


金蝶 Apusic 应用服务器嵌入式版 用户手册

法律声明

版权所有 © 深圳市金蝶天燕云计算股份有限公司2023。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明

和其他金蝶天燕商标均为深圳市金蝶天燕云计算股份有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

免责声明

本文档可能含有预测信息，包括但不限于有关未来的财务、运营、产品系列、新技术等信息。由于实践中存在很多不确定因素，可能导致实际结果与预测信息有很大的差别。因此，本文档信息仅供参考，不构成任何要约或承诺。金蝶天燕可能不经通知修改上述信息，恕不另行通知。

目录

1 前言

- 1.1 面向对象

2 产品简介

- 2.1 概述
- 2.2 JavaEE规范
- 2.3 平台环境
- 2.4 系统环境

3 产品清单

4 使用向导

- 4.1 安装
 - 4.1.1 基于源码方式构建
 - 4.1.1.1 导入Maven仓库
 - 4.1.1.1.1 前置条件
 - 4.1.1.1.2 安装资源依赖
 - 4.1.1.1.2.1 aams-spring-boot-starter
 - 4.1.1.1.2.2 aams-spring-boot-websocket
 - 4.1.1.1.3 引入资源依赖
 - 4.1.1.1.3.1 aams-spring-boot-starter
 - 4.1.1.1.3.2 aams-spring-boot-websocket
 - 4.1.1.2 基于可执行jar替换
 - 4.1.1.2.1 前置条件
 - 4.1.1.2.2 删除弃用依赖
 - 4.1.1.2.3 引入资源依赖

- 4.2 授权认证
 - 4.2.1 操作方法

5 功能说明

- 5.1 容器功能说明
 - 5.1.1 安全管理
 - 5.1.1.1 HTTPS
 - 5.1.1.1.1 简介
 - 5.1.1.1.2 配置方法
 - 5.1.1.2 国密配置
 - 5.1.1.2.1 简介
 - 5.1.1.2.2 配置方法
 - 5.1.1.2.2.1 注册国密实现jar包
 - 5.1.1.2.2.2 引入依赖
 - 5.1.1.2.2.3 配置文件
 - 5.1.1.2.2.4 启动应用
 - 5.1.1.2.2.5 访问方法
 - 5.1.1.2.2.6 其他问题
 - 5.1.2 国际化支持
 - 5.1.2.1 简介

- 5.1.2.2 配置方法
- 5.1.3 日志管理
 - 5.1.3.1 应用日志
 - 5.1.3.1.1 简介
 - 5.1.3.1.2 使用方法
 - 5.1.3.2 访问日志
 - 5.1.3.2.1 简介
 - 5.1.3.2.2 使用方法
- 5.1.4 监控支持
 - 5.1.4.1 简介
 - 5.1.4.2 使用方法
 - 5.1.4.2.1 springboot1.x 的版本
 - 5.1.4.2.1.1 开启监控
 - 5.1.4.2.1.2 查看监控
 - 5.1.4.2.1.3 访问目录描述
 - 5.1.4.2.2 springboot2.x 的版本
 - 5.1.4.2.2.1 开启监控
 - 5.1.4.2.2.2 查看监控
 - 5.1.4.2.2.3 访问目录描述

6 附录

- 6.1 附录A Web通用配置
- 6.2 附录B 日志参数
- 6.3 附录C actuator配置参数

1 前言

本文档为金蝶 Apusic 应用服务器嵌入式版 V10（简称 AAMS 嵌入式版 V10）使用说明，详细介绍金蝶 Apusic 应用服务器嵌入式版安装和相关配置方法。

