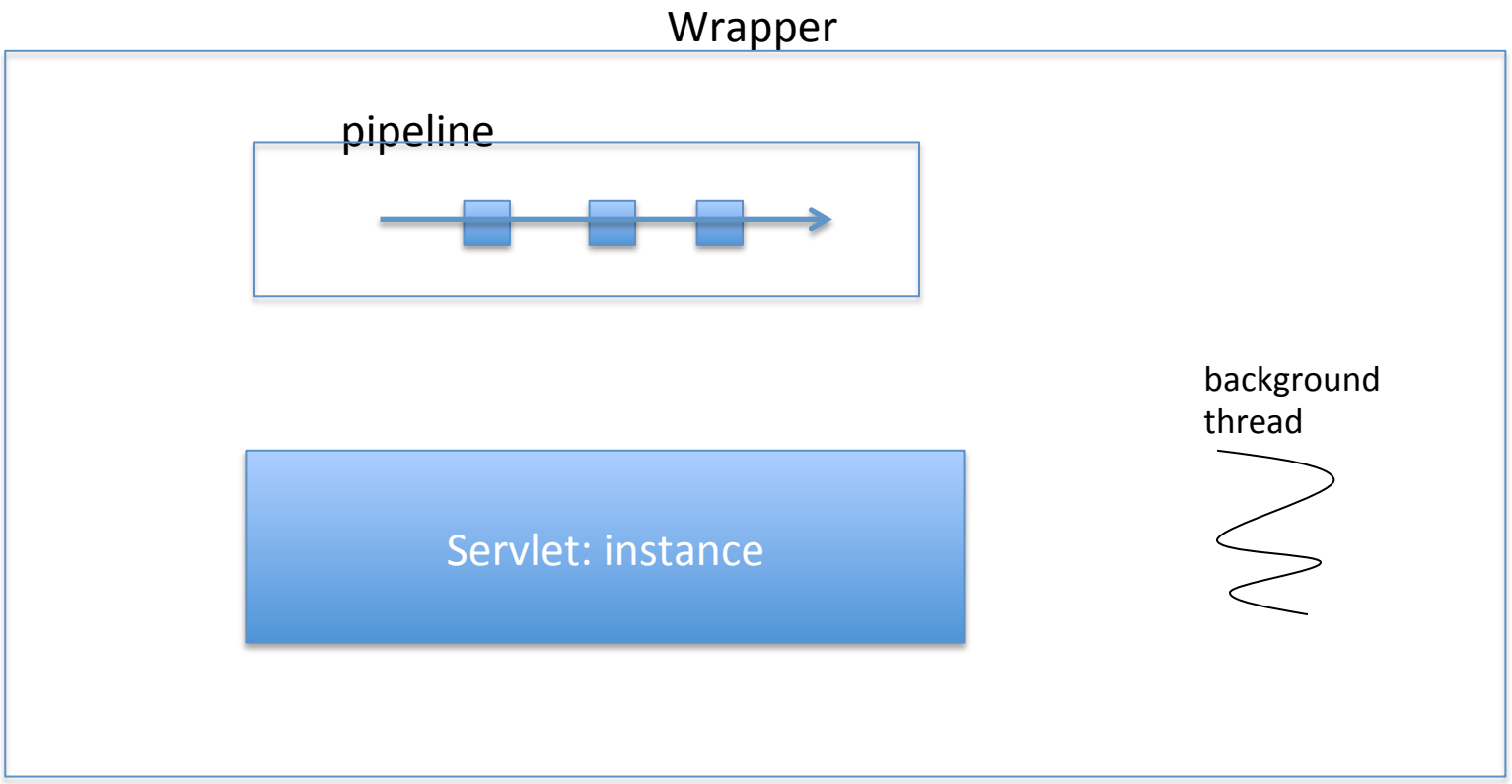
# Apache-Tomcat: Container

- Context: 应用上下文



# Apache-Tomcat: Container

- Wrapper: 对应规范里定义的Servlet；这层封装便于拦截和扩展



# Apache-Tomcat: Container

- 关于容器中的后台线程

1) Servlet规范定义了 App Reload：

*服务器应该能够更新一个新版本的应用程序，而无需重启容器。*

2) 并不是每个子容器都会启动后台线程，默认情况，只有Engine会启动后台线程(tomcat7)

