東方通 TongTech

400-650-7088

TongWeb V7.0

嵌入式版

响应式编程容器用户指南

版本：70E6P7A01

北京东方通科技股份有限公司

2024年01月

声明

版权

版权声明

- 本文档的所有权和知识产权归属于北京东方通科技股份有限公司（以下简称“东方通”）或其关联公司所有。
- 除非另有明确约定，东方通不授予任何明示或暗示的许可或权利，使用者不得以任何方式将本文档中的任何内容用于商业目的。
- 东方通对本文档享有版权。本文档的所有内容，包括但不限于文字、图像、图表、图标、示意图、屏幕截图等，均受版权法和国际版权条约的保护。
- 未经东方通明确授权，任何人不得对本文档以任何形式复制、修改、传播、分发、展示或进行衍生创作。

商标声明

- 東方通、**TongTech** 标识以及其他相关东方通图形、徽标、服务名称和商标（以下统称为“东方通商标”）是东方通或其关联公司的注册商标或商标。
- 未经东方通明确授权，任何人不得以任何方式使用、复制或展示东方通商标。此外，未经东方通事先书面同意，不得将东方通商标与其他商标、标识或徽标进行混淆、链接或结合使用。
- 除非另有明确约定，本商标声明不授予任何明示或暗示的许可或权利，对东方通商标的使用须获得东方通的明确授权。
- 本文档提及的所有其他商标、标识、徽标或产品名称均为其各自所有者的财产，并可能受其各自的商标法保护。

免责声明

请在使用东方通产品之前仔细阅读本免责声明，并根据自身情况判断是否继续使用。如有任何问题，请联系我们的客户支持团队。

- 本产品的使用是基于用户自己的判断和风险评估。本文档仅作为使用指导，不对使用本产品所产生的结果做任何明示或暗示的担保。
- 东方通不对用户未按照本文档中的指导正确使用东方通产品而导致的任何损失或损害承担责任；
- 本文档可能会包含第三方提供的内容、链接或资源，这些内容由第三方自行负责。东方通对于这些内容的准确性、完整性、合法性或可靠性不承担责任。用户在使用这些内容时应自行判断和承担风险。
- 由于产品版本升级或其他原因，本文档内容会不定期更新。东方通保留在不事先通知用户的情况下随时对文档进行修改、更新或中止的权利。

如需获取有关东方通产品的许可或使用权，请联系东北京东方通科技股份有限公司或授权代理商。任何违反本声明的行为将受到适用法律的追究。

版本变更说明

手册版本	适用产品	更新内容
70E6P7A01	7.0.E.6_P7	无
70E6P6A01	7.0.E.6_P6	无
70E6P5A01	7.0.E.6_P5	**新增如下章节：** • 一键解压安装，新增支持一键解压并安装资源 jar 包。 **修改如下章节：** • 产品包，新增 “installAll” 和 “deployAll” 脚本，一键解压安装资源 jar 包。
70E6P4A01	7.0.E.6_P4	**新增如下章节：** 响应式 Web 框架 WebFlux，WebFlux 适配支持。
70E6P3A01	7.0.E.6_P3	无
70E6P2A01	7.0.E.6_P2	无
70E6P1A01	7.0.E.6_P1	**修改如下章节**: • 引入资源依赖 章节，修改引入资源依赖的版本号，从“4.1.81.Final”修改为“4.1.85.Final”。 • lib目录说明 章节，新增“license-client-4.4.1.jar”包。 • License Server 远程认证 章节，新增“server.tongweb.license.license-public-key”配置参数说明，指定 license-sdk 请求加密公钥。

前言

本文档是TongWeb嵌入式版产品用户手册之一，详细介绍了TongWeb 嵌入式版响应式编程容器的安装及使用。

阅读前注意事项

通过阅读本文档，您确认并同意自行承担因未具备必要专业背景和知识而导致的任何风险或后果。在使用本文档中提供的信息和指南时，请始终谨慎，并在必要时寻求专业人士建议和指导。

- 适用对象

  本文档主要适用于使用本产品的系统管理员阅读，部分内容同样适用于基于本产品进行应用开发或应用部署的人员阅读。

- 专业背景

  本文内容可能涉及到操作系统、服务器硬件、网络等相关领域的知识。请确保您具备相关背景和知识，以便更好地理解和应用本手册的内容。

- 技能要求

  为了能够充分理解和应用本文档的内容，建议您具备如下技能:

