新巴巴项目SSM

2018年7月26日

22:43

计算机生成了可选文字:
vsrc/main/resources
vconfig
冈activeMQ.xml
冈annotation.xml
冈dataSource.xml
冈dubbo-provider.xml
冈mybatis.xml
冈property.xml
冈redis.xml
冈solr.xml
冈transaction.xml
区application-context.xml
0jdbc.properties
囟sqlMapConfig.xml

annotation.xml内容如下

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:mvc="http://www.springframework.org/schema/mvc"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:aop="http://www.springframework.org/schema/aop"

xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:task="http://www.springframework.org/schema/task"

xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"

xsi:schemaLocation="http://www.springframework.org/schema/beans

<http://www.springframework.org/schema/beans/spring-beans-4.0.xsd>

<http://www.springframework.org/schema/mvc>

<http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd>

<http://www.springframework.org/schema/context>

<http://www.springframework.org/schema/context/spring-context-4.0.xsd>

<http://www.springframework.org/schema/aop>

<http://www.springframework.org/schema/aop/spring-aop-4.0.xsd>

<http://www.springframework.org/schema/tx>

<http://www.springframework.org/schema/tx/spring-tx-4.0.xsd>

<http://www.springframework.org/schema/task>

<http://www.springframework.org/schema/task/spring-task-4.0.xsd>

<http://code.alibabatech.com/schema/dubbo>

<http://code.alibabatech.com/schema/dubbo/dubbo.xsd>">

<context:component-scan base-package="cn.itcast"/>

</beans>

计算机生成了可选文字:
<beanid="activeMQConnectionFactory
<propertyname="brokerURL“value
<propertyname=“value
<propertyname=password"value
</bean>
cl彐55=org.apaChe.activemq.ActiveMQConnectionFactory>
<beanid=pooledConnectionFactoryBean"cl彐55=org.apache.activemq.pool.PooledConnectionFactoryBean>
<propertyname=connectionFactory"e="activeMQConnectionFactory/>
<propertyname="maxConnections"value=“2/>
</bean>
<beanid="Sing1eConnectionFactory"CI彐55=0丆．sprlngframework.jms.connection.SingleConnectionFactory>
<propertyname="targetConnectionFactory"汇e《=pooledConnectionFactoryBean“/>
</bean>
<beanid="jmsTemp1ate"cl彐55=0丆g．springframework.jms.core·JmsTemp1ate">
<propertyname=connectionFactory"e="SingleConnectionFactory/>
<propertyname="defaultDestinationName"value=productld",/>
</bean>
<beanid="cust01n-tvIessageLIstener"Cl彐55=cn．亡caS亡．COre.meSSage.CustomMessageLIStener>
</bean>
监听彐ct主veMQ中罡否有氵肖息
<beancl彐55=0丆．listener.DefaultMessageListenerContainer>
<propertyname=connectionFactory"e="SingleConnectionFactory"/>
<propertyname="destinationName"value=productld/>
<propertyname=messageLIStener"e《="customMessageLIStener"/>
</bean>

计算机生成了可选文字:
<beanid="dataSource“cl彐55=com.alibaba.druid.pool.DruidDataSource>
<propertyname="driverC1assName"value="${jdbc.driver}“/>
<propertyname=“u“value="${jdbc.u丆丿“/>
username"“/>
<propertyname=
password"value="${jdbc.password}“/>
<propertyname=
</bean>