# Apache-Tomcat: Container

- 关于容器中的后台线程

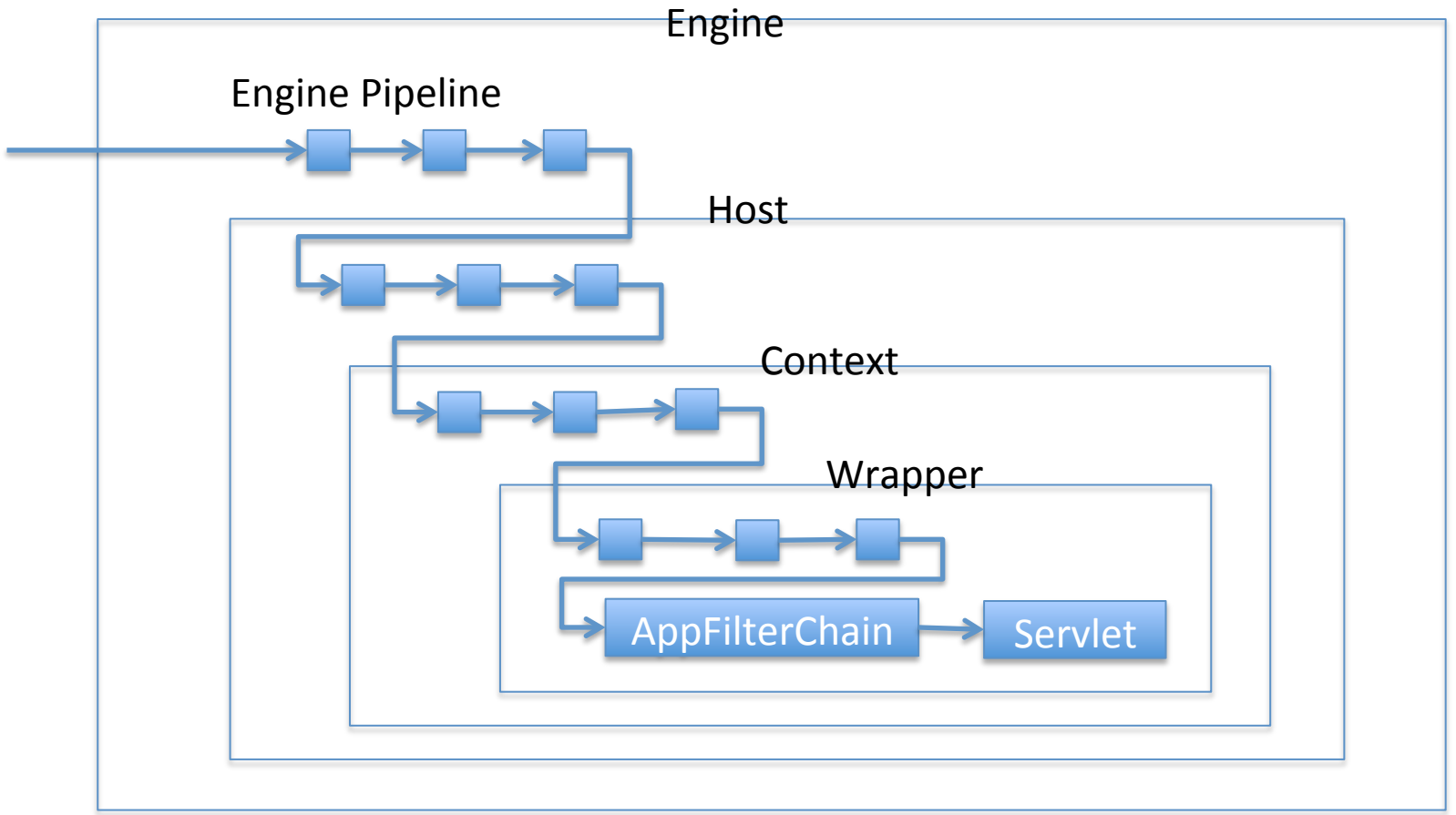
3) Reload的时候，做了什么？

stop appContext

start appContext

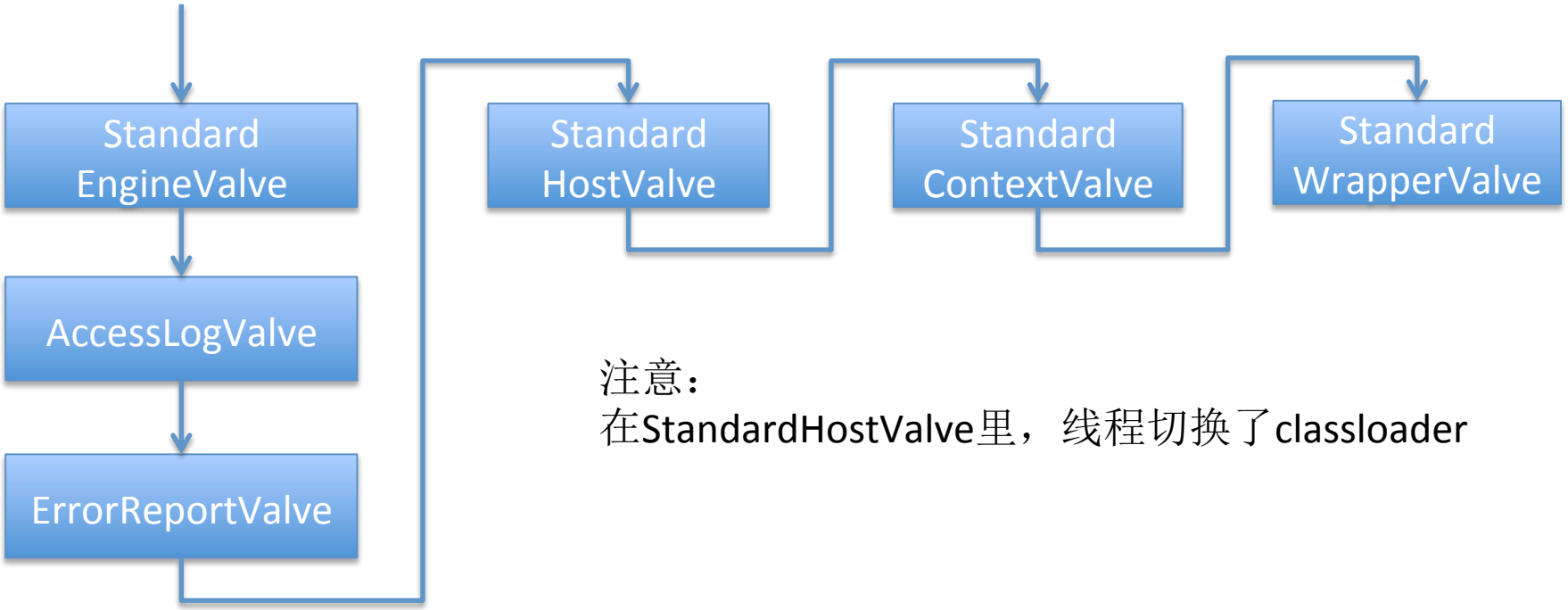
# Apache-Tomcat: Container

- Pipeline: 一次请求



# Apache-Tomcat: Container

- 真实世界中的Pipeline: 默认情况




注意：
在StandardHostValve里，线程切换了classloader

AccessLogValve 会把writer.flush交给容器background线程定期执行

# Apache-Tomcat

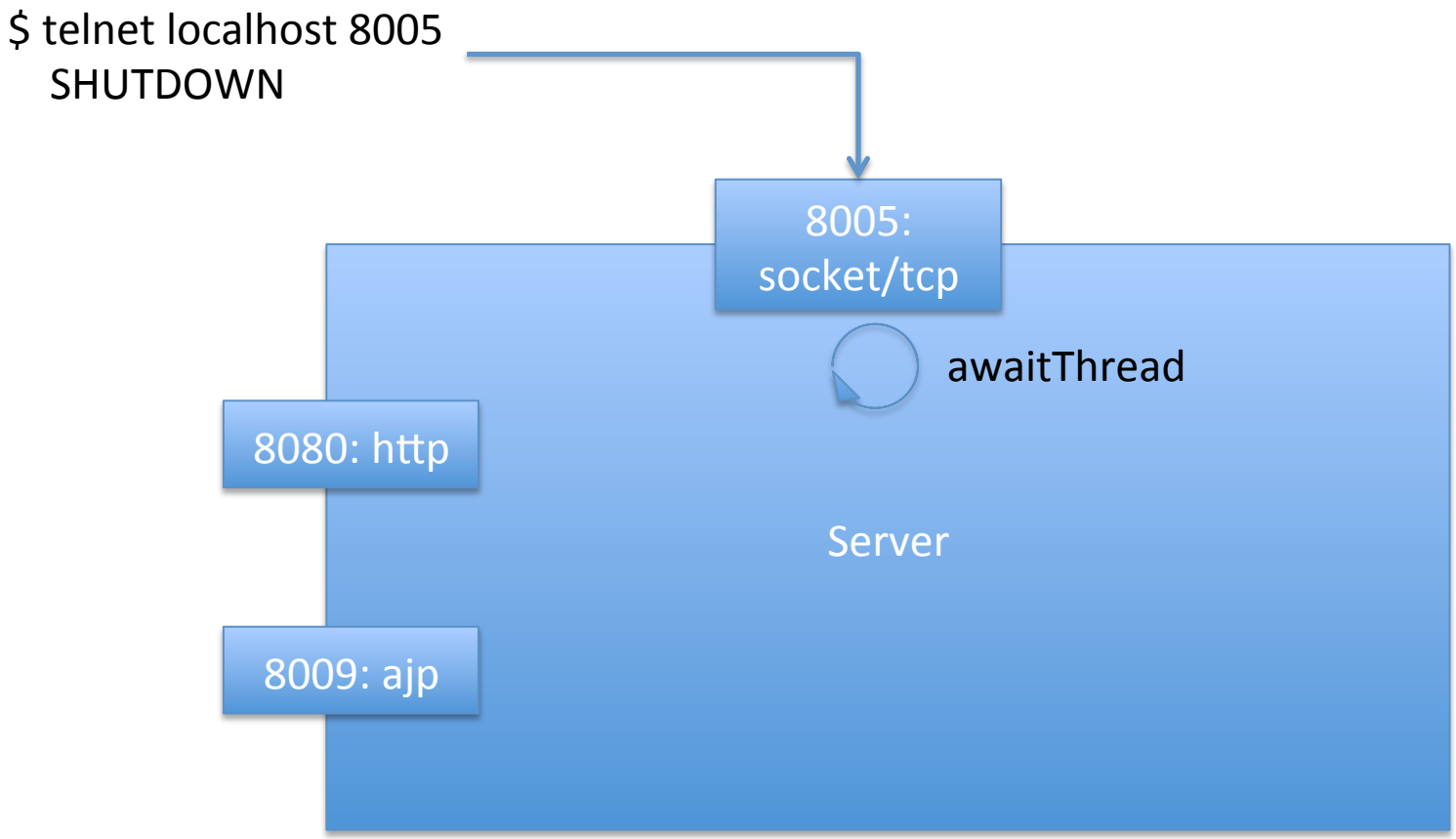
- Server的启动
  - 配置文件：server.xml
  - 组件解析和组装过程: digester

# Apache-Tomcat

- Server的关闭
  1) viaport：通过socket发送关闭指令
     注意避免重复执行shutdownHook

  2) 系统Signal: kill

# Apache-Tomcat




catalina.sh stop 是另外启动一个java进程，viaport方式

# Apache-Tomcat

- Server的停止过程

  1）pause connector

  2）stop container

  3）stop connector

# Apache-Tomcat

- App Context的停止过程

  1）停止所有子容器，也就是每个wrapper
  2）停止filter
  3）停止manager
  4）停止listener
  5）停止JNDI: namingResource (app级别的)
  6）停止pipeline
  7）停止resource
  8）停止realm
  9）停止cluster (集群)
  10）停止 loader (负载均衡)

# 一些零碎的小技巧(tips)

# Tips: 1

1) 调试tomcat:

```
$ cat debug.sh
#!/bin/sh
export JPDA_SUSPEND=y
./catalina.sh jpda run
```

# Tips: 2

2) 建议Servlet3的complete方法放到finally里

