前台WEB层框架功能和开发说明

|  |  |
| --- | --- |
| 版本号 | 0.1 |
| 文档标识 |  |
| 日期 | 2016年10月13日 |

# 总体概况

前台WEB层涉及到的主要框架结构如下：



* 框架在服务端支持页面和组件模型、通过模板引擎支持页面模板和组件模板的解析和输出。
* 框架提供一套响应式前端UI，支持多终端、多平台、多浏览器兼容
* 框架提供JS脚本库，封装了DOM操作、事件绑定、与服务端交互等功能
* 框架基于组件模型和前端UI封装一系列核心的动态组件提供给开发使用
* 前端和服务端使用统一的数据总线(IData和IDataset)进行数据交互

# 前台界面框架

前台界面框架基于模板化、组件化技术实现界面生成和界面组装。

## 运行流程

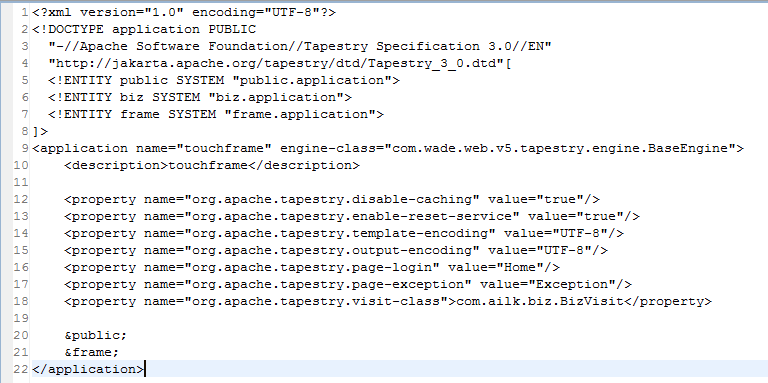


1. 总体流程



(b)获取页面对象

* 启动应用服务器，加载部署的WEB应用
* WEB应用寻找部署描述符（web.xml），做为程序运行的入口,并加载servlet、filter、monitor、session的一些配置
* 客户端提交访问路径发起请求到服务器端，服务器端在web.xml中寻找到客户端访问的Servlet
* 通过app Servlet中配置的名称找到同名的.application配置文件,如quickstart.application
* 加载.application中配置，包括Engine、Visit、Page的一些配置，并初始化Engine,Page等实例
* 根据客户端提交访问的页面别名，在.application文件中找到注册的页面别名对应的page文件
* 通过page文件找到对应的java，html文件，进行页面、组件的执行和界面输出



(c)application入口文件示例

## 页面开发

* 一个页面由一个页面模板(.html)，一个页面规范(.page),一个页面类 (.java)组成
* 每一个页面有一个唯一的名字，页面名字用来定位页面规范和页面模板，页面名字定义在application配置文件中
* 页面规范会定位页面模板对应的页面类，以及定义页面模板中用到的属性方法，
* 页面类中定义需要给HTML模板使用的属性和监听方法

页面开发基本步骤：

1. 准备页面需要的资源，包括页面模板(.html)、页面规范(.page)、页面类(.java)，以及具体的业务逻辑。
2. 注册页面，在.application文件中将写好的页面规范(.page)注册到.application中，并定义唯一的名字。
3. 配置测试菜单，在数据库或菜单页面中新加菜单项，点击菜单打开页面进行调试