  - 掌握 Linux 系统基本操作
  - 熟悉嵌入式相关知识

- 术语和概念

  本文档可能使用一些专业术语和概念。请确保您熟悉这些术语和概念，或者有能力查阅相关资料以便进一步理解。

- 实践经验

  为了最大程度地受益于本文档，建议您具备一定实践经验。这将帮助您更好地应用文档中的操作指南和建议。

请注意，本文档不适用于没有相关专业背景和知识的用户。如果您对本文档的内容或所需背景有任何疑问，请在使用之前咨询相关专业人士或寻求额外的支持。

用户手册集

TongWeb 嵌入式版为您提供如下用户使用手册。

编号	用户手册	说明
1	000_TongWeb_V7.0嵌入式版_产品介绍	详细介绍的TongWeb嵌入式版的概述、产品功能、产品特色等信息。
2	001_TongWeb_V7.0嵌入式版_JavaEE标准容器用户指南	详细介绍了 TongWeb 嵌入式版 JavaEE 标准容器的安装、使用及功能说明等信息。

编号	用户手册	说明
3	002_TongWeb_V7.0嵌入式版_JakartaEE标准容器用户指南	详细介绍了 TongWeb 嵌入式版 JakartaEE 标准容器的安装、使用及功能说明等信息。
4	003_TongWeb_V7.0嵌入式版_响应式编程容器用户指南	详细介绍了 TongWeb 嵌入式版响应式编程容器的安装、使用及适配网关等信息。
5	004_TongWeb_V7.0嵌入式版_转换工具指南	详细介绍了 TongWeb 嵌入版转换工具的使用说明。
6	005_TongWeb_V7.0嵌入式版_常见问题	详细介绍了使用 TongWeb 嵌入版产品所遇到的问题及解决方案等。