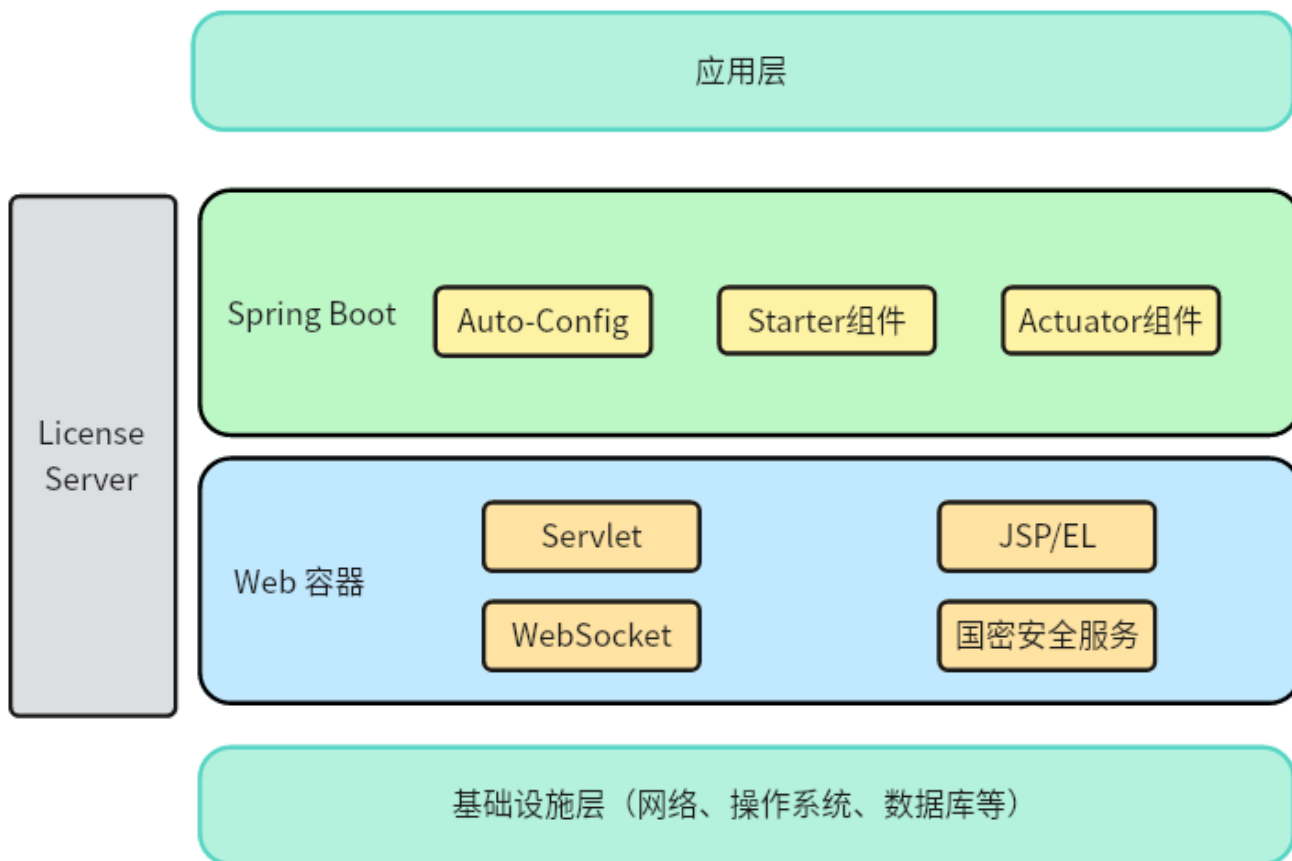
1.1 面向对象

本手册主要面向对象为使用金蝶 Apusic 应用服务器嵌入式版进行应用开发的开发人员，以及相关管理人员和运维人员。

2 产品简介

2.1 概述

金蝶 Apusic 应用服务器嵌入式版是一款同时支持JavaEE企业级标准与 React 响应编程标准的嵌入式应用服务器，集成Spring Boot框架使用。



2.2 JavaEE规范

规范/API	版本
Servlet	4.0
JSP	2.3
Expression Language	3.0
WebSocket	1.1

2.3 平台环境

操作系统	说明
Windows	Windows7、Windows10、Windows Server
Linux	CentOS、Ubuntu、SUSE、Redhat
国产平台	银河麒麟系列、中标麒麟系列、普华、中科红旗、深度等

2.4 系统环境

系统组件	系统要求
Java环境	JDK8及以上，针对HTTP2建议使用8U252以上
内存	至少需要 512MB 的内存，建议1024MB以上
磁盘空间	至少需要 1024MB 的磁盘空间，建议10G以上

3 产品清单

产品包	说明
aams-spring-boot-starter-all-[version].jar	AAMS starter
aams-spring-boot-websocket-starter-[version].jar	WebSocket资源包
aas-sm-[version].jar	国密资源包

