

Jarboot使用手册

一、什么是Jarboot?

Jarboot是一个功能强大易用的 Java进程 启动器，使用jarboot可以方便的启动一批的Java进程，支持按照依赖的优先级顺序启动、进程守护、文件监控重启、进程在线调试。

二、安装和启动

1、下载安装包（支持docker）

下载地址：

1. **Gitee**: <https://gitee.com/majz0908/jarboot/releases>
2.  **Docker Hub**: <https://registry.hub.docker.com/r/mazheng0908/jarboot>

使用Docker，镜像名为**mazheng0908/jarboot**，执行如下docker命令：

```
sudo docker run -itd --name jarboot -p 9899:9899 mazheng0908/jarboot
```

Docker支持挂载的目录：/jarboot/conf, /jarboot/services, /jarboot/logs, /jarboot/data

2、解压zip安装包

3、启动服务

在Windows系统上使用 `startup.cmd` 启动服务，在Linux或MacOS上使用 `startup.sh` 启动。

#启动前确保已经安装了jdk1.8或以上版本

#Linux or MacOS

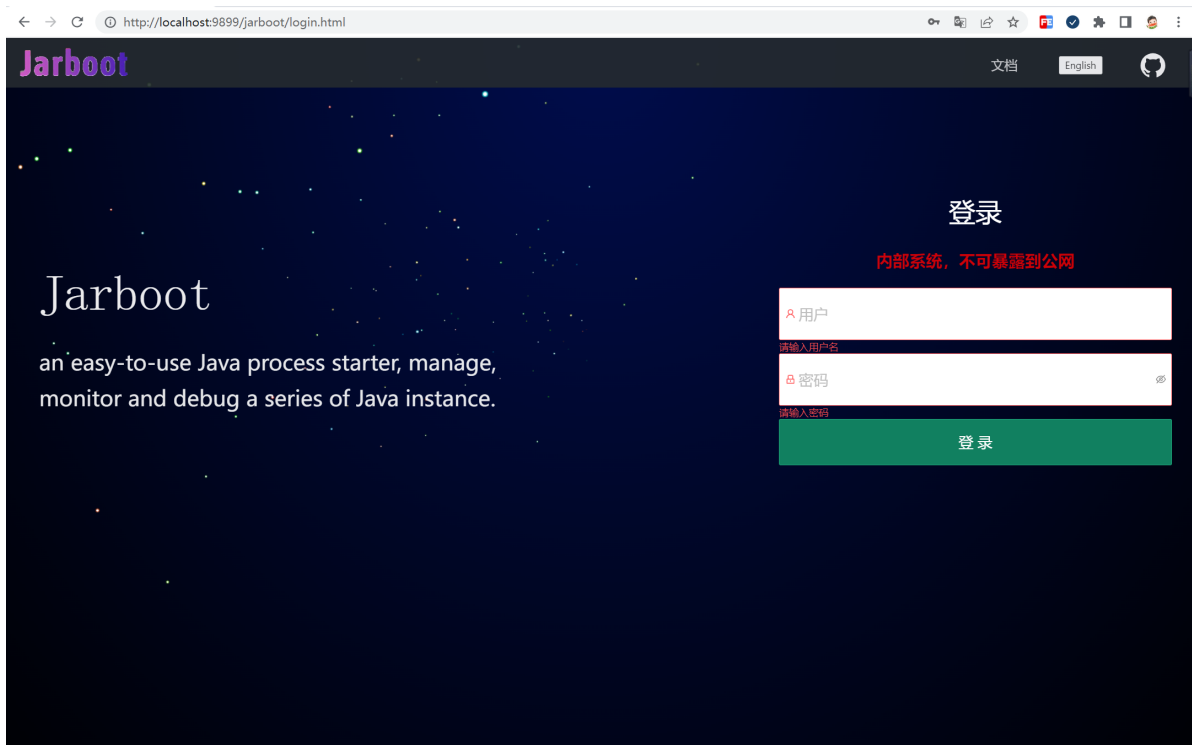
`sh startup.sh`