计算机生成了可选文字:
<dubbo：彐pp1主C彐t主0nname="babasport—servlce—product"/>
<dubbo：registry彐dds55="192．168．200．128：2181"
p汇0t0C01：'zookee/>
<dubbo
<dubbo
<dubbo
<dubbo
<dubbo
<dubbo
<dubbo
：汇eg主st汇y彐dd汇s55："N./“
<dubbo：p汇0t0C01PO汇t：“20880“
name="dubbo“/>
：5e汇V上Ce
：5e汇V上Ce
：5e汇V上Ce
：5e汇V上Ce
：5e汇V上Ce
：5e汇V上Ce
cn．亡cas亡
cn．亡cas亡
cn．亡cas亡
cn．亡cas亡
cn．亡cas亡
cn．亡cas亡
·TestTbSerV1ce"e="testTbSerV1ce/>
·COEe·SCe
.BrandSerV1ce"e="brandSerV1ce/>
·COEe·SCe
UploadServ1ce"e《=“oadServ1ce/>
·COEe·SCe·
·ProductSerV1ce"汇e《=productSerV1ce/>
·COEe·SCe
·CO0丆Se丆V“e《=“CO0丆Se丆V/>
·COEe·SCe
·SkuSerV1ce"e="skuSerV1ce/>
·COEe·SCe

计算机生成了可选文字:
<beancl彐55=Sq1SessionFactoryBean>
<propertyname="dataSource“e《="dataSource/>
<propertyname="configLocation"“/>
</bean>
<beancl彐55=spr1ng.mapper.MapperScannerConfigurer"
<propertyname="basePackage"value=cn．亡caS亡．core.dao/>
</bean>

计算机生成了可选文字:
<beancl彐55=0丆g．factory.config.PropertyPlaceholderconfigurer"
<propertyname='oca亡ons>
<1主st>
<value>classpath：jdbc.properties</value>
</1主st>
</property>
</bean>

计算机生成了可选文字:
<beanclass="redis.clients.jedis.Jedis">
<constructor—argv彐Iue='92．68．．28“index
<constructor—argvalue=“6379“index='“/>
</bean>
type
Java·lang.String"/>

计算机生成了可选文字:
<beanid="s01rServer"cl彐55=SO．CIen亡．SOj．-impl.HttpS01rServer>
<constructor—argvalue="h亡亡p：//92．68．．28：8080/so“
</bean>

计算机生成了可选文字:
<beanCl彐55=
0丆．sprlngframework·jdbc.datasource.DataSourceTransactionManager>
<propertyname="dataSource“e《="dataSource/>
</bean>
<tX：

计算机生成了可选文字:
<IITLPOrt汇e501_1汇Ce=
config/*·xm1“/>

计算机生成了可选文字:
bdbc
1
·driver.com．mysq1·jdbc·Driver
2jdbc·ur1=jdbc：mysq1：//10C彐1h05t：3306/b彐b彐5p0tl91?characterEncoding=UTF—8
3jdbc
·username-—-rOOt
4jdbc
·p彐55w0《：00t

计算机生成了可选文字:
<？XITLIV汇5主on='．“encoding="UTF—8"？
<！DOCTYPEconfiguration
—//mybatis.org//DTDConfig3.0//EN"
PUBLIC
"http：/,/mybatis·org/dtd/mybatis—3—config·dtd">
<con上主gu汇彐t主on>
<typeA1iases>
<packagename=CFI·亡S亡．COre·“/>
</typeAIiases>
</con上主gu汇彐t主on>

freeMarker.xml

计算机生成了可选文字:
<beanid="staticPageSerV1ce"cl彐55=cn．亡caS亡．core·SCe·StaticPageServiceImp1>
<propertyname="freeMarkerConfigurer"
<beancl彐55=0丆．sprlngframework.web.servlet.view.freemarker.FreeMarkerConfigurer"
<propertynamettemplateLoaderPathvalue="/WEB—INF/ft1/"/>
</bean>
</property>
</bean>

**Log4j.properties**

log4j.rootLogger=info, stdout

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=%d %p [%c] - %m%n

log4j.logger.com.opensymphony.xwork2=debug

log4j.logger.org.springframework.web=debug

log4j.logger.com.ibatis=debug

log4j.logger.java.sql.Connection=debug

log4j.logger.java.sql.PreparedStatement=debug

**数据库连接信息：jdbc.properties**

driverClassName=com.mysql.jdbc.Driver

url=jdbc:mysql://localhost:3306/babasport?characterEncoding=UTF-8

username=root

password=root

**Mybatis核心配置文件：SqlMapConfig.xml**

<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"<http://mybatis.org/dtd/mybatis-3-config.dtd>">