4 使用向导

下面介绍 AAMS 作为一个完整的产品进行使用，\${AAMS_HOME} 表示产品安装的根目录。

4.1 安装

安装方式有两种：一种是基于应用系统源码重新构建可执行jar包，另一种是基于可执行jar包进行替换。

4.1.1 基于源码方式构建

4.1.1.1 导入Maven仓库

将金蝶 Apusic 应用服务器嵌入式版产品资源Jar包安装到本地Maven仓库。通过维护pom.xml文件管理 金蝶 Apusic 应用服务器嵌入式版产品资源Jar包依赖。

4.1.1.1.1 前置条件

需要提前准备Maven构建管理工具。

操作步骤：

1、下载并安装Maven管理工具

2、配置Maven环境变量

- Windows环境：

在“环境变量”的 Path 里添加 maven 的安装路径

- Linux环境：

在“/etc/profile”文件中添加Maven环境变量信息。

```
export MAVEN_HOME=Maven安装目录
export PATH=$PATH:$MAVEN_HOME/bin
```

配置完成后，保存，执行 source /etc/profile使配置生效

执行如下命令，检查Maven安装情况

```
mvn -v
```

有显示Maven信息，表示安装成功

```
Apache Maven 3.9.2 (c9616018c7a021c1c39be70fb2843d6f5f9b8a1c)
Maven home: /opt/maven/apache-maven-3.9.2
Java version: 1.8.0_361, vendor: Oracle Corporation, runtime: /opt/java/jdk1.8.0_361/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.4.240-1.el7.elrepo.x86_64", arch: "amd64", family: "unix"
```

4.1.1.1.2 安装资源依赖

将金蝶 Apusic 应用服务器嵌入式版产品资源Jar包安装到本地Maven仓库。

需要准备对应的产品资源jar包以及对应的pom.xml。

4.1.1.1.2.1 AAMS-SPRING-BOOT-STARTER

springboot1.x版本依赖安装：

```
mvn install:install-file -Dfile=aams-spring-boot-starter-all-1.5.20.RELEASE.jar -DgroupId=com.apusic
-DartifactId=aams-spring-boot-starter-all -Dversion=1.5.20.RELEASE -DpomFile=pom.xml -Dpackaging=jar
```

springboot2.0-springboot2.4版本依赖安装：

```
mvn install:install-file -Dfile=aams-spring-boot-starter-all-2.1.7.RELEASE.jar -DgroupId=com.apusic -DartifactId=aams-spring-boot-starter-all -Dversion=2.1.7.RELEASE -DpomFile=pom.xml -Dpackaging=jar
```

springboot2中2.4以上版本依赖安装：

```
mvn install:install-file -Dfile=aams-spring-boot-starter-all-2.4.0.jar -DgroupId=com.apusic -DartifactId=aams-spring-boot-starter-all -Dversion=2.4.0 -DpomFile=pom.xml -Dpackaging=jar
```

4.1.1.1.2.2 AAMS-SPRING-BOOT-WEBSOCKET

springboot1.x版本websocket依赖安装：

```
mvn install:install-file -Dfile=aams-spring-boot-websocket-starter-1.5.20.RELEASE.jar -DgroupId=com.apusic -DartifactId=aams-spring-boot-websocket-starter -Dversion=1.5.20.RELEASE -Dpackaging=jar
```

springboot2.x(包含2.4及以上版本)版本websocket依赖安装：

```
mvn install:install-file -Dfile=aams-spring-boot-websocket-starter-2.1.7.RELEASE.jar -DgroupId=com.apusic -DartifactId=aams-spring-boot-websocket-starter -Dversion=2.1.7.RELEASE -Dpackaging=jar
```

验证安装：

访问 maven 的本地仓库地址“[.../repository/com/apusic/](#)”，可以在对应仓库地址 中找到上述的 jar 包，此时即表示 jar 包部署到本地仓库成功。

4.1.1.1.3 引入资源依赖

4.1.1.1.3.1 AAMS-SPRING-BOOT-STARTER

使用AAMS作为运行容器，可以在pom.xml中添加aams-spring-boot-starter和AAMS的依赖，同时排除tomcat 的依赖。

1.由于 spring-boot-starter-web 插件默认带了 tomcat 的 starter，所以在 pom.xml 中需要排除 tomcat 的 starter，如下面的配置：