.application文件中注册页面名的配置示例

1. 页面规范编写

页面规范是扩展名为 .page 的文本文件，它作为衔接页面模板和页面类的枢纽，需要定义页面类、页面模板的路径，同时需要定义页面模板所用到的所有对象，如：



补充说明：

页面规范中能够定义页面类的路径，页面规范对应的页面模板的映射规则就是页面规范文件同目录下同文件名(不含扩展文件名)的.html文件

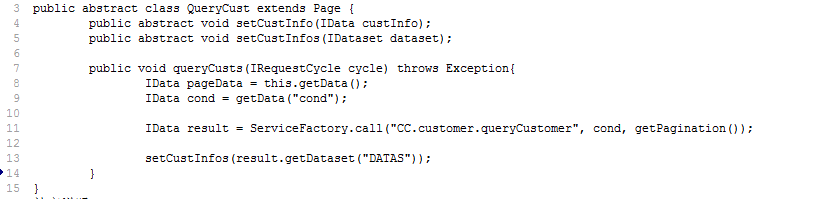
一个页面规范必须对应一个页面模板和一个页面类

一个页面模板必须对应一个页面规范，而一个页面类可以对应多个页面规范，也就是说多个页面模板可以共用一个页面类

1. 页面类编写

页面类作为MVC中的控制层，主要是用来衔接业务逻辑与页面之间的枢纽，将页面的请求提交给业务逻辑，由业务逻辑处理后反馈结果给页面。

页面类主要编写响应页面的事件以及反馈给页面的结果对象，如：



补充说明：

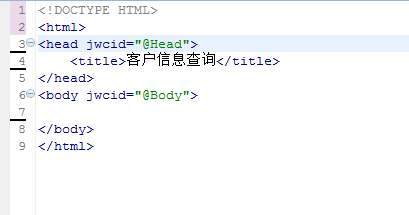
页面类必须继承wade-web中的com.ailk.web.BasePage类或者其子类，否则无法和页面模板、页面规范组成一个功能

定义的提供给页面模板使用的属性方法需要使用抽象方法

定义的响应事件，必须带有参数IRequestCycle cycle，否则页面中无法识别该方法是基于页面的响应事件

1. 页面模板编写

页面模板主要是将业务逻辑中获取的数据展现在页面中，以及对页面操作的一些控制，页面模板基本结构如下：



补充说明：

页面中模板中用jwcid标识的表示动态内容，head表示页面模板的相关资源及说明，其中Head组件主要是导入公用的一些资源，如JavaScript库，Body组件对于大部分组件来说是组件容器的支撑

ognl标记表示用Java表达式来动态表达内容

## 组件开发

前台WEB组件是对前台公用界面和功能可重用部分的封装实现。

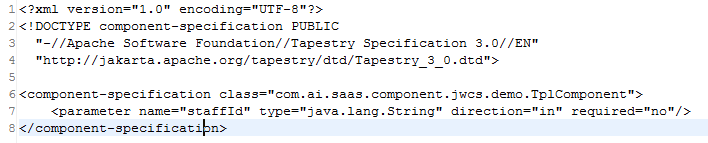
组件开发分为两种：

* 单一型组件，一般基于最小粒度的组件使用单一型组件，该类型的组件的HTML代码在组件类中编写，所以组件只包含组件类，组件规范，单一型组件又分为表单型组件和普通单一组件。
* 组合型(模板)组件，一般基于一块页面区域使用组合型组件，该类型的组件的HTML代码在组件模板中实现，所以组件包含组件模板，组件类，组件规范。

组件开发基本步骤：

1. 准备组件需要的资源，包括组件模板(.html)、组件规范(.jwc)、组件类(.java)。
2. 编写组件，对于组合型组件需要编写组件模板，对于单一性组件则不需要。
3. 注册编写好的组件注册到组件库中，在. library文件中将写好的组件规范注册到. library中，并定义唯一的组件名字。
4. 注册组件库，在.application文件中将写好的组件库文件注册到.application中，并定义唯一的组件库名字。
5. 在页面中使用jwcid=”@组件库注册名:组件注册名”的方式引入组件，并进行调试。
6. 组件规范编写