技术支持

东方通产品将为您提供全方位的技术支持，您可以通过以下方式获得技术支持：

- 网址：www.tongtech.com
- 电话：400-650-7088
- 邮箱：support@tongtech.com

您在取得技术支持时，请提供如下信息：

- 您的姓名
- 您的公司信息
- 您的联系方式
- 操作系统及其版本
- 产品版本号
- 日志等错误的详细信息

目录

1. 产品简介

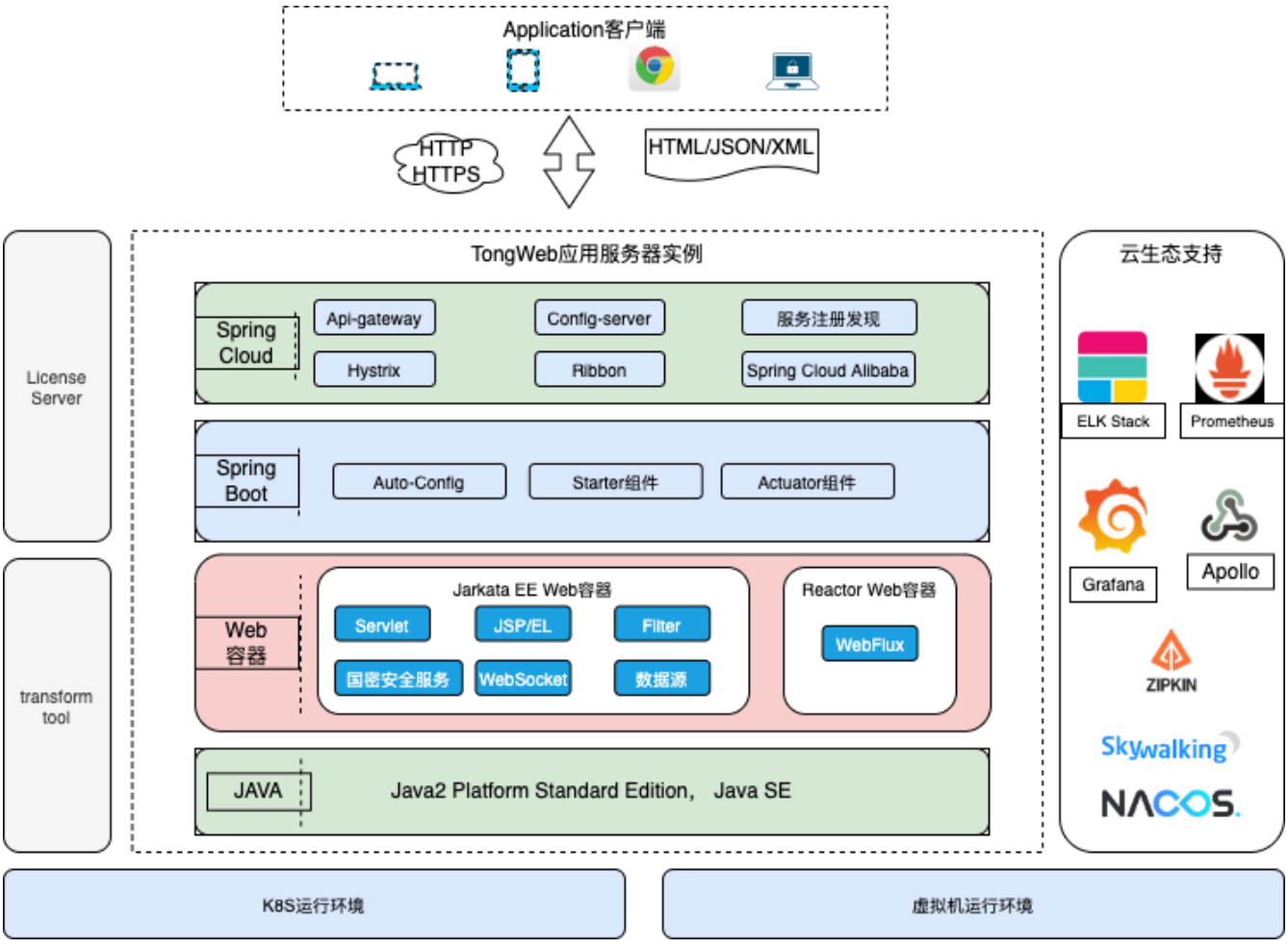
1.1. 响应式编程标准容器服务

TongWeb 嵌入式版响应式编程标准容器服务，支持响应式编程标准，可以兼容 spring webflux 体系。

其底层采用异步非阻塞式的通信模型，为用户提供一个在高并发环境中具有高性能、高可靠性的 web 容器。

同时，响应式编程标准容器服务“tongweb-spring-boot-reactor-starter”，兼容spring cloud gateway微服务网关组件。

spring cloud gateway 在微服务架构中一般用于作为系统的入口，为每个客户提供一套既可以基于编码也可以基于配置来使用的API， 提供了身份验证、监控、负载均衡、缓存、协议转换、限流熔断、静态响应处理等一系列能力。



1.2. 响应式 Web 框架 WebFlux

Spring WebFlux 是 Spring Framework 5.0 中引入的一种新的响应式 Web 框架。它基于 Reactor 框架实现，通过 Reactor 项目实现 Reactive Streams 规范，从而实现完全异步和非阻塞的框架。

Spring WebFlux 本身不会加快程序执行速度，但在高并发情况下，借助异步 IO 能够以少量而稳定的线程来

处理更高的吞吐量，从而规避文件 IO/网络IO 阻塞而导致的线程堆积。

WebFlux 具有以下特性：

- **异步非阻塞**

  可以举一个上传例子。相对于 Spring MVC 是同步阻塞IO模型，Spring WebFlux这样处理：线程发现文件数据没传输好，就先做其他事情，当文件准备好时通知线程来处理（这里就是输入非阻塞方式），当接收完并写入磁盘（该步骤也可以采用异步非阻塞方式）完毕后再通知线程来处理响应（这里就是输出非阻塞方式）。

- **响应式函数编程**

  相较于 Java8 Stream 同步、阻塞的 Pull 模式，Spring Flux 采用 Reactor Stream 异步、非阻塞 Push 模式。

  使用 Spring Flux 时，可使用 Java lambda 表达式书写代码，这种方式更接近自然语言的形式，易于理解。

1.3. 平台环境

操作系统	说明
Windows	Windows7、Windows10、Windows Server
Linux	CentOS7、Ubuntu、SUSE
Mac OS	Mac OS
国产平台	龙芯系列、飞腾系列、华为系列、申威系列、海光系列

1.4. 系统环境

系统组件	系统要求
Java环境	JDK8以上
内存	至少需要512MB的内存
硬盘空间	至少需要1024MB磁盘空间

1.5. 网关适配

分类	版本信息
Spring Boot	• Spring Boot 2.0.X • Spring Boot 2.1.X • Spring Boot 2.2.X • Spring Boot 2.3.X • Spring Boot 2.6.X • Spring Boot 2.7.X
Spring Cloud	• Spring Cloud 2021.0.x • Spring Cloud 2020.0.x • Spring Cloud Hoxton • Spring Cloud Greenwich • Spring Cloud Finchley