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
  <exclusions>
    <exclusion>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-tomcat</artifactId>
    </exclusion>
  </exclusions>
</dependency>
```

2.增加 aams 的 starter 依赖, 如 pom.xml 中的如下描述：

如果springboot版本是1.x版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

如果springboot版本是springboot2.0-sprigboot2.4版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
```

如果springboot版本是springboot2.4以上版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.4.0</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>2.4.0</version>
</dependency>
```

4.1.1.1.3.2 AAMS-SPRING-BOOT-WEBSOCKET

项目需要使用websocket，需要在pom.xml中增加aams-spring-boot-websocket的依赖。

如果springboot版本是1.x版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-websocket-starter</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

如果springboot版本是2.x版本（包含2.4及以上版本），pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-websocket-starter</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
```

4.1.2 基于可执行jar替换

若应用已经编译为可运行资源 jar 包，且采用的是 Spring Boot 内置的 Tomcat 服务器或其他应用服务器，需要把 Tomcat 嵌入式或其他应用服务器嵌入式的 jar 包删除，然后根据不同的 Spring Boot 版本把 AAMS 嵌入式所需的 jar 包添加进去。

4.1.2.1 前置条件

应用项目已开发完成，且编译为可运行资源jar包。

4.1.2.2 删除弃用依赖

以Tomcat嵌入式应用为例。

- 1.使用rar 或 7zip 等压缩工具打开已编译完成的 jar 包。
- 2.进入"BOOT-INF/lib" 目录，删除如下jar包。

```
tomcat-embed-core-x.x.x.jar
tomcat-embed-el-x.x.x.jar
tomcat-embed-websocket-x.x.x.jar
tomcat-embed-jasper-x.x.x.jar
spring-boot-starter-tomcat-x.x.x.jar
```

4.1.2.3 引入资源依赖

- 1.使用rar 或 7zip 等压缩工具打开已编译完成的 jar 包。
- 2.进入"BOOT-INF/lib" 目录，添加如下jar包。

如果springboot版本是1.x版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

如果springboot版本是springboot2.0-springboot2.4版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
```

如果springboot版本是springboot2.4及以上版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.4.0</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
```

```
<artifactId>spring-boot-starter-actuator</artifactId>
<version>2.4.0</version>
</dependency>
```

若需要用到 jsp 或 el 规范，则需要放入如下 jar 包：

```
mvn install:install-file -Dfile=aas-embed-el.jar -DgroupId=com.apusic.ams.embed -DartifactId=aas-embed-el -Dversion=10.1 -Dpackaging=jar
mvn install:install-file -Dfile=aas-embed-websocket.jar -DgroupId=com.apusic.ams.embed -DartifactId=aas-embed-websocket -Dversion=10.1 -Dpackaging=jar
mvn install:install-file -Dfile=aas-embed-jasper.jar -DgroupId=com.apusic.ams.embed -DartifactId=aas-embed-jasper -Dversion=10.1 -Dpackaging=jar

<dependency>
  <groupId>com.apusic.ams.embed</groupId>
  <artifactId>aas-embed-jasper</artifactId>
  <version>10.1</version>
</dependency>
<dependency>
  <groupId>com.apusic.ams.embed</groupId>
  <artifactId>aas-embed-el</artifactId>
  <version>10.1</version>
</dependency>
```

若原项目中用到了 websocket 规范，在删除了包tomcat-embed-websocket-x.x.x.jar 之后，还需要放入如下 jar 包：

springboot1.x版本依赖引入：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aas-spring-boot-websocket-starter</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

springboot2.x(包含2.4及以上版本)版本依赖引入：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aas-spring-boot-websocket-starter</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
```

4.2 授权认证

金蝶 Apusic 应用服务器嵌入式版启动需要依赖授权文件进行校验，可联系金蝶天燕销售人员申领授权文件（license.xml）。

4.2.1 操作方法

支持通过 java -D 命令行参数加载配置：

```
-Dlicpath=xx/license.xml
```

5 功能说明

5.1 容器功能说明

5.1.1 安全管理

介绍安全管理相关配置。

5.1.1.1 HTTPS

5.1.1.1.1 简介

HTTPS（全称：Hypertext Transfer Protocol Secure），是由 HTTP 加上 TLS/SSL 协议构建的可进行加密传输、身份认证的网络协议，可以有效提高网站安全性。金蝶 Apusic 应用服务器嵌入式版支持 HTTPS。

5.1.1.1.2 配置方法

使用 HTTPS 时，可以通过 Spring Boot 配置加载对应 application.properties、application.yml 等文件加载配置项。

application.properties 配置如下：