```
AsyncContext asyncContext = request.startAsync();
try{
    bizCode
}finally{
    aysncContext.complete();
}
```

# Tips: 3

3) 输出tomcat运行时的详细信息：

在  conf/logging.properties 添加:

.level = FINE

或只针对某个包：

org.apache.coyote.http11.level = FINE

# Tips: 4

4) 进程意外消失(非jvm crash)?

案例1：

Java进程启动时被传染了

ulimit –t 300

# Tips: 4

4) 进程意外消失(非jvm crash)?

案例2：ssh窗口关闭引发的问题

```
#!/bin/bash
cd /data/server/tomcat/bin/
./catalina.sh start
tail -f /data/server/tomcat/logs/catalina.out  #这句会有什么副作用?
```

http://hongjiang.info/why-kill-2-cannot-stop-tomcat/

http://hongjiang.info/shell-script-background-process-ignore-sigint/

# Tips: 5

5) 内存泄露的例子：

# Tips: 6

6) 每隔1小时full gc问题

由两种情况导致:

a) JreMemoryLeakPreventionLister

每小时执行一次System.gc

在 7028 和 6036 之后的版本里都已修复

b) rmi.dgc 的 gc间隔是1小时

# Tips: 7

7) 禁用DNS查询

修改server.xml文件中的enableLookups参数值为false

request.getRemoteHost() 只返回IP

# Tips: 8

8) 支持软连接

<Context allowLinking="true" />

把 classes 软连接到 eclipse 工程 bin目录

# Tips: 9

9) processorCache 和 socket.processorCache

默认

socket.processorCache 是500，
processorCache 是200

# Tips: 10

10) keepAlive ?