组件规范作为衔接组件模板和组件类的枢纽，需要定义组件类、组件模板的路径，同时需要定义组件模板所用到的所有对象，如：



补充说明：

组件规范中能够定义组件类的路径，组件规范对应的组件模板的映射规则就是通过组件规范文件同目录下同文件名(不含扩展文件名)的.html文件。

组件中定义的所有属性，都可以开发给使用者，这样引用组件时可以随意定义组件的属性达到动态使用组件的效果。

组件规范配置属性说明：

component-specification 组件定义节点，包括以下常用属性：

class：组件类的路径

allow-body：yes|no 是否允许组件中有body

allow-informal-parameters： yes|no 是否允许组件有非正式参数

parameter：组件参数配置节点，包括以下常用属性

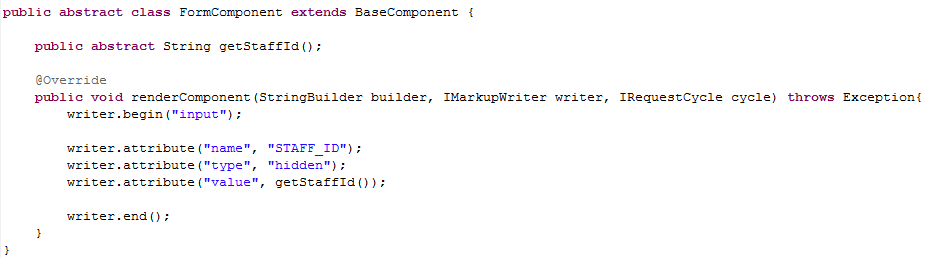
required： yes|no 该参数是否必须

direction：form|in 参数的输入输出形式，form表示为输入/输出型，in表示输入型

default-value：参数为空时的默认值

1. 组件类编写（单一型）

单一型组件通常用于表单元素的组件封装，并在类中直接输出要构造的html结构



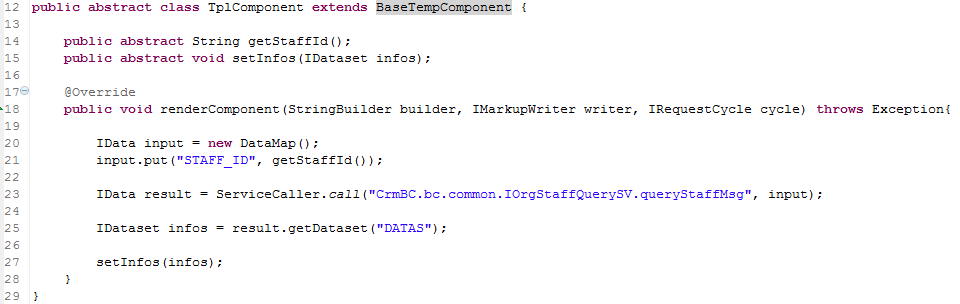
补充说明：

组件类必须继承com.ailk.web.BaseComponent或者其子类。

组件类必须实现renderComponent(StringBuilder builder, IMarkupWriter writer, IRequestCycle cycle) throws Exception方法

1. 组件类编写(组合型)

组合型组件的组件类是用来衔接业务逻辑与组件之间的枢纽，将处理后的结果反馈给组件，并将数据绑定到组件模板上，如：



补充说明：

组件类必须继承com.ailk.web.BaseTempComponent或者其子类，否则无法和组件模板、组件规范组成一个功能。

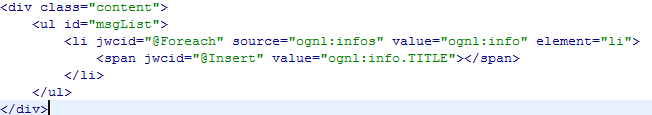
组件类必须实现renderComponent(StringBuilder builder, IMarkupWriter writer, IRequestCycle cycle) throws Exception方法，该方法用来展现组件模板中的效果

1. 组件模板编写(组合型)

只有组合型组件才有组件模板

组件模板是与组件规范文件在同一目录(类路径下)、同名且扩展名为html的文件

组件模板通常是一个HTML片段，主要是将组件类调用业务逻辑中获取的数据展现在页面中，组件模板结构如下：

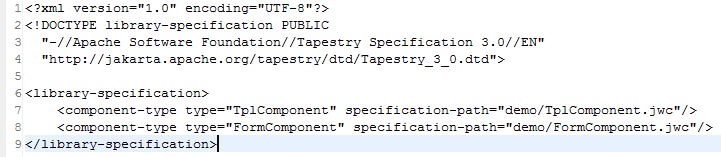


补充说明：

组件模板中可以嵌套使用其他组件，嵌套的除核心组件库外，其他的组件需要在组件库中定义。

1. 组件注册

注册组件到组件库中(.library)，如：



注册组件库到全局配置中(.application)，如：



## 数据交互

1. 数据格式

前台的页面类，组件类中统一使用

com.ailk.common.data.impl.DataMap

和

com.ailk.common.data.impl.DatasetList

两种数据类向前端返回数据，通常在页面和组件规范定义中使用对应的接口类定义数据类型，对应的接口类:

com.ailk.common.data.IData

com.ailk.common.data.IDataset

前端页面中对应的javascript数据结构，分别是

Wade.DataMap 和 Wade.DatasetList

1. 前端和前台数据交互

前端界面与前台服务端进行数据交互主要有以下两种方式：

1. ognl数据绑定方式
2. 页面或组件可以在java类中设置绑定的数据，在页面或组建模板中通过ognl绑定页面或组建规范中定义的数据，在页面的监听方法执行或组件的render执行就可以将动态数据加载或展现在前端。
3. 前端通过ajax方式对页面进行组件刷新或局部Part刷新时，也可以进行同样的ognl数据绑定，并将数据绑定后生成的html片段返回给前端。
4. JSON数据交互方式