<configuration>

<typeAliases>

<package name="cn.itcast.core.pojo"/>

</typeAliases>

</configuration>

**服务消费方：Dubbo-consumer.xml**

<dubbo:application name="babasport-console"/>

<!-- <dubbo:registry address="192.168.200.128:2181" protocol="zookeeper"/> -->

<dubbo:registry address="N/A" check="false"/>

<dubbo:reference interface="cn.itcast.core.service.TestTbService" id="testTbService"

url="dubbo://127.0.0.1:20880"/>

<dubbo:consumer timeout="60000" check="false"/>

**服务提供方：Dubbo-provider.xml**

<dubbo:application name="babasport-service-product"/>

<!-- <dubbo:registry address="192.168.200.128:2181" protocol="zookeeper"/> -->

<dubbo:registry address="N/A" />

<dubbo:protocol port="20880" name="dubbo"/>

<dubbo:service interface="cn.itcast.core.service.TestTbService" ref="testTbService"/>

**全选：**

function checkBox(name,checkedOrNot) {

$("input[name="+name+"]").attr("checked",checkedOrNot);

}

**批删：**

function optDelete() {

var size = $("input[name=ids]:checked").size();

if(size <= 0) {

alert("请至少选择一个");

return;

}

if(!confirm("你确定删除？")) {

return;

}

$("#jvForm").attr("action","/brand/delBatch.do");

$("#jvForm").attr("method","post").submit();

}

**MySQL 命令操作导入表：**

mysql -uroot -p

\*\*\*\*

show databases;

use babasport191;

show tables;

source D:\....table.sql

**FastDFS上传图片java代码：**

public static String uploadPic(byte[] pic, String name, Long fileSize) throws Exception {

ClassPathResource resource = new ClassPathResource("fdfs\_client.conf");

String path = resource.getClassLoader().getResource("fdfs\_client.conf").getPath();

ClientGlobal.init(path);

TrackerClient trackerClient = new TrackerClient();

TrackerServer trackerServer = trackerClient.getConnection();

StorageClient1 storageClient1 = new StorageClient1(trackerServer, null);

String ext = FilenameUtils.getExtension(name);

NameValuePair[] meta\_list = new NameValuePair[3];

meta\_list[0] = new NameValuePair("fileName", name);

meta\_list[1] = new NameValuePair("fileExt", ext);

meta\_list[2] = new NameValuePair("fileSize", String.valueOf(fileSize));

String url = storageClient1.upload\_file1(pic, ext, meta\_list);

return url;

}

**富文本编辑器：**

1.jsp文件

$(function(){

var tObj;

$("#tabs a").each(function(){

if($(this).attr("class").indexOf("here") == 0){tObj = $(this)}

$(this).click(function(){

var c = $(this).attr("class");

if(c.indexOf("here") == 0){return;}

var ref = $(this).attr("ref");

var ref\_t = tObj.attr("ref");

tObj.attr("class","nor");

$(this).attr("class","here");

$(ref\_t).hide();

$(ref).show();

tObj = $(this);

if(ref == '#tab\_3'){

// 编辑器参数

                var kingEditorParams = {

//指定上传文件参数名称

filePostName : "uploadFile",

//指定上传文件请求的url。

uploadJson : '/upload/uploadFck.do',

//上传类型，分别为image、flash、media、file

dir : "image"//,

//                                         width : '1000px',

//                                         height : '400px'

};

KindEditor.create('#productdesc',kingEditorParams);

KindEditor.sync();

}

});

});

});

2.java文件

@RequestMapping("/uploadFck")

public void uploadFck(HttpServletRequest request, HttpServletResponse response)

throws Exception {

MultipartRequest mr = (MultipartRequest) request;