```
#https配置
#应用服务器监听端口
server.port=8089
#是否开启SSL协议
server.ssl.enabled=true
#配置SSL证书目录
server.ssl.key-store=classpath:keystore.jks
#SSL证书类型
server.ssl.key-store-type=JKS
#私钥密码
server.ssl.key-password=xxxxxxx
#密钥库文件密码
server.ssl.key-store-password=xxxxxxx
#服务器使用的密码和证书在密钥库的别名
server.ssl.key-alias=apusicuserkey
#客户端证书验证，值为 need,want,none 之一
server.ssl.client-auth=none
#信任库文件，用于验证客户端证书
server.ssl.trust-store=classpath:truststore.jks
#信任库密码
server.ssl.trust-store-password=xxxxxxx
#信任库类型
server.ssl.trust-store-type=JKS
#允许使用的密码套件
server.ssl.ciphers=*
```

5.1.1.2 国密配置

5.1.1.2.1 简介

金蝶 Apusic 应用服务器嵌入式版支持国密 SSL 协议通信，支持国家密码局认定的国产密码算法。

5.1.1.2.2 配置方法

5.1.1.2.2.1 注册国密实现 JAR 包

将 AAMS 的国密实现 Jar 包 -\${AAMS_HOME}/lib/aas-sm-3.0.jar 注册到 Maven 库中，以备应用打包引用。

```
mvn install:install-file -Dfile=aas-sm-3.0.jar -DgroupId=com.apusic.ams.embed -DartifactId=aas-sm -
Dversion=3.0 -Dpackaging=jar
```

5.1.1.2.2.2 引入依赖

在 pom.xml 文件中添加国密实现依赖。

```
<dependency>
  <groupId>com.apusic.ams.embed</groupId>
  <artifactId>aas-sm</artifactId>
  <version>3.0</version>
</dependency>
```

5.1.1.2.2.3 配置文件

引入依赖之后，需要在项目的 application.properties或application.yml添加国密SSL配置。

application.properties配置如下：

```
#国密配置：
#表示SSL通信协议使用国密通信协议
server.ssl.protocol=SMv1.1
#国密证书私钥类型
server.ssl.key-store-type=PKCS12
#配置国密密钥库文件
server.ssl.key-store=classpath:keystore.p12
#私钥密码
server.ssl.key-password=xxxxxxx
#密钥库文件密码
server.ssl.key-store-password=xxxxxxx
```

5.1.1.2.2.4 启动应用

Springboot 应用中使用国密需要在应用启动中添加 `-Dcom.apusic.security.ssl.EnableGMTLS=true` `-Djava.security.egd=file:/dev/./urandom` 参数，例如：

```
java -Dcom.apusic.security.ssl.EnableGMTLS=true -Djava.security.egd=file:/dev/./urandom -jar
springboot-gm-demo.jar
```

5.1.1.2.2.5 访问方法

项目配置国密SSL协议之后，需要使用支持国密SSL协议的浏览器访问，例如360安全浏览器、奇安信等。浏览器需要开启国密通信功能。

以360安全浏览器为例，进入“设置”-“选项”-“安全设置”，勾选“国密通信协议”，设置“国密SSL通讯白名单”，重启浏览器。

选项

基本设置

界面设置

标签设置

鼠标手势

快捷键

高级设置

实验室

安全设置

核心安全防护

- 多层系统防护 文件，驱动，进程，注册表多层防护，保护系统安全
- 沙箱防护 页面沙箱防护，防止未知病毒侵害
- 挖矿防护 禁止执行数字货币挖矿脚本，防止硬件资源被利用
- 隔离防护 网页，代码，插件，CPU进程隔离
- 系统级防注入 防恶意代码注入，保护浏览器安全

隐私安全设置

清理上网痕迹设置...

管理HTTPS/SSL证书...

- 检查服务器证书吊销状态
- 拦截证书风险
- 开启安全键盘功能，输入密码时使用虚拟键盘，可防止密码被键盘记录器窃取
- 自动停用来源不明的扩展
- 开启“禁止跟踪(DNT)”功能

国密通信协议

- 启用国密SSL协议支持

国密SSL通讯白名单

HTTPS协议优先走国密SSL协议

根证书管理

信任的根证书

5.1.1.2.2.6 其他问题

如果使用 Oracle JDK，由于 JVM 默认的加密字节长度限制，需要将 lib/jce_policy-8.zip 解压，并替换 %JAVA_HOME%\jre\lib\security 文件夹下相应 jar 文件，解放字节限制。

5.1.2 国际化支持

5.1.2.1 简介

为方便不同区域用户使用金蝶 Apusic 应用服务器嵌入式版支持，金蝶 Apusic 应用服务器嵌入式版提供国际化支持。

5.1.2.2 配置方法

默认日志输出采用的是英文格式输出。如需改成中文格式需要在 spring boot 的配置文件中编写代码定义日志输出格式配置，如下所示。