#Windows 执行startup.cmd或者双击startup.cmd

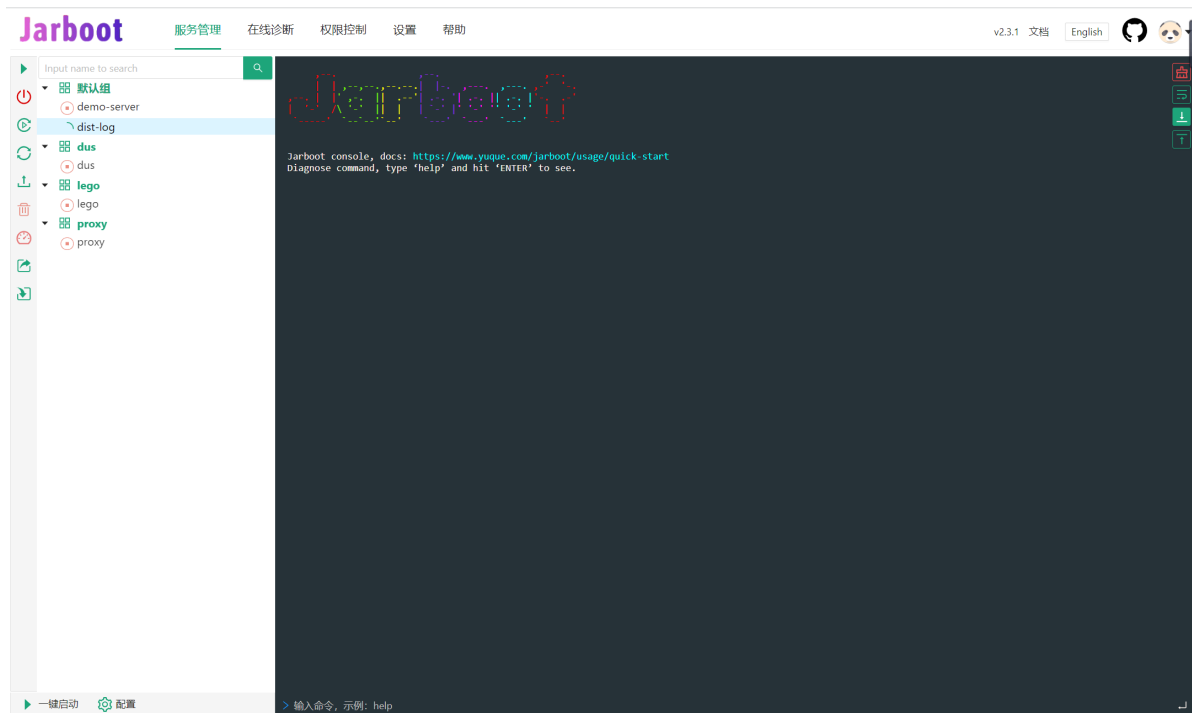
startup.cmd

启动成功后试用浏览器访问：<http://127.0.0.1:9899> 即可进入系统主界面。

默认的用户名：**jarboot** 密码：**jarboot**



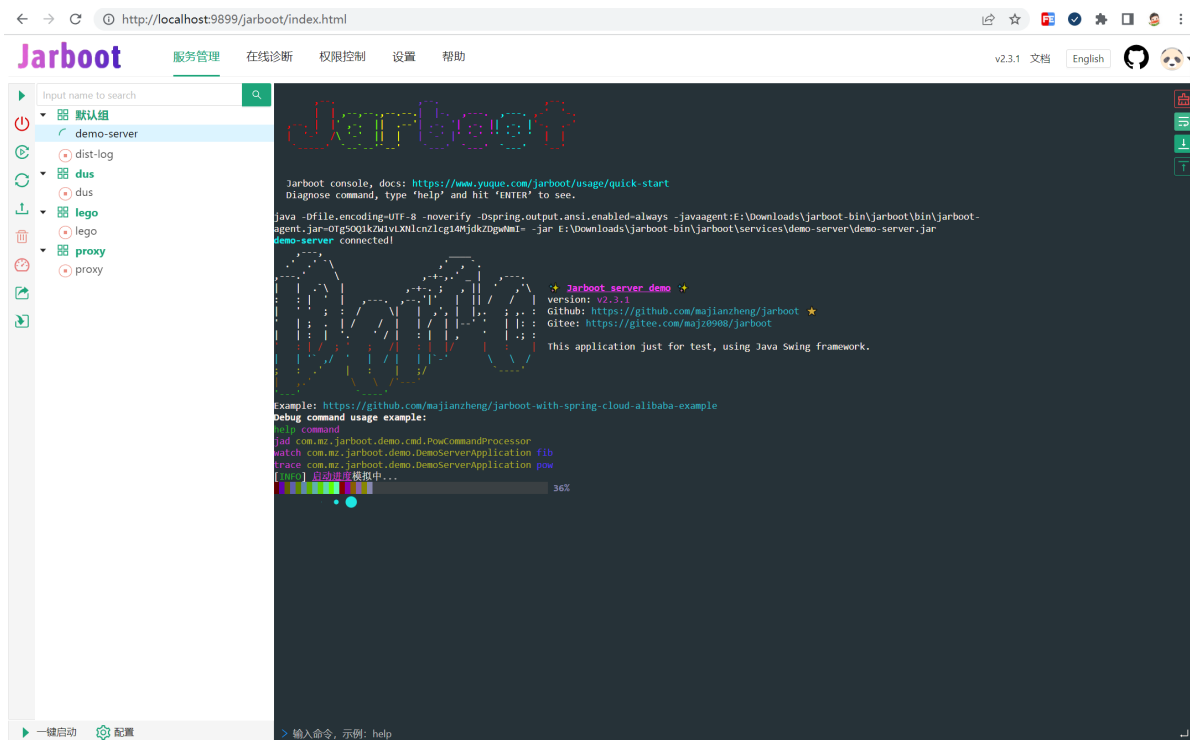
4、进入主界面



三、快速上手

服务启动

点击  按钮启动选择的服务。如下**demo-server**是安装包中自带的一个示例服务，点击启动后便会在界面上显示进程的控制台输出，启动完成后提示启动的耗时，示例程序启动如下图：



如何增加自己的Java服务

方式1：直接在 `services` 目录下创建自己服务的目录，目录的名字即为服务的名字，然后将自己的 jar 文件放入其中。

方式2：通过 Jarboot 提供的上传界面，点击工具栏的上传图标 ，一步步开始上传。

1. 点击上传的按钮，输入要上传的服务名，**如果不存在则创建新的服务，若存在则更新**；
2. 逐个上传文件，期间上传的所有文件会在临时目录，点击确定会将文件统一提交到服务的目录；
3. 所有文件上传完成后，点击 **确定**，将最终更新文件到服务目录，点击 **取消** 则清除临时目录；

输入要更新或新增的服务的名称

名字：

test

取消

下一步

注意：如果输入的名字是服务列表中不存在的，则会显示配置【启动命令】和【程序传入参数】的配置项。

上传test文件

名字：

test


启动命令：

示例： 1) -jar xx.jar 2) MainClassName 3) -cp xx.jar **/*.MainClass mainMethod 4) -classpath **/jar **/*.ClassName

程序传入参数：

Main arguments

💡 更多的配置信息，请到服务配置页面。



点击或拖拽文件到此区域上传
支持单个或批量上传。

取消

提交

配置启动的jar文件

当文件夹中存在多个jar文件时，需要配置启动时所使用的jar文件是哪个

服务配置

点击底部工具栏的配置按钮，即可切换到服务的配置界面，如下图：

Jarboot

服务管理 在线诊断 权限控制 设置 帮助

v2.3.1 文档 English

Input name to search

🔍

默认组

demo-server

dist-log

dus

dus

lego

lego

proxy

proxy

dus - 配置

名字：

dus

 组：

dus

启动命令：

-jar E:\Downloads\jarboot-bin\jarboot\services\dus\dus.jar --spring.cloud.nacos.config.server-addr=52.82.79.186:37002 --spring.cloud.nacos.config.namespace=dgmanager-dev --spring.cloud.nacos.config.file-extension=yaml