Map<String, MultipartFile> fileMap = mr.getFileMap();

Collection<Entry<String, MultipartFile>> entrySet = fileMap.entrySet();

for (Entry<String, MultipartFile> entry : entrySet) {

MultipartFile pic = entry.getValue();

String path = uploadService.uploadPic(pic.getBytes(),pic.getOriginalFilename(),pic.getSize());

JSONObject jo = new JSONObject();

jo.put("error", 0);

jo.put("url", path);

response.setContentType("application/json;charset=UTF-8");

response.getWriter().write(jo.toString());

}

}

**Redis使用：**

1.xml文件：

<bean class="redis.clients.jedis.Jedis">

<constructor-arg value="192.168.200.128" index="0" type="java.lang.String"/>

<constructor-arg value="6379" index="1"/>

</bean>

2.java文件

1）public void testJedis() throws Exception {

Jedis jedis = new Jedis("192.168.200.128", 6379);

Long pno = jedis.incr("pno");

System.out.println(pno);

jedis.close();

}

2）public void testSpringJedis() throws Exception {

System.out.println(jedis.get("pno"));

}

**修改Sku：**

function addSku(skuid) {

$("#m" + skuid).attr("disabled", true);

$("#p" + skuid).attr("disabled", true);

$("#i" + skuid).attr("disabled", true);

$("#l" + skuid).attr("disabled", true);

$("#f" + skuid).attr("disabled", true);

var url = "/sku/addSku.do";

var params = {

"id" : skuid,

"marketPrice" : $("#m" + skuid).val(),

"price" : $("#p" + skuid).val(),

"stock" : $("#i" + skuid).val(),

"upperLimit" : $("#l" + skuid).val(),

"deliveFee" : $("#p" + skuid).val(),

};

$.post(url,params,function(data){

alert(data.message);

},"json");

}

**solr配置IK分词器：**

1.schema.xml

<!-- 自定义的数据类型 支持IKAnalyzer-->

<fieldType name="text\_ik" class="solr.TextField">

<analyzer type="index" isMaxWordLength="false" class="org.wltea.analyzer.lucene.IKAnalyzer"/>

<analyzer type="query" isMaxWordLength="true" class="org.wltea.analyzer.lucene.IKAnalyzer"/>

</fieldType>

<!-- 自定义字段 name\_ik 网站查询的关键词字段 -->

<field name="name\_ik" type="text\_ik" indexed="true" stored="true" />

2.solr.xml

<bean id="solrServer" class="org.apache.solr.client.solrj.impl.HttpSolrServer">

<constructor-arg value="http://192.168.200.128:8080/solr"/>

</bean>

**ActiveMQ消息队列：**

计算机生成了可选文字:
」MS应用程序接口
ConnectionFactory（连接工厂）-
--用户用来创邃到JMS提供者的连接的被箭对象。
--连接代表了应用程序和消息服务器之间的通信链路·
Connection（连接）-
Destination（目标）-
--消息发布和接收的地点，或者是队列，或者是主题。
MessageProducer（消息生产者）-
--由会话创建的对象，用于发送消息到目标。
MessageConsumer（消息消费者）---由会话创建的对象，用于接收发送到目标的消就
Message（消息）-
-是在消费者和生产者之间传送的对象·
Session（会话）-
一表示一个单线程的上下文，用于发送和接收消息。

<http://192.168.200.128:8161/admin/> activeMQ 访问路径

tcp://192.168.200.128:61616

activeMQ.xml

<bean id="activeMQConnectionFactory" class="org.apache.activemq.ActiveMQConnectionFactory">

<property name="brokerURL" value="tcp://192.168.200.128:61616"/>

<property name="userName" vlaue="admin"/>

<property name="password" vlaue="admin"/>

</bean>

<bean id="pooledConnectionFactoryBean" class="org.apache.activemq.pool.PooledConnectionFactoryBean">