```
@Bean
public Locale locale(){
    return Locale.CHINA;
}
```

5.1.3 日志管理

5.1.3.1 应用日志

5.1.3.1.1 简介

金蝶 Apusic 应用服务器嵌入式版支持日志功能，支持集成Spring Boot方法实现。

5.1.3.1.2 使用方法

可以通过Spring Boot 配置加载对应 application.properties或application.yml 配置文件，配置Spring Boot 标准日志配置信息。

application.properties配置如下：

```
#日志配置
#关闭日志输出
#logging.level.root: OFF
#日志级别
logging.level.root=info
#使用相对路径的方式设置日志输出的位置
#logging.file.path=logs
#控制台日志输出格式
logging.pattern.console=%d{yyyy-MM-dd hh:mm:ss} [%thread] %-5level %logger{50} - %msg%n
#日志文件输出格式
logging.pattern.file=%d{yyyy-MM-dd} === [%thread] === %-5level === %logger{50} === - %msg%n
#定义日志文件名称
logging.file.name=logs/aams-log-test.log
```

5.1.3.2 访问日志

5.1.3.2.1 简介

金蝶 Apusic 应用服务器嵌入式版支持访问日志记录输出能力，集成Spring Boot方法配置对接。

5.1.3.2.2 使用方法

可以通过Spring Boot 配置加载对应 application.properties或application.yml 配置文件，配置日志信息。

application.properties配置如下：

```
server.aas.basedir=./
server.aas.accesslog.enabled=true
```

5.1.4 监控支持

5.1.4.1 简介

金蝶 Apusic 应用服务器嵌入式版提供监控功能，自带了springboot actuator 框架，监控对应资源。

5.1.4.2 使用方法

直接引入aams-spring-boot-starter-x.x.RELEASE.jar

5.1.4.2.1 springboot1.x 的版本

在项目 pom.xml 中引入 aams-spring-boot-starter的依赖。

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>1.5.20.RELEASE</version>
</dependency>
```

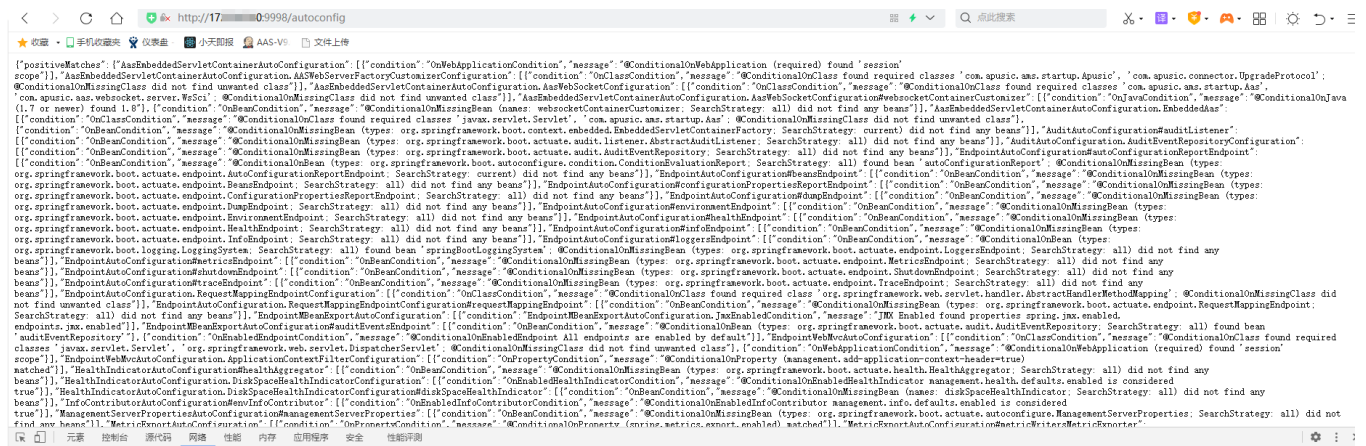
5.1.4.2.1.1 开启监控

在 spring boot 的配置文件中配置开启监控，在 application.properties 配置如下。

```
#配置端口
management.port=9099
#配置监控访问上下文
#management.context-path=/actuator
#设置是否开启安全性
management.security.enabled=false
```