2. 产品清单

2.1. 产品包

注：{n} 表示具体版本号

产品包	适用标准
tongweb-spring-boot-reactor-7.0.E.{n}.zip	响应式编程产品安装包。
deployAll-{n}.bat	Windows环境下运行该命令，一键解压产品安装包，并将 lib 目录中的 jar 安装到企业私仓。
deployAll-{n}.sh	Linux环境下运行该命令，一键解压产品安装包，并将 lib 目录中的 jar 安装到企业私仓。
installAll-{n}.bat	Windows环境下运行该命令，一键解压产品安装包，并将 lib 目录中的软件包安装到本地 maven 仓库。
installAll-{n}.sh	Linux环境下运行该命令，一键解压产品安装包，并将 lib 目录中的软件包安装到本地 maven 仓库。
license.dat	使用响应式编程容器，需要依赖授权文件，申领授权文件（license.dat）可联系东方通销售人员。

2.2. 脚本文件说明

解压 TongWeb 嵌入式版 reactor 压缩包，文件说明如下所示。

文件	文件说明
installMavenJar.bat	Windows环境下运行该命令，可将lib目录中的软件包安装到本地maven仓库。
installMavenJar.sh	Linux环境下运行此命令，可将lib目录中的软件包安装到本地maven仓库。
deployMavenJar.bat	Windows环境下运行该命令，可将lib目录中的jar安装到企业私仓。
deployMavenJar.sh	Linux环境下运行该命令，可将lib目录中的jar安装到企业私仓。

2.3. lib目录说明

在lib目录中，TongWeb 嵌入式版 reactor 需要使用的软件包清单，{n} 表示具体版本号，如下表所示。

文件	文件说明
reactor-tongweb-core-7.0.E.{n}.jar	响应式编程容器核心，可结合tongweb-spring-boot-reactor-starter用于运行spring cloud gateway网关。
reactor-tongweb-http-7.0.E.{n}.jar	响应式编程容器支持http，可结合tongweb-spring-boot-reactor-starter用于运行spring cloud gateway网关。
tongweb-spring-boot-reactor-starter-7.0.E.{n}.jar	用于运行spring cloud gateway网关。
tongweb-reactor.pom	tongweb-spring-boot-reactor-starter-7.0.E.{n}.jar工程包依赖描述文件。
tongweb-lic-sdk-*.jar	license-server 的客户端。

3. 安装指引

3.1. 安装资源 jar 包

您可以通过安装脚本安装 TongWeb 嵌入式版工程包。

解压 tongweb-spring-boot-reactor-7.0.E.{n}.zip 资源包，进入解压包运行 `installMavenJar` 脚本或者 `deployMavenJar` 脚本即可将 TongWeb 嵌入版安装到仓库中。

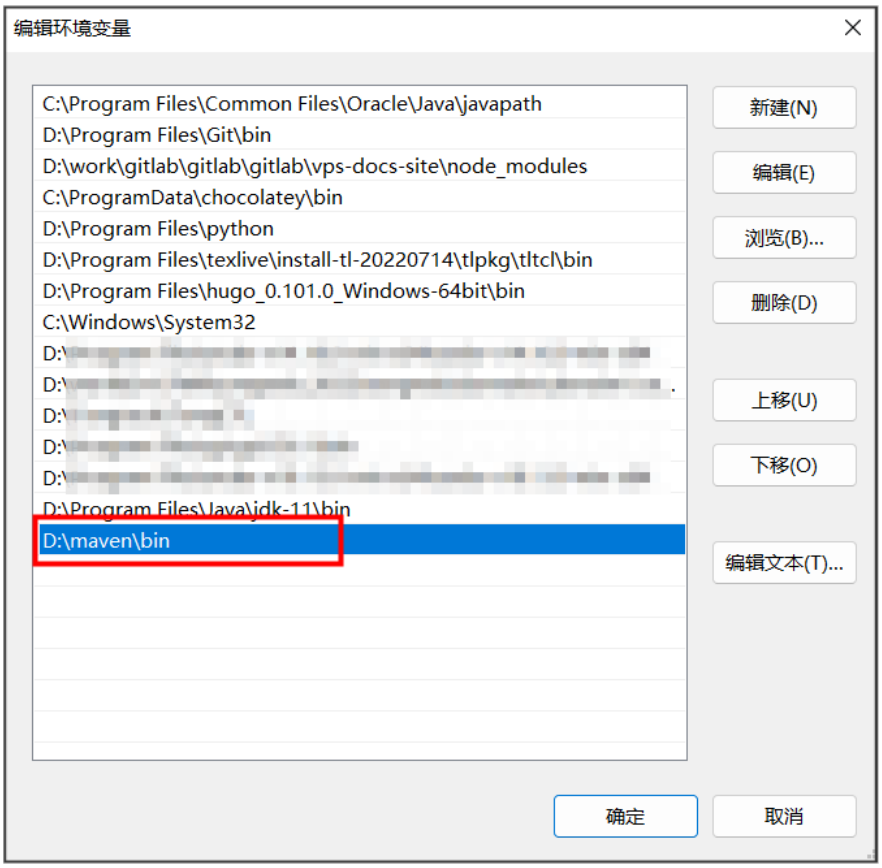
- installmavenJar 脚本：安装到本地 maven 仓库。
- deployMavenJar 脚本：安装到远端 nexus 或兼容 nexus 企业私有仓库。