<property name="connectionFactory" ref="activeMQConnectionFactory"/>

<property name="maxConnections" value="2"/>

</bean>

<bean id="singleConnectionFactory" class="org.springframework.jms.connection.SingleConnectionFactory">

<property name="targetConnectionFactory" ref="pooledConnectionFactoryBean"/>

</bean>

<bean id="jmsTemplate" class="org.springframework.jms.core.JmsTemplate">

<property name="connectionFactory" ref="singleConnectionFactory"/>

<property name="defaultDestinationName" value="productId"/>

</bean>

<bean id="customMessageListener" class="cn.itcast.core.message.CustomMessageListener">

</bean>

<!-- 监听activeMQ中是否有消息 -->

<bean class="org.springframework.jms.listener.DefaultMessageListenerContainer">

<property name="connectionFactory" ref="singleConnectionFactory"/>

<property name="destinationName" value="productId"/>

<property name="messageListener" ref="customMessageListener"/>

</bean>

Freemarker.java文件的写法：

public void staticizeProductPage(Map<String, Object> rootMap, String id) {

String pathname = "/html/product/"+ id +".html";

String realPath = servletContext.getRealPath(pathname);

File file = new File(realPath);

if (!file.exists()) {

file.mkdirs();

}

Writer out = null;

try {

Template template = configuration.getTemplate("product.html");

out = new OutputStreamWriter(new FileOutputStream(file), "UTF-8");

template.process(rootMap, out);

} catch (IOException e) {

e.printStackTrace();

} catch (TemplateException e) {

e.printStackTrace();

} finally {

if (null != out) {

try {

out.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**判断是否登录：**

$(function(){

var options = {

url:"http://localhost:8180/isLogin.aspx",

type:"post",

dataType:"jsonp",

success:function(data){

if (data) {

$("#login").hide();

$("#regist").hide();

} else {

$("#logout").hide();

$("#myOrder").hide();

}

}

}

$.ajax(options);

});

**Java文件中：**

@RequestMapping(value="/isLogin")

public @ResponseBody MappingJacksonValue isLogin(String callback,

HttpServletRequest request, HttpServletResponse response)

throws Exception {

Integer result = 0;

String csessionID = sessionProvider.getCsessionIDFromRedis(RequestUtil.getCSESSIONID(request, response));

if (null != csessionID) {

result = 1;

}

MappingJacksonValue mjv = new MappingJacksonValue(result);

mjv.setJsonpFunction(callback);

return mjv;

}

@RequestMapping("/uploadPic")

public void uploadPic(@RequestParam(required = false)MultipartFile pic, HttpServletResponse response)

throws Exception {

String path = uploadService.uploadPic(pic.getBytes(),pic.getOriginalFilename(),pic.getSize());

System.out.println(path);

JSONObject jo = new JSONObject();

jo.put("path", path);

response.setContentType("application/json;charset=UTF-8");

response.getWriter().write(jo.toString());

}

计算机生成了可选文字:
原因，
强制关闭Redis快照导致不能持久化。
解决方案，
彳iconfigsetstop-writes-on-bgsave-errorno命专，舌，关闭酉己置I页stop-writes-on-bgsave-error角晡央讠亥问题。
root@ubuntu:/usr/local/redis/bin#/redis-cli
127．0．0．1：6379>configsetstop-writes-on-bgsave-errorno
OK
127．0．0．1：6379>IpushmyCoIour"red"
(integer)1

计算机生成了可选文字:
人0下
〔榧出购
2〕畢出的购是亨或不是室·，
3〕加丰是刨一个莉车桐
4〕將当前的黧0昷址人购物车“
引保存购车到（。焱-再与叵0览器疒
引存购车到巾、窒（“0
1〕〔出购车，