前端使用ajax方式与前台服务端交互时，页面类或HttpHandler类可以通过setAjax(IData)或setAjax(IDataset)的方式，向前端返回JSON格式数据。

框架默认把返回数据解析成JS的 $.DataMap或$.DatassetList数据类型，也支持自定义的方式进行解析，具体参见API文档。

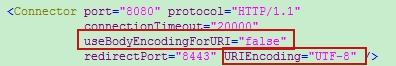
## 常见问题

1.中文和全角字符编码问题

要求所有的HTML、配置文件、java代码均等文本文件均使用UTF-8字符集编码，在开发时，需要将Eclipse工程的默认文本编码设置为UTF-8。

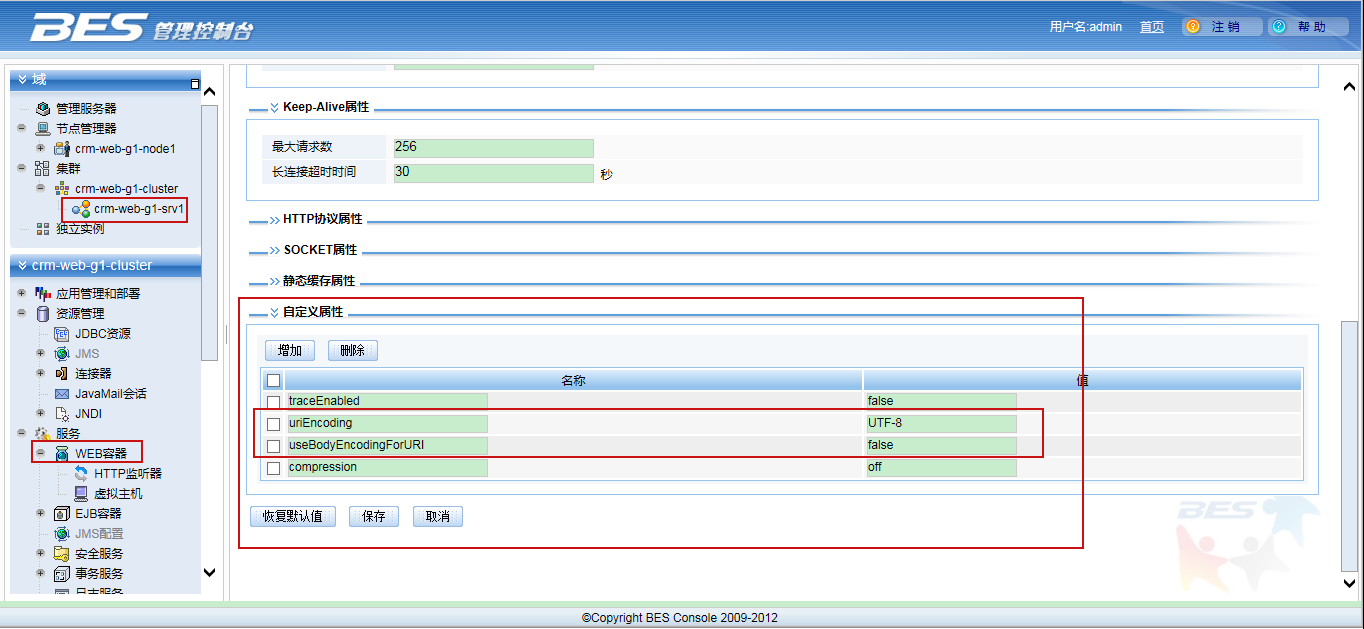
服务端配置：

本地开发时，需要在Tomcat的服务配置server.xml里配置URI参数字符集：



配置： useBodyEncodingForURI="false" URIEncoding="UTF-8"

测试、生产环境，需要设置在BES中间件集群服务的WEB容器自定义属性中配置URI字符集参数：



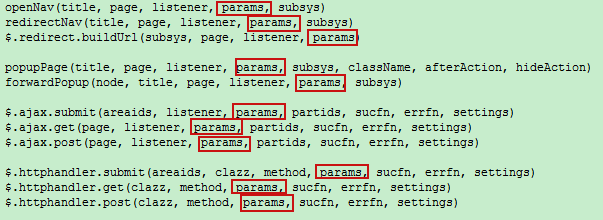
配置自定义属性：useBodyEncodingForURI为false， uriEncoding为UTF-8