3.1.1. 准备 maven 构建管理工具

将嵌入式版 TongWeb 产品资源 jar 包安装入本地 maven 仓库前，您需要提前准备 maven 构建管理工具。

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，例如：安装路径为 “D:\maven\bin”。



   - Linux环境

     a. 执行如下命令，打开“/etc/profile”文件。

```
sudo vi /etc/profile
```

b. 按“i”进入编辑模式，在文件中，添加如下环境变量信息。

请将 maven 安装路径修改为实际 maven 路径。

```
export MAVEN_HOME={maven安装路径}
export PATH=${PATH}:${MAVEN_HOME}
```

c. 配置完成后，按“Esc”退出编辑，输入“:wq!”，保存并退出。

3. 执行如下命令，查看maven是否安装成功。

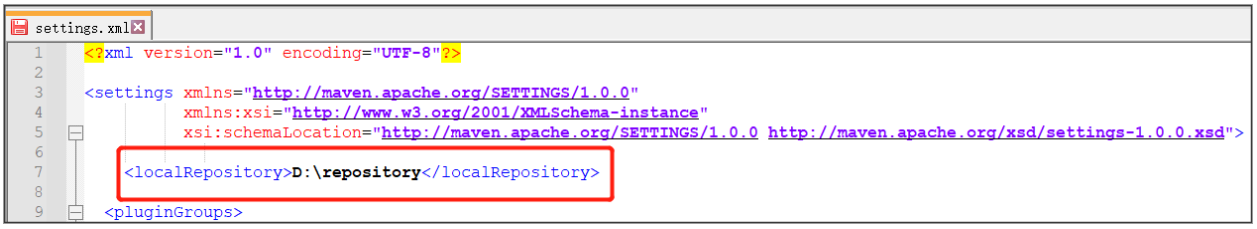
```
mvn -v
```

若回显信息如下所示，则说明maven管理工具安装成功。

```
Apache Maven 3.6.0 (97c98ec64a1fdfee7767ce5ffb20918da4f719f3; 2018-10-25T02:41:47+08:00)
Maven home: D:\software\apache-maven-3.6.0\bin\..
Java version: 1.8.0_191, vendor: Oracle Corporation, runtime: D:\Java\jdk1.8.0_191\jre
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

4. 指定 maven 的本地仓库路径，例如：Windows 环境下，将本地仓库指定为“D:\repository”。
   a. 进入 maven 安装目录，例如：D:/maven/conf。
   b. 打开 “settings.xml”文件。
   c. 将 “<localRepository>”配置项指定为 “D:\repository”，如下所示。

   说明： 请根据实际路径修改。



3.1.2. 一键解压安装

安装到本地仓库

使用 installAll-{n} 脚本，解压 TongWeb 嵌入式版资源包，将 lib 目录中的 jar 包安装到本地 maven 仓库。

前置条件

installAll-{n} 脚本与 TongWeb 嵌入式版资源包在同级目录。

操作步骤

1. 运行 installAll-{n} 安装脚本，将 lib 下的 jar 包安装到本地 maven 仓库。

- Windows环境

```
installAll-{n}.bat
```

- Linux环境

```
./installAll-{n}.sh
```

2. 验证安装。
   a. 进入 maven 的本地仓库目录，安装 maven 步骤4 中指定的本地仓库目录。例如:"D:/repository"。
   b. 进入 “…/repository/com/tongweb/” 可以找到安装的 lib 目录下的 jar 包，此时即表示 jar 包部署到本地仓库成功。

安装到企业私仓

使用 `deployAll-{n}` 脚本，解压 TongWeb 嵌入式版资源包，并将 lib 目录中的 jar 包安装到远端 nexus 或兼容 nexus 企业私有仓库。

语法说明

```
deployAll-{n}.sh [url] [sid]
```

- **url**：需要上传的企业私仓地址。

  注意：必须输入企业私仓完整路径信息。

- **sid**：maven 的 “setting.xml” 中配置的 <server> 标签下的 id 信息。

  注意：必须在 “setting.xml” 中配置企业私仓密钥信息。

操作步骤

1. 运行 `deployAll-{n}` 脚本，将 lib 下的 jar 包安装到企业私仓。

   例如：企业私仓的地址为“http://127.0.0.1:8081/nexus/content/repositories/central/”；sid 以“central”为例。

   请根据实际环境进行修改。

   - Windows环境