**Redis单机版搭建：**

计算机生成了可选文字:
tot彐1144
一汇W一汇W一汇一
一汇W一汇W一汇一
一汇W一汇W一汇一
一汇W一汇W一汇一
d汇》了X汇WX汇
一x·
一汇W一汇W一汇一
一汇W一汇W一汇一
一汇W一汇W一汇一
一汇W一汇W一汇一
一汇W一汇W一汇一
一x·
一x·
一汇》了X汇WX汇
一x·
一汇W一汇W一汇一
d汇》了X汇》了X汇
一x·
d汇》了X汇》了X汇
一x·
d汇》了X汇WX汇
一x·
1
1
1
1
6
1
1
1
1
1
1
1
1
1
2
10
5
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
汇00t
25890
53
1439
1487
4096
11
151
4223
5201
41403
271
280
281
7109
4096
4096
4096
p汇
p汇
p汇
p汇
Dec
p汇
p汇
p汇
p汇
p汇
p汇
p汇
p汇
p汇
Dec
p汇
p汇
1
1
1
1
9
1
1
1
1
1
1
1
1
1
9
1
1
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
2015
00—RELEASENOTES
BUGS
CONTRIBUTING
COPYING
dep5
INSTALL
彐k巳f主1巳
MANIFESTO
README
汇d主5。conf
runtest
runtest—cluster
runtest—sentinel
·C
t巳5t5
ut主15
[root@hostname
redis—3.0
123Ro102《：

Gcc+ 编译c++的工具

1.Makefile执行make需要的文件：会多出来6个执行脚本

2.redis.conf配置文件

3.src源码和编译后的文件位置

步骤：

1.编译：

计算机生成了可选文字:
1·root一root一25吕90一Apr一一上一一．一20工5一00，REL．_凵、0了=
—rw—rw—r—
—rw—rw—r—
rwxrwxr
—rw—rw—r—
—rw—rw—r—
—rwxrwxr—x.
—t¯WXrWXr—X.
—rwxrwxr—x.
—rw—rw—r—
rWXrWXr—X.
rwxrwxr
1
1
1
6
1
1
1
1
1
1
1
1
2
10
5
root
root
root
root
root
root
root
1·root
root
root
root
root
root
root
root
151APr
S201APr
271APr
280APr
281APr
re市5一3．0．0〕#make
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
53APr
1439
APr
1487
APr
4096
APr
11APr
4223
APr
41403
APr
7109APr
4096APr
4096APr
4096
APr
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
2015
BUGS
2015
CONTR工日UTING
2015
COPYING
201s
INSTA±L
201S
201sMakefile
MANIFESTO
2015
201s
README
S．conf
2015
2015
runtest
runtest—cluster
2015
runtest—sentinel
2015
，conf
2015
201s
2015
2015
Croot@hostname

2.安装：

计算机生成了可选文字:
Croot@hostnamere市5一3．0．0]#makePREFIX=/redis/63790nstall

3.复制配置文件：

计算机生成了可选文字:
root@hostnameres．3．0．0〕孝CPredis.conf/reds台379/
E
root@hostnamered5一3．0．0〕#

4.配置后台启动：

计算机生成了可选文字:
RedisdoeSnotrunasa
use'yes'、fyouneedt．
Edefaul
#N0teth
diswi11writeapidfie1n/var/run/redis.pidwhen—nized.
daem
·n1ZeeS
#whenrunningÄ1ized,Redswritesapfdfen/var/run/redis.pfdby
INSERT

5.