确认是否真的需要

BIO下当线程使用率超过75%，就取消了keepAlive

# 第二部分：Ali-Tomcat

国内规模最大的tomcat用户
可能也是世界规模最大的tomcat用户

# Ali-Tomcat: 使用情况

大部分BU都在用或准备用：

淘宝、天猫、阿里云、一淘、来往、无线
阿里妈妈、聚划算、AliExpress 等

# Ali-Tomcat: 使用情况

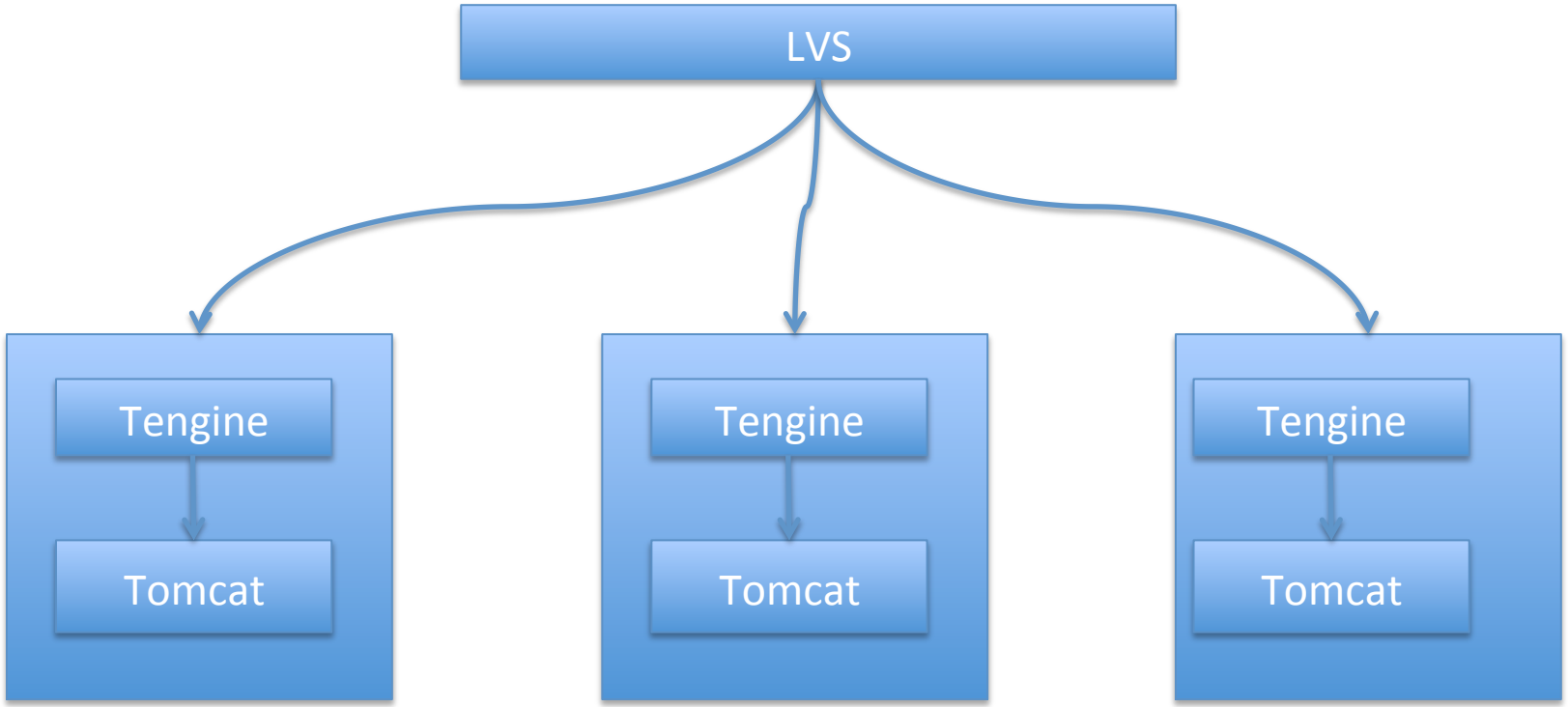
- 运维方式：

1) 开发与PE职责分离
  - 开发只负责产出应用打包文件，对容器配置不关心
  - 线上的运行环境全由PE操控

2) 发布
  - 应用打包为 appName.war
  - 应用所使用的中间件产品打包为 pandora.sar (原taobao-hsf.sar)
  - 发布系统保证pandora.sar 里的中间件产品为最新版本(兼容的前提下自动升级)