```
deployAll-{n}.bat http://127.0.0.1:8081/nexus/content/repositories/central/ central
```

   - Linux环境

```
./deployAll-{n}.sh http://127.0.0.1:8081/nexus/content/repositories/central/ central
```

2. 验证安装
   - 进入 maven 的企业私仓目录，安装 maven 步骤4 中指定的本地仓库目录。
   - 可以在对应仓库地址中找到安装的 lib 目录下的 jar 包，此时即表示jar包部署到企业私仓成功。

3.1.3. 手动解压安装

安装到本地仓库

使用 installMavenJar 脚本，将 TongWeb 嵌入式版所有资源 jar 包安装到本地 maven 仓库中。

操作步骤

1. 解压嵌入式版 TongWeb 产品资源包。
2. 进入解压后的根目录。

   为了方便后续描述，本章节后文中将以“${TongWeb_HOME}”指代嵌入式版 TongWeb 的解压目录。
3. 运行 `installMavenJar` 安装脚本，将 lib 下的 jar 包安装到本地 maven 仓库。
   - Windows环境

     ```
     installMavenJar.bat
     ```
   - Linux环境

     ```
     ./installMavenJar.sh
     ```
4. 验证安装。
   a. 进入 maven 的本地仓库目录，安装 maven 步骤4中指定的本地仓库目录。例如:"D:/repository"。
   b. 进入 “…/repository/com/tongweb/” 可以找到安装的 lib 目录下的 jar 包，此时即表示 jar 包部署到本地仓库成功。

安装到企业私仓

使用 `deployMavenJar` 脚本，将 TongWeb 嵌入式版本所有资源 jar 包安装至企业私仓。

语法说明

```
deployMavenJar.sh [url] [sid]
```

- **url**：需要上传的企业私仓地址。

  注意：必须输入企业私仓完整路径信息。
- **sid**：在 maven 的“setting.xml”中配置的 server 部分 id 信息。

  注意：必须在"setting.xml"中配置企业私仓密钥信息。

操作步骤

1. 解压嵌入式版 TongWeb 产品资源包。
2. 进入解压后的根目录。
3. 运行 `deployMavenJar` 脚本，将 lib 下的 jar 包安装到企业私仓。

   例如：企业私仓的地址为“http://127.0.0.1:8081/nexus/content/repositories/central/”；sid 以“central”

为例。

请根据实际环境进行修改。

- Windows环境

```
deployMavenJar.bat http://127.0.0.1:8081/nexus/content/repositories/central/ central
```

- Linux环境

```
./deployMavenJar.sh http://127.0.0.1:8081/nexus/content/repositories/central/ central
```

4. 验证安装
   - 进入 maven 的企业私仓目录，安装 maven 步骤4 中指定的本地仓库目录。
   - 可以在对应仓库地址中找到安装的 lib 目录下的 jar 包，此时即表示jar包部署到企业私仓成功。

3.2. License 授权认证

嵌入式版 tongweb-spring-boot-reactor-starter 启动需依赖授权文件校验，申领授权文件（license.dat）可联系东方通销售人员。

嵌入式版 tongweb-spring-boot-reactor-starter 支持两种授权认证方式，分别是“本地认证”和“远程认证”。

3.2.1. license.dat 本地认证

通过读取本地的 license.dat 文件进行授权认证。

当使用 tongweb-spring-boot-reactor-starter.jar 包时，通过 Spring Boot 配置加载对应 application.properties、application.yml 等文件加载配置项。

您可以通过在“application.properties”或者“application.yml”配置文件配置 license。

配置参数说明

- server.tongweb.license.type = file ，表示 license 认证方式为本地认证。
- server.tongweb.license.path = license 存放路径，表示 license.dat 文件的存放路径。

  License 存放路径支持“本地磁盘路径”和“classpath路径”，详细配置步骤请参见配置 license 存放路径。

配置 license 存放路径

“本地磁盘路径”和“classpath路径”配置方式，如下所示。

- 本地磁盘路径

```
server.tongweb.license.path=D:\\license.dat
```

- classpath路径
  - 若授权文件（license.dat）放入 “工程/resources” 目录

在“application.properties” 或者“application.yml” 配置文件中，增加如下配置。

```
server.tongweb.license.path=classpath:\\license.dat
```

或

```
server.tongweb.license.path=classpath:license.dat
```

- 若授权文件（license.dat）放入 “工程/src/main/java/”目录

  a. 首先在 “pom.xml” 文件中，新增如下配置。