5.1.4.2.1.2 查看监控

启动服务，格式为 http://[IP]:[management.port]/[management.context-path]/[访问目录项]。如访问 http://[IP]:[management.port]/autoconfig。



5.1.4.2.1.3 访问目录描述

路径	说明
/configprogs	描述配置属性如何注入 Bean
/beans	描述配置应用程序上下文里全部 Bean 以及其关系
/caches	获取缓存信息
/autoconfig	提供自动配置报告，记录自动配置通过情况
/env	获取全部环境属性
/health	报告应用程序的监控指标
/loggers	获取 loggers
/dump	获取线程活动快照
/metrics	报告各种应用程序度量信息，比如内存用量和 HTTP 请求计数
/trace	提供基本的 http 请求跟踪信息
/mapping	报告全部 URL 路径

5.1.4.2.2 springboot2.x 的版本

在项目 pom.xml 中引入 aams-spring-boot-starter 的依赖。

如果 springboot 版本是 springboot2.x-springboot2.4 版本，则 pom 文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>2.1.7.RELEASE</version>
</dependency>
```

如果springboot版本是sprigboot2.4版本，则pom文件中引入如下内容：

```
<dependency>
  <groupId>com.apusic</groupId>
  <artifactId>aams-spring-boot-starter-all</artifactId>
  <version>2.4.0</version>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
  <version>2.4.0</version>
</dependency>
```

5.1.4.2.2.1 开启监控

在 spring boot 的配置文件中配置开启监控，在application.properties 配置如下。