VM参数：

bootvmoptions

程序传入参数：

Main arguments

JDK：

JDK home path

工作目录：

环境变量：

eg: ENV1=val1,ENV2=val2

启动优先级：

1

进程守护：

☒

文件路径监控：

☒

提交

重置

关闭

配置界面对应的配置文件为 `boot.properties`，该文件在服务的目录下，界面中每个配置选项对应了配置文件中的一个key。配置文件内容如下：

```
#Properties file Jarboot created.
#Fri Jul 07 16:02:08 CST 2023
daemon=true
vm=boot.vmoptions
jdkPath=
jarUpdateWatch=true
command=\ -jar E:\Downloads\jarboot-bin\jarboot\services\lego\lego.jar --
spring.cloud.nacos.config.server-addr=52.82.79.186:37002 --
spring.cloud.nacos.config.namespace=dgmanager-dev --
spring.cloud.nacos.config.file-extension=yaml
env=NACOS_PORT=37002,NACOS_NAMESPACE=dgmanager-
dev,NACOS_HOST=52.82.79.186,NACOS_FILE_EXTENSION=yaml
args=
workDirectory=
priority=1
group=lego
```

如何启动？使用哪个jar文件启动？

需要配置command字段，即上图中的第一条【启动命令】。示例： 1) -jar xx.jar 2) MainClassName 3) -cp xx.jar ..MainClass mainMethod 4) -classpath **

1、默认：服务目录下只存在一个jar文件时不用配置

在对应的服务目录下如果只有一个jar文件，默认会使用这个文件启动，可以不配置启动所使用的jar文件。比如，我们通常使用的SpringBoot，如果编译成jar文件的话默认会只生成一个可运行的jar文件，内部包含了运行时所需的所有文件。

2、相对路径：使用相对于服务目录的路径指定启动的jar文件

首先，如果要启动的jar文件就在服务的目录下，那么直接输入文件的名字，不需要加路径；如果在其他的目录，则可以使用相对路径配置，如上图中的 `-jar ../bin/nacos-server.jar` 配置就是使用了相对路径，相对于当前的服务目录。

3、绝对路径：使用绝对路径指定要启动的jar文件

使用文件系统中的绝对路径确定jar文件的位置。

对应配置文件的key为： `command`，可以到 `boot.properties` 配置文件中直接配置

如何配置工作目录？

默认为服务的目录，同样支持**相对路径**和**绝对路径**。

在界面中的【**Working Directory**】（【**工作目录**】）处配置。

对应配置文件的key为：**workDirectory**，可以到**boot.properties**配置文件中直接配置。

工作目录可以各个服务单独配置，也可以全局配置一个目录，但是最终都可以使用绝对路径配置

* 名字: 组:

启动命令:

VM参数:

程序传入参数:

JDK:

工作目录:

环境变量:

启动优先级:

进程守护: ☒

文件路径监控: ☒

概览 权限控制 设置 帮助

工作空间:

默认的VM参数:

jarboot启动后自动启动服务: ☐

进程崩掉了，如何自动启动？进程守护开启

默认开启，开启后如果Java进程突然挂掉，则会自动将其启动。

对应配置界面选项为：【Daemon】

对应配置文件的key为：daemon，可以到boot.properties配置文件中直接配置

有依赖关系？启动/停止顺序如何配置？

如果多个进程间存在依赖关系，一个服务启动必须依赖另一个服务，此时如何确保依赖的服务先行启动呢？

启动时会按照优先级**从大往小的顺序启动**。停止时顺序是相反的，优先级相同的一组会并行启动。

对应配置界面选项为：【Priority】（【启动优先级】）

对应配置文件的key为：priority，可以到**boot.properties** 配置文件中直接配置

如何监控文件的更新，并自动重启服务？

一般用于持续集成部署，开启了该选项，则监控到服务文件夹下的jar文件更新时，会自动重启服务。

目录监控实现了防抖设计（在一定时间内的多次更新只会触发一次重启。

注意：目前该功能只会监控**服务目录下的.jar扩展名的文件**，如果使用了其他目录下的jar文件启动，启动的jar文件更新时，则不会生效。

对应配置界面选项为：【File path Watch】（【文件路径监控】）

对应配置文件的key为：jarupdatewatch，可以到**boot.properties** 配置文件中直接配置