```
...
<resources>
  <resource>
    <directory>src/main/java</directory>
    <includes>
      <include>**/*.properties</include>
      <include>**/*.dat</include>
    </includes>
    <filtering>false</filtering>
  </resource>
  <resource>
    <directory>src/main/resource</directory>
    <includes>
      <include>**/*.properties</include>
      <include>**/*.dat</include>
    </includes>
    <filtering>false</filtering>
  </resource>
</resources>
...
```

  b. 再在“application.properties” 或者“application.yml” 配置文件中，增加如下配置。

```
server.tongweb.license.path=classpath:\\license.dat
```

  或

```
server.tongweb.license.path=classpath:license.dat
```

配置示例说明

```
#配置认证方式
server.tongweb.license.type = file
#配置license.dat文件路径地址
server.tongweb.license.path=D:\\demo\\license.dat
```

3.2.2. License Server 远程认证

License Server 远程认证，是通过远程搭建的 License Server 服务进行远程授权认证。

当使用 tongweb-spring-boot-reactor-starter.jar 包时，通过 Spring Boot 配置加载对应 application.properties、application.yml 等文件加载配置项。

您可以通过在“application.properties”或者“application.yml”配置文件中，配置 License Server 信息。

前置条件

已搭建 License Server 服务。

配置参数说明

参数	参数说明
server.tongweb.license.type= remote	配置为 “remote”，表示远程校验，即使用 License Server 进行校验。
server.tongweb.license.license-ips	配置 License Server 的 IP 地址和端口，如 “127.0.0.1:8888”。当使用 https 时，请务必配置为 “https://[ip]:[port]”。
server.tongweb.license.license-public-key	指定 license-sdk 请求加密公钥。
server.tongweb.license.ssl-enabled	是否开启 SSL。
server.tongweb.license.ssl.key-store	Key store 路径。即加密密钥库文件。用于保护客户端和服务器之间的通信。 Keystore 中的私钥用于加密和解密客户端和服务器之间的数据，而证书用于验证服务器的身份。加密私钥和加密证书打包成加密密钥库文件。
server.tongweb.license.ssl.key-store-password	用于访问 Key store 的密码。
server.tongweb.license.ssl.key-store-type	Key store 类型。
server.tongweb.license.ssl.trust-store	Trust store 路径。即签名密钥库文件，但它存储的是客户端信任的证书。在客户端连接到服务器时，它会检查服务器提供的证书是否存在于 truststore 中。 若是，则客户端会信任该服务器，并允许建立安全的 SSL 连接。签名私钥和签名证书打包成密钥库文件。

参数	参数说明
server.tongweb.license.ssl.trust-store-password	用于访问 trust store 的密码。
server.tongweb.license.ssl.trust-store-type	Trust store 类型。
server.tongweb.license.sync	同步授权校验开关，默认为同步校验。当为空或者“true”时为同步校验，当为“false”时为异步校验。
server.tongweb.license.license-public-key	License-SDK 请求加密公钥，从 License Server 管理控制台获取，必须配置。 需要从 License Server 管理控制台获取。 1. 登录 License Server 管理控制台。 2. 在上传的 license.dat 证书所在行的操作列，单击“生成Key”，即可获得公钥信息。

配置示例说明

