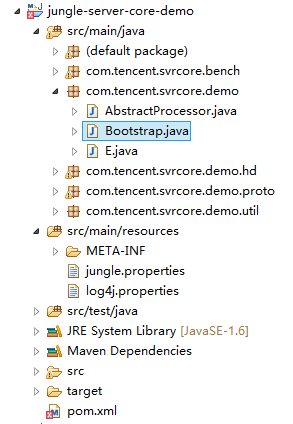
**Jungle Server Core使用与扩展**

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **时间** | **修改人** | **备注** |
| 1.0 | 2016/7/6 | mapleliang | - |
|  |  |  |  |

# 使用篇

Jungle使用maven构建，一般工程源码结构：



1. svr启动类com.tencent.svrcore.demo.Bootstrap：

创建依赖管理容器guice、加载配置文件（jungle.properties）、定义传输层协议tcp/udp、定义svr协议类型（CodecService）、定义线程模型（WorkerService）、定义业务命令号映射服务（ProcessorService）。

抽象基类AbstractProcessor：

派生出各各具体命令号处理接口，为其提供统一的日志、监控、异常处理等逻辑。一般还提供统一、单例的后端访问服务（ClientIoService），以便统一管理对后端的连接。

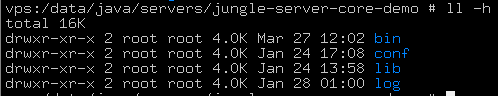
1. svr配置文件jungle.properties：

在guice中由JungleCoreModule负责加载。// TODO 待续……

1. 项目依赖文件pom.xml：

svrcore当前支持协程（基于kilim）与线程池模式，由<kilim.weave.required>false</kilim.weave.required>控制。为true时表示使用协程模式，javac编译得到class文件后kilim weave tool将改写字节码，此时要求2.2版本的rar-plugin以支持正确打包<plugin><groupId>org.apache.maven.plugins</groupId><artifactId>maven-rar-plugin</artifactId><version>2.2</version></plugin>。

1. 运行时目录：



bin：

存放服务起停脚本，如restart.sh。运行命令如：

cd bin

nohup java -Xmx2024m -DJUNGLE\_APP\_NAME=jungle-server-core-demo -DJUNGLE\_SPEC\_CONF=../conf/ -DLOG4J\_CONFIG\_DIR=../conf/ -cp .:../lib/\* com.tencent.svrcore.demo.Bootstrap >> ../log/nohup.out 2>&1 &

conf：

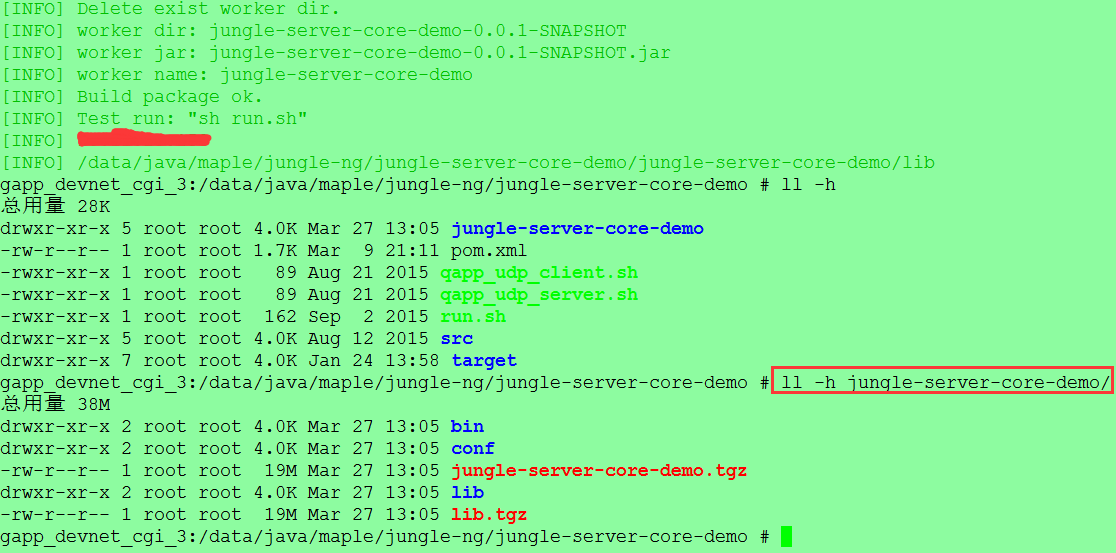
存放配置文件。jungle svr配置文件读取逻辑，先读conf下配置文件，再读类路径下配置文件，且后读取的项不覆盖先读取的项。因此一般在svn中把配置项都设为生产环境配置，部署在测试环境时，另配一份放测试机器用于测试环境；生产环境中conf下配置只放关键性配置如开关等。