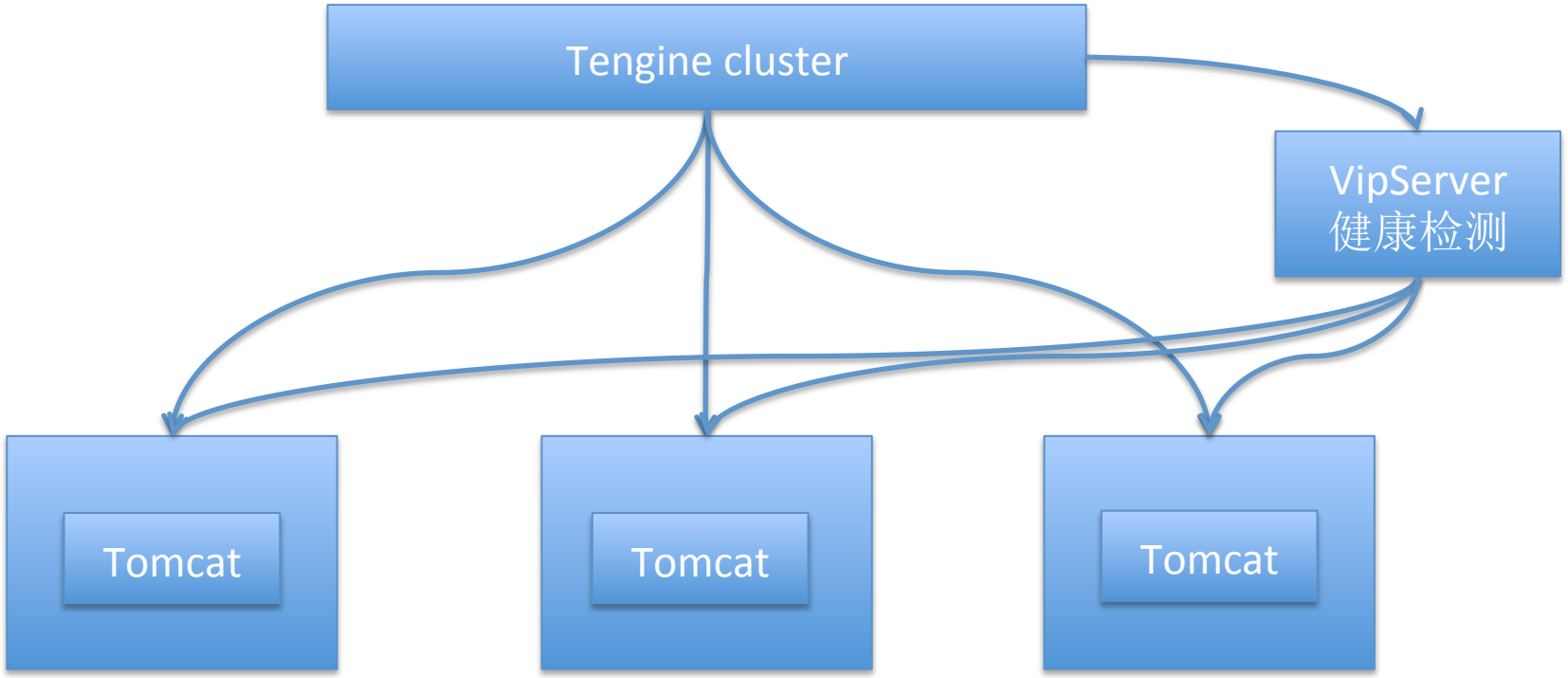
# Ali-Tomcat: 使用情况

- 部署方式1：



# Ali-Tomcat: 使用情况

- 部署方式2：



# Ali-Tomcat

- 基于Apache-Tomcat 7
  - JDK6、JDK7 (websocket)
- 为什么使用tomcat而不是jetty?
  - 对我们来说两者都够用
- 起初的核心需求：模块化(Pandora)
  - Pandora隔离了中间件(通用)产品与应用之间的jar依赖版本冲突，随发布自动升级中间件产品

# Ali-Tomcat: 改动

我们**裁减掉的**:

1) 集群
2) 多应用部署 (只支持单个应用)
3) **官方的**数据源

# Ali-Tomcat: 改动

我们**增加的**:

1) 模块化系统：Pandora
2) 监控系统: Monitor
3) 诊断工具：HouseMD2
4) 统一日志框架：Taobao-logback

# Ali-Tomcat: 改动

对 Connector 的bugfix（NIO下潜在的泄露）

1) ClosedByInterruptException:

- http://hongjiang.info/java-nio-channel-closed-by-interrupt-exception/
- 导致计数不能释放，达到maxConnections后acceptor不再接受请求