```
#配置认证方式
server.tongweb.license.type=remote
#配置为 License Server 的 IP 地址和端口，如“127.0.0.1:8888”。当使用 https 时，请务必配置为“https://[ip]:[port]”。
server.tongweb.license.license-ips=127.0.0.1:8888
#配置 license-sdk 请求加密公钥
server.tongweb.license.license-public-key=license-sdk 请求加密公钥
#配置是否启用 ssl
server.tongweb.license.ssl-enabled=true
#配置加密密钥库文件
server.tongweb.license.ssl.key-store=classpath:client.p12
#配置加密密钥库文件的密码
server.tongweb.license.ssl.key-store-password=123456
#配置加密密钥库文件的格式
server.tongweb.license.ssl.key-store-type=PKCS12
#配置签名密钥库文件，路径支持绝对路径和classpath
server.tongweb.license.ssl.trust-store=classpath:server.jks
#配置签名密钥库文件的密码
server.tongweb.license.ssl.trust-store-password=123456
#配置签名密钥库文件的格式
server.tongweb.license.ssl.trust-store-type=JKS
```

4. 引入资源依赖（网关）

TongWeb 嵌入式版 tongweb-spring-boot-reactor-starter 组件提供对 Spring Cloud Gateway 适配支持。

首先需要排除 gateway 中 reactor netty 的相关依赖，再引入 tongweb-spring-boot-reactor-starter，就可以正常使用 Spring Cloud Gateway。

4.1. 前置条件

已安装资源 jar 包。

4.2. 配置示例

配置示例，如下所示。

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>io.netty</groupId>
      <artifactId>netty-bom</artifactId>
      <version>4.1.85.Final</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
    <dependency>
      <groupId>io.projectreactor</groupId>
      <artifactId>reactor-core</artifactId>
      <version>3.4.23</version>
    </dependency>
  </dependencies>
</dependencyManagement>
<dependencys>
...
  <dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-gateway</artifactId>
    <exclusions>
      <exclusion>
        <groupId>io.projectreactor.ipc</groupId>
        <artifactId>reactor-netty</artifactId>
      </exclusion>
      <exclusion>
        <groupId>io.projectreactor.addons</groupId>
```

```
            <artifactId>reactor-extra</artifactId>
        </exclusion>
        <exclusion>
            <groupId>io.projectreactor.netty</groupId>
            <artifactId>reactor-netty-http</artifactId>
        </exclusion>
        <exclusion>
            <groupId>io.projectreactor.netty</groupId>
            <artifactId>reactor-netty</artifactId>
        </exclusion>
        <exclusion>
            <groupId>io.projectreactor</groupId>
            <artifactId>reactor-core</artifactId>
        </exclusion>
    </exclusions>
</dependency>
<dependency>
    <groupId>com.tongweb</groupId>
    <artifactId>tongweb-spring-boot-reactor-starter</artifactId>
    <version>7.0.E.{n}</version>
</dependency>
...
<dependencys>
```

5. 配置说明

TongWeb 支持在“application.properties”或者“application.yml”配置文件中，对连接相关参数进行配置。


该配置说明适用于 reactor 容器和网关配置。

配置示例，如下所示。

```
server:
  tongweb-reactor:
    initial-buffer-size: 1MB
```

5.1. 通用参数配置说明

参数说明，如下表所示。

配置参数	说明	默认值
server.tongweb-reactor.h2c-max-content-length	H2C 请求的最大内容长度（需在配置文件中关闭 SSL，并开启 HTTP2 才能生效）。	0B
server.tongweb-reactor.initial-buffer-size	解码 HTTP 请求的初始缓冲区大小。	128B
server.tongweb-reactor.max-chunk-size	解码 HTTP 请求块（chunk）的最大值。	8KB
server.tongweb-reactor.max-initial-line-length	解码 HTTP 请求初始行的最大值。	4KB
server.tongweb-reactor.validate-headers	请求解码时是否验证 header 里包含的 CRLF 字符。	true

5.2. TSL/SSL

TongWeb reactor 容器支持以 TSL/SSL 的方式保障 HTTP 通信的安全。

使用 SSL 的配置示例，如下所示。

```
server:
  ssl:
    enabled: true
    key-store: classpath:scg-keystore.p12
    key-alias: gateway
    key-store-type: PKCS12
```

```
  key-store-password: 123456
```

参数说明，如下表所示。

配置参数	说明
key-alias	证书别名。
key-store-password	密钥库密码。
key-store	证书位置。
key-store-type	密钥库类型。

若需要信任下游的所有 SSL 证书，可开启如下配置。

```
spring:
  cloud:
    gateway:
      httpclient:
        ssl:
          useInsecureTrustManager: true
```

TongWeb reactor 容器支持配置 SSL/TSL 连接建立阶段握手的超时时间。

配置示例，如下所示。

```
spring:
  cloud:
    gateway:
      httpclient:
        ssl:
          handshake-timeout: 10000
          close-notify-flush-timeout: 3000
          close-notify-read-timeout: 0
```

附录 A: 术语及缩略语

术语	说明
响应式编程标准容器服务	TongWeb 嵌入式版响应式编程标准容器服务，支持响应式编程标准，可以兼容 spring webflux 体系。 其底层采用异步非阻塞式的通信模型，为用户提供一个在高并发环境中具有高性能、高可靠性的 web 容器。
License Server	License Server 是为 TongWeb 实例量身定制的授权监控中心，对于多版本、多实例的 TongWeb 进行统一鉴权校验、调度管理、监控。 方便用户在大规模的业务系统中使用 TongWeb。
H2C	HTTP/2协议定义了两个版本：HC 和 H2C。 H2C 版本是建立在明文的 TCP 之上的 HTTP/2 协议，这个标志被用在 HTTP/1.1 的升级协议头域和其它任何直接在 TCP 层之上的 HTTP/2 协议。