lib：

存放依赖jar包。

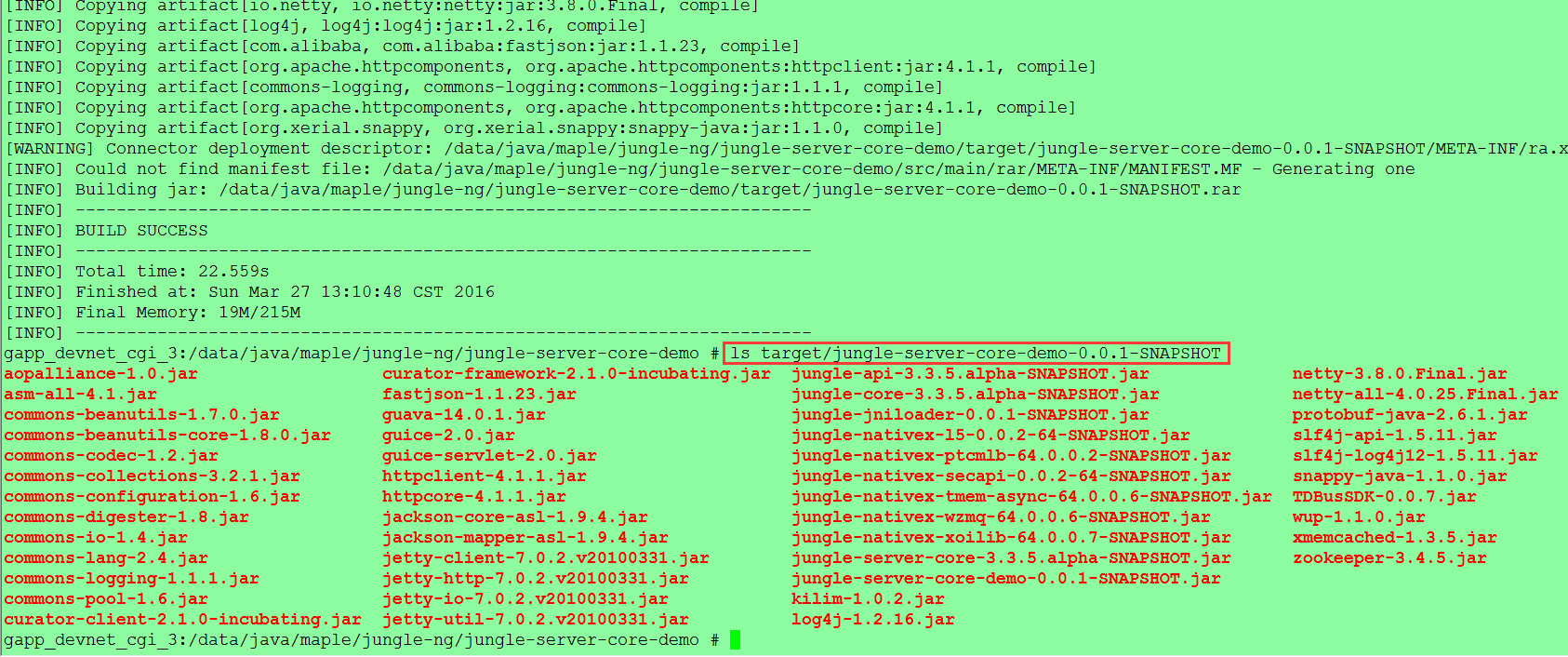
1. 打包：

jmake是jungle团队构建的一个简易打包脚本，在pom.xml目录下执行后得到



其中jungle-server-core-demo.tgz是bin conf lib三个目录的打包，lib.tgz是lib目录的打包

如果没有jmake，也可以在pom.xml目录执行如下命令mvn clean package rar:rar，得到



目录target/jungle-server-core-demo-0.0.1-SNAPSHOT即为jmake得到的lib目录，剩余的bin conf目录需要开发者自行创建。

# 扩展篇

1. server core支持哪些协议？如何新增协议支持？

框架对协议的编解码都封装在CodecService中。当前支持的协议

应用部纯protobuf协议qapp

应用部QunPacket包头+（protobuf包体/binary json包体）

平台部手Q业务svr协议[SSO（MSF）](http://km.oa.com/group/15585/articles/show/176963)

社平部独立app业务svr协议[WNS](http://km.oa.com/group/wns)

平台部OIDB

互动视频通信协议qtalk

框架对网络包抽象出一些共性属性，如：

sequence，客户端维护的请求包ID。svr原样返回，用于在同一个socket连接中同时发送不同的请求包

cmd，命令号。ProcessorService使用该属性映射请求包到Processor

routerId，路由标识。RouterService使用该属性映射请求包到IP端口，其值可填一个L5 id、CMLB appid、统一服务名，甚至简单实现可填IP地址

1. ProcessorService



ProcessorService接口只有一个方法Processor<IoPacket, IoPacket> map(Channel ch, IoPacket req);，作用是收到客户端请求包后，把请求包交到对应的svr类处理。

内层的PropertiesProcessorService表示从一个.properties文件中读取命令号与处理接口，格式类似

server.appname=java\_svrcore\_demo

java\_svrcore\_demo.p.HelloWorld=com.tencent.svrcore.demo.hd.HelloWorld

java\_svrcore\_demo.p.Test=com.tencent.svrcore.demo.hd.Test

其中server.name表示该svr的名称，绿色部分表示命令号。

目前server core本身只支持了从properties文件中读取映射关系。开发者也可以根据需要自行定制ProcessorService，如从xml中读取，从网络中读取，或者重写map方法以支持按正则的方式匹配命令号等。

NotFoundProcessorService：如果客户端请求过来的是一个未定义的命令号，框架会直接忽略未识别的命令号请求包。此时可以由此类给一个默认的服务器动作，定制响应错误报文到客户端。

1. WorkerService

WorkerService抽象了线程模型。当前有三种实现ThreadPoolWorkerService、KilimBasedWorkerService、ExtremeWorkerService。如果由Guice来管理依赖，框架将自动根据是否执行了kilim weave来绑定实现；ExtremeWorkerService使用时需指定配置server.worker\_service.extreme=true。

目前框架使用kilim改写javac得到的字节码来实现协程，如果使用者想引入更优的协程实现方式，可能需要自行定义WorkerService。

1. ServerIoService

ServerIoService抽象了一个svr的几个共性：传输层、编解码器、线程模型、处理器等。当前有三种实现TcpServerIoService、UdpServerIoService、ZmqServerIoService。其中ZmqServerIoService是对ZMQ的简单封装（ZMQ建立连接需要握手），仅支持tcp。ZmqServerIoService+QunCodecService即jungle-web-core中的<L5Worker />。