# Ali-Tomcat: Pandora

- 一个轻量级的模块化系统
- 隔离依赖，解决jar冲突问题
- 对所托管的模块提供查询和管理

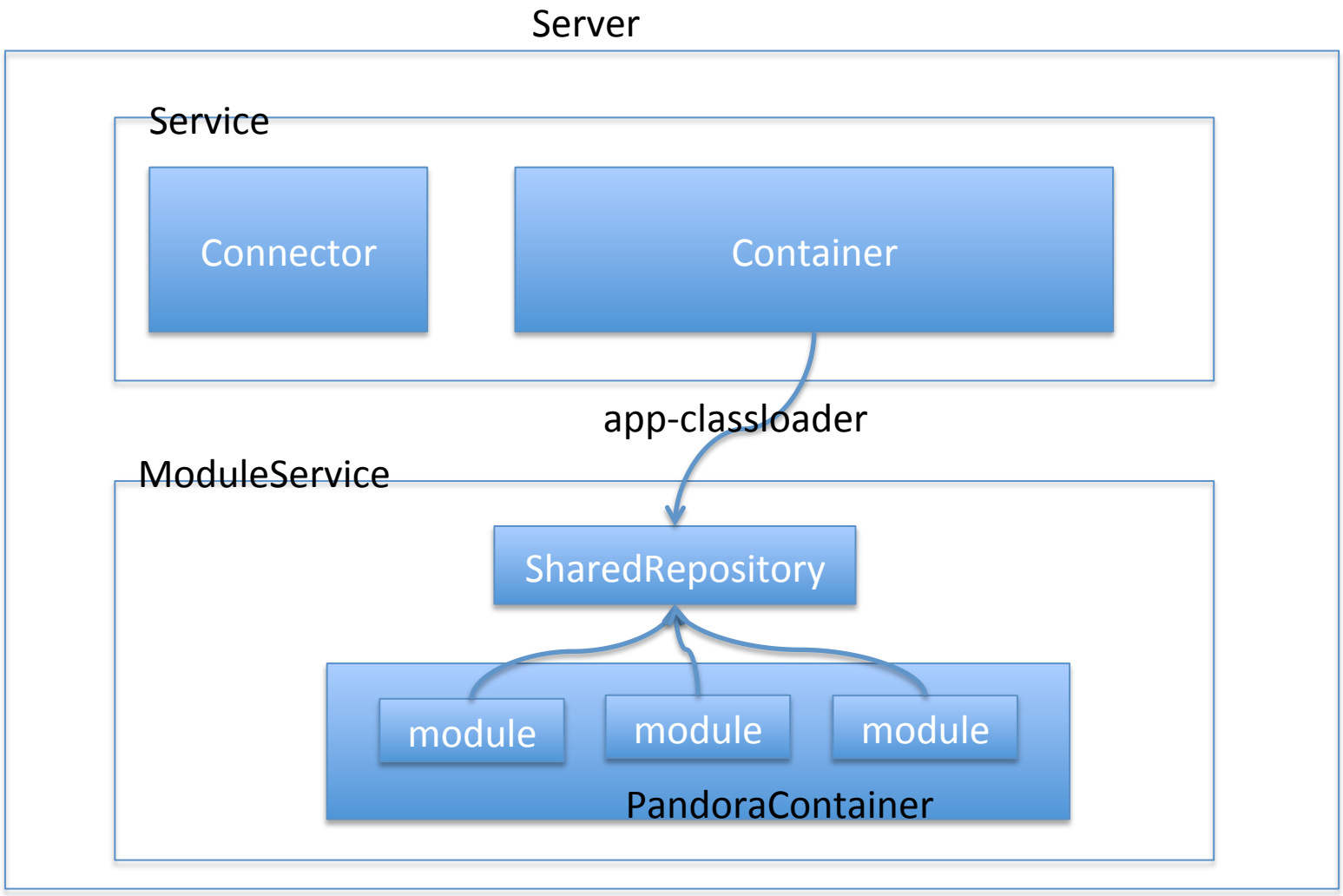
# Ali-Tomcat: Pandora

pandora.sar的结构：

```
├── META-INF
├── lib
└── plugins
    ├── monitor.jar.plugin
    ├── pandora.qos.service.jar.plugin
    ├── hsf.jar.plugin
    ├── notify.jar.plugin
    ├── eagleEye.jar.plugin
    ├── config-client.jar.plugin
```

每个plugin定义自己需要导出(**共享**)的

# Ali-Tomcat: Pandora



# Ali-Tomcat: Pandora

Pandora的QOS服务(telnet或http)

全局命令：

help ： 查询帮助信息
cd ： 更换到不同插件
ls ： 列出当前容器部署的所有插件
v ： 查询指定插件版本
find ： 查询指定Class的导出情况
source：查看一个类属于哪个jar,
被哪个ClassLoader加载
同时可以反编译.class文件成java源码

实现了Commandprovider接口的模块也可以提供自己的命令
比如 hsf 模块 online/offline

# Ali-Tomcat: Pandora

## QOS的source命令例子：

pandora> source com.taobao.hsf.logger.LoggerInit

location=/Users/zhuyong/pandora/plugins/hsf/lib/hsf.app.spring-2.1.0.3.jar!/com/taobao/hsf/logger/LoggerInit.class

classLoader=com.taobao.pandora.loader.PandoraModuleClassLoader,appname=hsf

sourceLoader=com.taobao.pandora.loader.PandoraModuleClassLoader,appname=pandora.qos.service

// 代码省略

# Ali-Tomcat: Pandora

QOS的source命令例子：

1) location代表该class所在的jar包位置；

2) classLoader代表真正加载该class的ClassLoader；

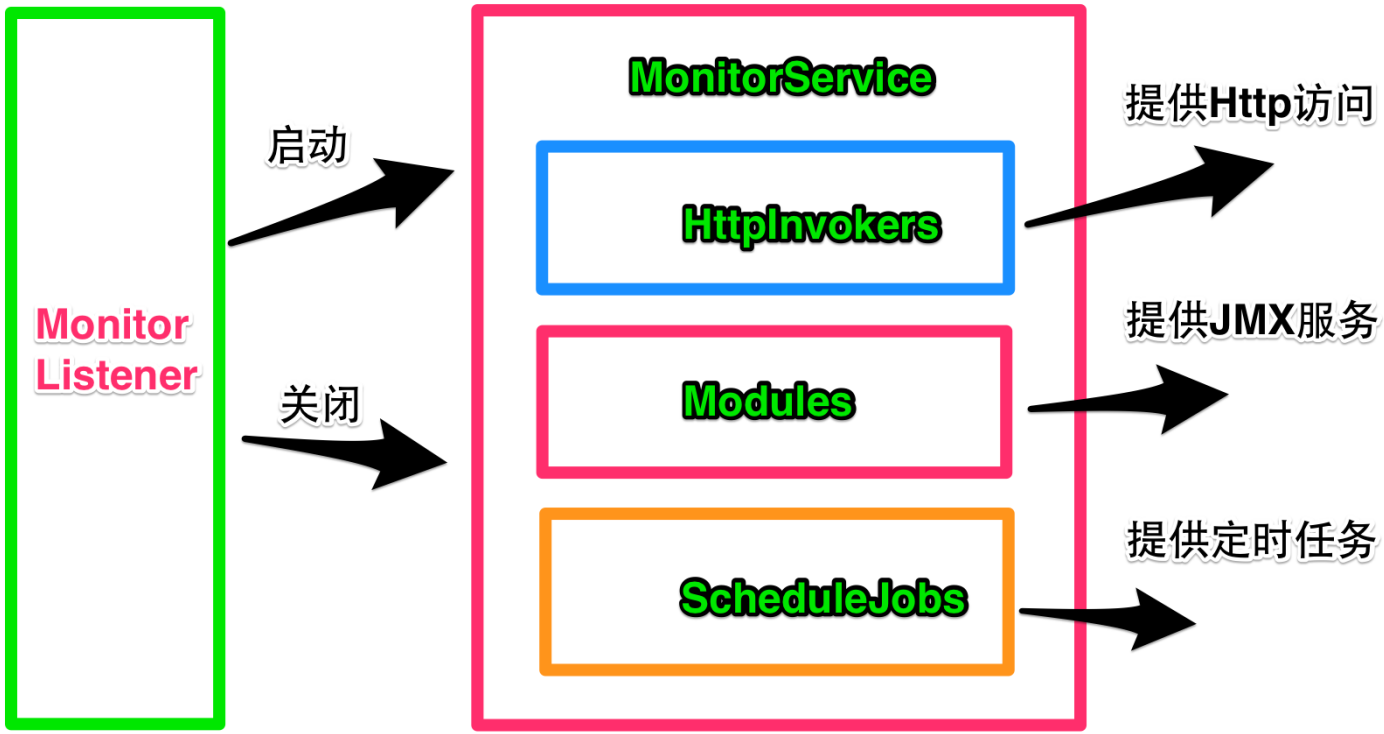
3) sourceLoader代表当前加载class所使用的ClassLoader；

4) 上例中，classLoader和sourceLoader不相同，则表示pandora.qos.service ClassLoader加载LoggerInit时，该类已经被hsf ClassLoader加载并导出了