```
#actuator端口 如果不配置做默认使用上面server.port端口
management.server.port=9999
#默认值访问health,info端点,用*可以包含全部端点
management.endpoints.web.exposure.include=*
#修改访问路径
management.endpoints.web.base-path=/actuator
```

5.1.4.2.2.2 查看监控

启动服务，格式为 http://[IP]:[management.server.port]/[management.endpoints.web.base-path]。如访问 http://[IP]:[management.port]/actuator。



5.1.4.2.2.3 访问目录描述

路径	说明
/actuator	actuator服务访问上下文
/configprogs	描述配置属性如何注入Bean
/beans	描述配置应用程序上下文里全部Bean以及其关系
/caches	获取缓存信息
/conditions	提供自动配置不给，记录自动配置通过情况

/env	获取全部环境属性
/health	报告应用程序的监控指标
/loggers	获取loggers
/heapdump	获取heapdump
/metrics	报告各种应用程序度量信息，比如内存用量和HTTP请求计数
/threaddump	获取threaddump
/httptrace	提供基本的http请求跟踪信息
/mapping	报告全部URL路径
/scheduledtasks	获取计划任务

6 附录

6.1 附录A Web通用配置

参数	说明	默认值
server.port	应用服务器监听端口	8080
server.address	应用服务器监听地址	无
server.server-header	设置 HTTP 响应头 Server 的值	无
server.max-http-header-size	设置最大 HTTP 消息头大小	8192
server.connection-timeout	连接超时时间, 读取 HTTP 请求行超时时间	20000ms
server.ssl.enabled	是否允许 SSL 通信	true
server.ssl.ciphers	允许使用的密码套件	无
server.ssl.client-auth	客户端证书验证, 值为 need,want,none 之一	无
server.ssl.enabled-protocols	允许与客户端通信的协议名字	无
server.ssl.key-alias	服务器使用的密码和证书在密钥库的别名	无
server.ssl.key-password	密钥库密码	无
server.ssl.key-store	密钥库文件	无
server.ssl.key-store-password	密钥库文件密码	无
server.ssl.key-store-type	密钥库文件类型	无
server.ssl.protocol	服务器使用的 SSL 协议, 用于创建 SSLContext。	TLS
server.ssl.trust-store	信任库文件, 用于验证客户端证书	无
server.ssl.trust-store-password	信任库密码	无
server.ssl.trust-store-type	信任库类型	无
server.compression.enabled	是否启用压缩	false
server.compression.excluded-user-agents	设置不压缩的客户端	无
server.compression.mime-types	压缩的文件类型	text/html,text/xml,text/plain,text/css, text/javascript,application/javascript, application/json,application/xml
server.compression.min-response-size	超过该大小的文件才启用压缩	2048
server.servlet.context-path	应用上下文	无
server.servlet.session.persistent	是否报错 Session 在文件中	false
server.servlet.session.store-dir	Session 存储路径	无

server.servlet.session.timeout	Session 失效时间	30 分钟
server.servlet.session.cookie.domain	设置该 Cookie 的域名	无
server.servlet.session.cookie.http-only	设置为 true 表示防止客户端脚本读取 JSESSIONID	true
server.servlet.session.cookie.max-age	Cookie 失效的时间	无
server.servlet.session.cookie.name	设置 Cookie 的名称	JSESSIONID
server.servlet.session.cookie.path	Cookie 的使用路径	无
server.aas.basedir	AAMS 基路径，若不设置则使用临时文件夹	无
server.aas.max-threads	最大线程数	200
server.aas.min-spare-threads	最小空闲线程数	10
server.aas.max-http-post-size	最大 Post 数据大小	2097152
server.aas.max-http-header-size	请求或响应的最大 http header 大小	8192
server.aas.max-swallow-size	请求体大小限制	20971520
server.aas.uri-encoding	URI 的编码方式	utf-8
server.aas.max-connections	最大并发个数	10000
server.aas.accept-count	已连接但应用服务器未处理数	100
server.aas.max-keep-alive-timeout	连接保活超时时间	30000 (毫秒)
server.aas.max-keep-alive-requests	最多处理多少个请求后关闭连接，-1 表示不限制	100
server.aas.disable-upload-timeout	是否禁用上传超时	false
server.aas.relaxed-path-chars	允许请求行中的特殊字符	无
server.aas.relaxed-query-params	允许请求行中参数的特殊字符	无
server.aas.use-send-file	是否使用 sendfile 功能。	true
server.aas.license-path	License 路径	无
server.aas.resource.allow-caching	是否缓存静态文件	true
server.aas.resource.cache-ttl	缓存的 TTL	无

6.2 附录B 日志参数

参数	说明	默认值
server.aas.accesslog.enabled	是否打印 accesslog 日志，为 true 时为打印	false
server.aas.accesslog.pattern	日志格式	无
server.aas.accesslog.directory	日志文件存放的路径	无
server.aas.accesslog.prefix	日志名称前缀	access_log
server.aas.accesslog.suffix	日志名称后缀	.log

server.aas.accesslog.rotate	是否启用日志轮转	true
server.aas.accesslog.file-date-format	日志文件名中的日期格式	.yyyy-MM-dd
server.aas.accesslog.buffered	是否缓冲日志记录。设置 true 时，将异步记录日志	true

6.3 附录C actuator配置参数

参数	说明	默认值
management.port	actuator开放的端口,用于SpringBoot 1.X	无
management.context-path	actuator服务访问上下文，用于SpringBoot 1.X<	无
management.endpoints.web.base-path	actuator服务访问上下文，用于SpringBoot 2.X	无
management.security.enabled	actuator服务是否允许安全验证	无
management.server.port	actuator开放的端口,用于SpringBoot 2.X	无
management.endpoints.web.exposure.include	actuator开放endpoints 列表	health/info

金蝶天燕云计算股份有限公司（简称“金蝶天燕云”）成立于2000年，前身为“金蝶中间件公司”，是金蝶集团旗下新一代软件基础云平台服务商，云计算国家标准制定企业，国家信创产业核心软件企业。金蝶天燕是国家863重点研发计划与核高基重大专项承接企业，也是“两网一站四库十二金”国家重点工程的基础平台提供商，产品广泛应用于政府、军工、金融、能源等关键行业，累计服务客户总数超过10万家。



金蝶天燕云公众号

Apusic 金蝶天燕

信息技术应用创新核心企业
云计算国家标准制定企业
金蝶旗下软件基础云平台服务商
中电科集团太极股份成员企业

4008-555-800

www.apusic.com