./bin/redis-cli -h 192.168.200.128 -p 6379

防火墙关闭：

计算机生成了可选文字:
cat-7.O.47/bnn/tomcat-
[root@localhostsrc]#serviceiptables
Table:fi1ter
ChainINPUT(policyACCEPT)
numtarget
protOPtSOUrce
1
2
3
4
5
ACCEPT
ACCEPT
ACCEPT
ACCEPT
RE〕ECT
a〕1
all
tC
0．0．0．0/0
0．0．0．0/0
0．0．0．0／0
0．0．0．0/0
0．0．0．0／0
chainFORWARD(policyACCEPT)
numtarget
1
REJECT
protOPtsource
a11
0．0．0．0/0
status
destnnatlon
0．0．0．0/0
0．0．0．0／0
0．0．0．0／0
0．0．0．0/0
0．0．0．0/0
destination
0．0．0．0／0
destination
Stop
StateRELATED，ESTABLISHED
StateNEWtCPdpt：22
re〕ect-withicmp-host—prohibited
reject—withicmp—host—prohibited
chainOUTPUT(policyACCEPT)
numtarget
protOPtsource
[root@localhostsrc]#Serv1（enptaeS
iptables:Flushingfi
iptables:SettingchainstopolicyACCEPT：fi1ter
iptables:unloadingm
[root@localhost
iptables
[root@localhost
Croot@localhost
nptables
Croot@localhost
src]#
0：off
0：off
src]#
chkconfi
、tables
—list
4：
src]#chkconfigiptablesOff
src]#chkconfi
itables
—list
：0
：0
：0
on
4：off
5
：on
5：off
OK
OK
OK
6：off
6：off

Nginx搭建：

一：生成MakeFile make make install

二：转发 /nginx1.8/conf/nginx.conf

1.反向代理

计算机生成了可选文字:
upstreamtomcat5巳汇V巳汇{
5巳汇v巳汇192．168．200．132：8080；
5巳汇V巳汇
listen
5汇V汇namet汇彐Ck汇；
location/g汇oupl/00/{
汇00t/fastdfs/storageData/fdfs5t0汇彐g巳/d彐t彐；
ngxfastdfsmodul巳；
loc彐t主0
P汇OXY一p彐55http：//tomcat5巳汇v巳汇；
tm1；
汇00t
indexindex．htmlindex.htm;

2.负载均衡

计算机生成了可选文字:
upstreamtomcat5汇V汇{
5巳汇v巳汇192．168．200．132：8080w巳主ght：10；
5巳汇v巳汇192．168．200．133：8080w巳主ght：20；
5巳汇V巳汇
listen
5汇V汇namet汇彐Ck汇；
location/g汇oupl/00/{
汇00t/fastdfs/storageData/fdfs5t0汇彐g巳/d彐t彐；
ngxfastdfsmodul巳；
loc彐t主0
P汇OXY一p彐55http：//tomcat5巳汇v巳汇；
indexindex．htmlindex.htm;

三：故障移除

四：恢复添加

高可用：

安装keepalived-1.2.13

rpm -ivh keepalived-1.2.13-5.el6\_6.i686.rpm

rpm -ql keepalived 查看安装文件的目录

cd /etc/keepalived/

service keepalived start

service keepalived stop

/sbin/ip add show eth2

MySql集群搭建：

一：mycat配置过程

1.设置用户名及密码

计算机生成了可选文字:
total80
一rwxrwx·
-rwx·
931Sep
一rwxrwx·
_t-WX·
1109
11：19
一r¯WXrWXrWX·
943sep
11：19
_rwxrwx·
1109
-rwx·
一rwxrwx·
一t*WX·
一rwxrwx·
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
root
root
root
root
root
root
root
root
root
root
root
root
root
root
r00t
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
root
88
sep
340
sep
51
〕an
439
Sep
1647
〕un
15
sep
31
〕an
102
Sep
S126
〕an
4781
〕an
413
Sep
75
Sep
S1sep
2711
〕an
4152
Sep
1
1
10
1
1
17
1
1
1
1
1
11：19
11：19
10巧5
11：19
11：19
2015
111.
1109：28
11
11．
11．
423，
'19
：28
：11
：19
．19
、19
'32
autopartltion—long.txt
dnindex.properties
ehcache.xml
index—to—charset.properties
log4$
artltion—hash—int.txt
ar11on一，×
part•t10n—range—mo.txt
SCema·×m
sequence—db—conf·propertieS
Seuene_time_conf、properties
甲户
111：20wrapper.conf

mysql -uroot -proot -h127.0.0.1 -p8066