# Ali-Tomcat: Pandora

- Pandora已不局限只应用在中间件产品
- 一些业务线也在使用pandora替代osgi作为它们的模块化解决方案

# Ali-Tomcat: 监控




使用方式：对Server添加 LifecycleListener

<Listener className="com.taobao.tomcat.monitor.MonitorServiceListener"/>

# Ali-Tomcat: 监控

有哪些可以监控的：curl http://localhost:8006/urls

| url | description |
| --- | --- |
| http://localhost:8006/memory/info | Statistics heap or non-heap memory usage. |
| http://localhost:8006/memory/pools | Output each java memory pools usage. |
| http://localhost:8006/classloader/jars | WebappClassLoader jar info. |
| http://localhost:8006/classloader/locate | Locate jar through class name or jar name. |
| http://localhost:8006/thread/list | List all thread info. |
| http://localhost:8006/thread/stack | Print thread detail info(stack and lock). |
| http://localhost:8006/thread/deadlock | Find deadlock and print thread detail info. |
| http://localhost:8006/thread/cpu | Thread own cpu percentage. |
| http://localhost:8006/thread/busy | Print busy thread info. |
| http://localhost:8006/connector/stats | Print connector global info. |
| http://localhost:8006/connector/threadpool | Print connector threadpool info. |
| http://localhost:8006/connector/top | Output connector current info. |
| http://localhost:8006/connector/slow | Print slow connection. |
| http://localhost:8006/tomcat/version | Print Tomcat Version info. |

# Ali-Tomcat: 监控

**10分钟内连接器中的处理器在运作中响应时间大于等于1ms**

curl http://localhost:8006/connector/slow?rt=1

Response time larger or equals than 1 millis in 10 minutes under below:

| start_time | url | max_time | connector |
|---|---|---|---|
| 15:27:53 | /slow.do | 30015 | "http-nio-8080" |
| 15:29:52 | /slow.do | 30007 | "http-nio-8080" |
| 15:27:23 | /slow.do | 30006 | "http-nio-8080" |
| 15:27:15 | /slow.do | 3005 | "http-nio-8080" |
| 15:49:54 | /slow.do | 3003 | "http-nio-8080" |
| 15:28:06 | /slow.do | 3003 | "http-nio-8080" |
| 15:30:18 | /slow.do | 3002 | "http-nio-8080" |
| 15:28:03 | /slow.do | 1002 | "http-nio-8080" |
| 15:30:21 | /favicon.ico | 3 | "http-nio-8080" |
| 15:27:55 | / | 2 | "http-nio-8080" |
| 14:49:21 | / | 1 | "http-nio-8080" |
| 14:50:21 | / | 1 | "http-nio-8080" |
| 15:15:20 | / | 1 | "http-nio-8080" |
| 15:15:21 | / | 1 | "http-nio-8080" |
| 15:16:21 | / | 1 | "http-nio-8080" |
| 15:17:21 | / | 1 | "http-nio-8080" |

# Ali-Tomcat: 监控

应用加载了哪些jar:
http://localhost:8006/classloader/jars

All jars can find:

| path | type |
|---|---|
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/sampleweb.war/WEB-INF/lib/commons-logging-1.1.1.jar | Webapp |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/sampleweb.war/WEB-INF/lib/spring-2.5.6.jar | Webapp |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/taobao-hsf.sar/lib/pandora.api-2.0.2.jar | Pandora |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/taobao-hsf.sar/lib/pandora.container-2.0.2.jar | Pandora |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/taobao-hsf.sar/lib/pandora.framework.api-2.0.2.jar | Pandora |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/taobao-hsf.sar/lib/pandora.framework.util-2.0.2.jar | Pandora |
| /Users/weipeng2k/Documents/workspace/taobao-tomcat/output/build/opt/taobao/tomcat/deploy/taobao-hsf.sar/lib/pandora.thirdcontainer-2.0.2.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/commons-lang-2.3.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/commons-pool-1.3.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/config-client-1.6.6-20131127.094141-2.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/config-common-2.0.2.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/gson-2.2.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/hessian-3.0.13.bugfix-20120711.024044-23.jar | Pandora |
| /Users/weipeng2k/pandora/plugins/config-client/lib/log4j-1.2.14.jar | Pandora |

# Ali-Tomcat: 监控

查询当前**JVM**中**10**秒内，占用**CPU**资源超过**0.3%**的所有线程
curl http://localhost:8006/thread/busy?percent=0.3

```
Busy thread info under below (Statistic Interval is 10s, In 10s and cpu using above 0.3%):
```

| thread_id | state | thread_name | cpu_percentage |
|---|---|---|---|
| 131 | WAITING | http-nio-8080-exec-3 | 0.3 |
| 129 | WAITING | http-nio-8080-exec-1 | 0.3 |
| 137 | WAITING | http-nio-8080-exec-9 | 0.3 |
| 53 | RUNNABLE | http-nio-8080-Acceptor-0 | 0.6 |
| 51 | RUNNABLE | http-nio-8080-ClientPoller-0 | 0.3 |
| 135 | WAITING | http-nio-8080-exec-7 | 0.3 |
| 52 | RUNNABLE | http-nio-8080-ClientPoller-1 | 0.3 |
| 133 | WAITING | http-nio-8080-exec-5 | 0.3 |

# Ali-Tomcat: 监控

查询当前**Tomcat**中所有连接器中正在工作中的处理器工作信息
curl http://localhost:8006/connector/top?alive=true

"http-nio-8080"

| remote_addr | url | query_string | method | process_time | thread_id | stage |
|---|---|---|---|---|---|---|
| 0:0:0:0:0:0:0:1 | /slow.do | s=3 | GET | 1002 | 137 | SERVICE |

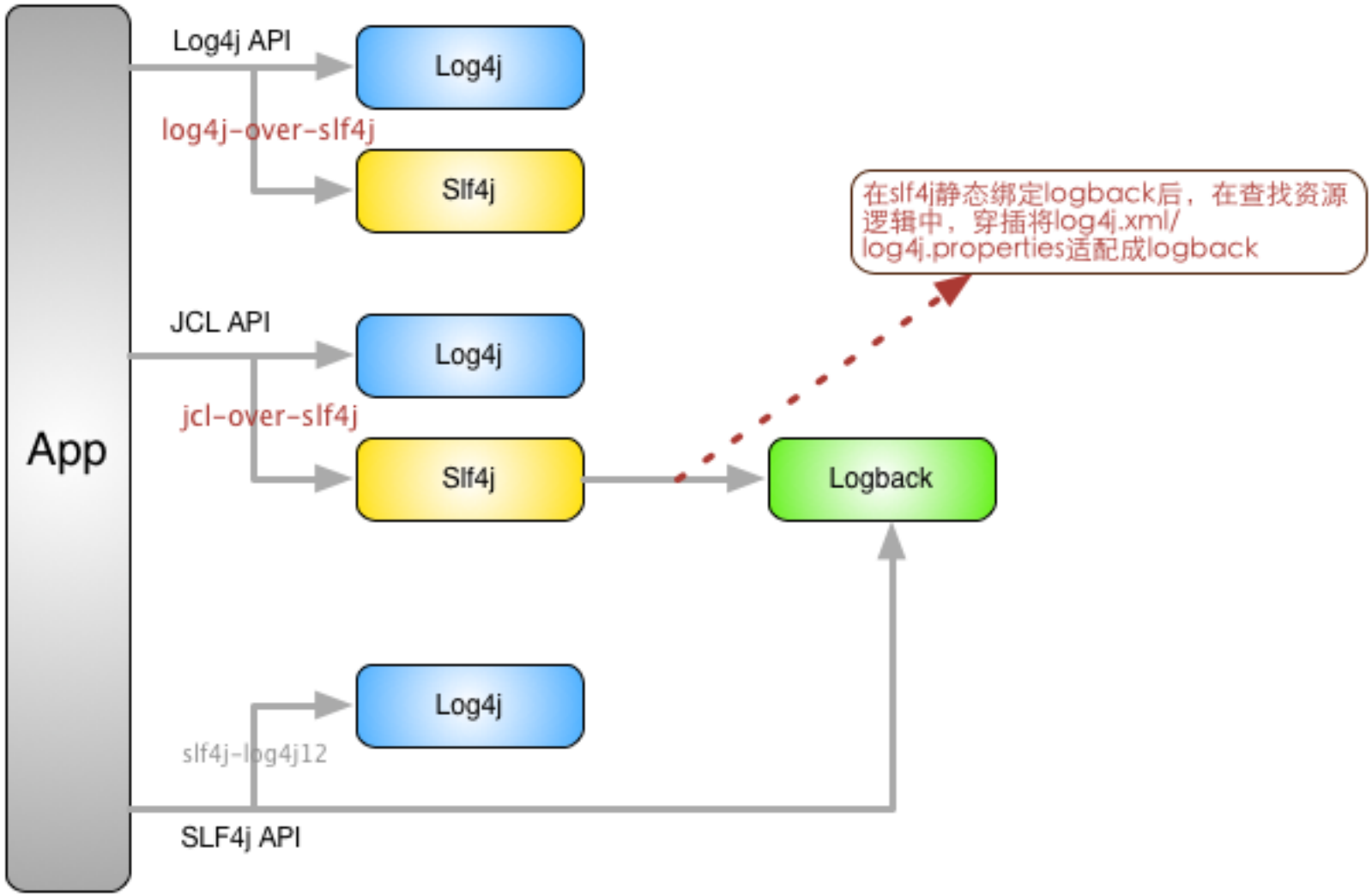
注：Monitor端口已和Pandora端口合并为一个

# Ali-Tomcat: 日志

统一日志框架：

- 目标是将app和中间件产品中所使用的各种日志框架在底层使用 logback 代理掉
- 起到收拢和控制的作用，提供了一些AOP扩展，比如为链路分析提供traceID

# Ali-Tomcat: 日志



# Ali-Tomcat: 日志

日志框架及日志系统搭配的主要使用场景分析

# Ali-Tomcat: 日志

统一日志框架托管在pandora中，除了核心功能，还提供了以下服务(QOS)：

1)日志查询
查询整个应用所有的日志内部细节，包括Level, Appender(Layout, RollingPolicy, Path)等

2)日志Level控制

3) log4j转logback配置及源配置信息查询

# Ali-Tomcat: 工具链

- 1) Maven plugin
  - 已提供
- 2) Eclipse/IDEA plugin
  - Eclipse**插件已提供**
  - IDEA插件尚未开发
- 3) Hotcode 热部署，JRebel的免费替代者
  - 开发中
- 4) HouseMD2 诊断工具，Btrace的替代者
  - 开发中

# Ali-Tomcat: 工程改动

- 对官方项目(单个工程)按模块进行拆分，并把构建工具ant改为maven

- 怎么避免合并新版本时的冲突？

  **先**用脚本将官方代码拆分成与我们目录结构一致，再进行代码合并

# Ali-Tomcat: 工程改动

- 把ant项目改为maven项目

目录结构：

```
├── bin
├── conf
├── deploy
├── lib
├── logs
├── package
├── pom.xml
├── res
├── rpm
├── server  // source code
├── temp
├── test
├── tools  // 诊断工具
└── work
```

# Ali-Tomcat: 工程改动

- Source Code

```
server
 ├── javax-mail-dummy
 ├── pom.xml
 ├── tomcat-annotations-api
 ├── tomcat-api
 ├── tomcat-bootstrap
 ├── tomcat-catalina
 ├── tomcat-catalina-ant
 ├── tomcat-coyote
 ├── tomcat-el-api
 ├── tomcat-extension
 ├── tomcat-jasper
 ├── tomcat-jasper-el
 ├── tomcat-jsp-api
 ├── tomcat-juli
 ├── tomcat-monitor
 ├── tomcat-servlet-api
 ├── tomcat-util
 ├── tomcat-websocket-api
 └── tomcat7-websocket
```

注：pandora和taobao-logback是独立项目，不包含在里面

即将开源

open source soon

# 我们在招人

- 联系： hongjiang.wanghj@alibaba-inc.com

# Q&A