测试、生产环境需要配置JVM文件编码和中文参数配置：

-Dfile.encoding=UTF-8 -Duser.language=zh -Duser.region=CN

中文和全角字符的传值取值：

对于框架中以下常用JS的API自定义参数部分，如果有中文或全角字符的，均需要做编码处理：



对于其它情况http get请求中有中文或全角参数的，也应该要做编码处理

编码方式(JS)："&CUST\_ID=1234456&CUST\_NAME=" + encodeURIComponent("张三")

编码方式(ognl): templateFile="ognl:'examples/template/' + urlEncode('任务模板') + '.xls'"

编码方式(java): java.net.URLEncoder.encode(str, "UTF-8")

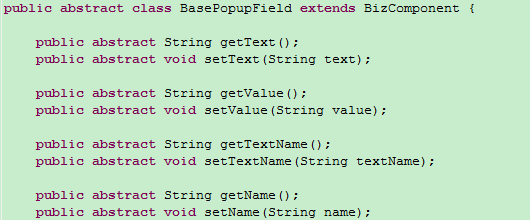
服务端取值时，不需要再做任何解码处理。

2.页面和组件实例缓存问题

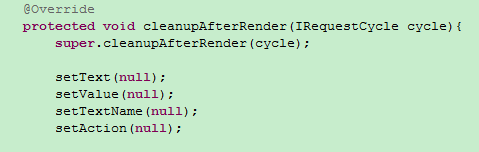
Tapestry模板引擎默认会对页面和组件的模板、页面和组件的实例进行缓存以提升性能、通常在开发时会设置JVM参数-Dorg.apache.tapestry.disable-caching=true设置来禁用这类缓存以方便开发调试，但在测试、生产环境则不应禁用它。

对于组件来说，缓存了实例则意味着同样会缓存实例的属性值。因此对于涉及业务数据属性的组件，需要在重写组件的cleanupAfterRender方法，将业务数据属性清空。

例如：组件BasePopupField有一系列属性



在组件重写的cleanupAfterRender方法里进行清空：



若不做清除处理，则存在组件加载未给属性重新设值的情况下仍然使用上一次属性值的情况。

3.BES中间件配置文件缓存问题

前台界面组件包含组件属性定义 .jwc 文件，组件库定义 .library 文件，在编译发布时，这些文件都会统一打包到jar包中去。jar包中也可能包含其它一些功能的xml或properties配置文件。

BES中间件在应用发布后，会把jar包中的这些配置文件提取出来并缓存到以下目录：

domain/nodes/server-node/instances/server实例/tmp/jsp/apps/部署应用/loader

重新发布应用时，该目录下的文件可能不会被更新，会导致界面组件属性定义、组件库定义文件修改后发布不生效的情况。

遇到此类情况，可以删除该目录下缓存的配置文件后重新发布应用解决。

4.在线文档地址： http://doc.wadecn.com:9000/wade5

# 统一会话管理

## 实现原理



* 统一会话的基本原理是在过滤器中自定义实现Request请求对象中的createSession和getSession实现，将会话数据存到远端分布式缓存中，实现WEB应用无状态化。
* 自定义HttpSession实现HttpSession,Serializable接口，并支持新建和修改判断。
* 每个HTTP请求线程创建或从分布式缓存中获取HttpSession数据后存放在线程上下文，在HTTP请求线程完成后，判断HttpSession是否为新建或有修改来决定是否需要做回存处理，避免和缓存的频繁交互。

## 服务部署

统一会话管理需要单独部署会话Redis缓存服务，由于会话数据属于关键性数据，在生产环境进行会话Redis缓存部署时，通常采取双中心进程做主从互备，并在主节点故障时自动切换到备节点。

测试开发环境只需部署单节点即可。



双中心主备节点同步

* 正常情况下，Web应用连接到两个中心的主节点，会话数据根据会话标识HASH散列均匀分布在8个主节点进程上
* 当主节点故障时，Web应用与主节点心跳检测失败，自动切换连接到主节点对应的备节点。

## 相关配置

1.会话选项配置

会话选项配置在WEB应用的sna.properties配置文件中，配置内容如下：



其中：

sna.app.name 是接入的系统名，只需要在NGBOSS外框应用配置

sna.session.cache 是使用的缓存类型，固定配置为redis

sna.session.timeout 为会话超时时间，单位为秒，默认不配置为180秒，若要配置其它值，则需保持每个WEB应用下的配置一致。

1. Web过滤器配置

WEB应用需要配置统一会话管理的过滤器

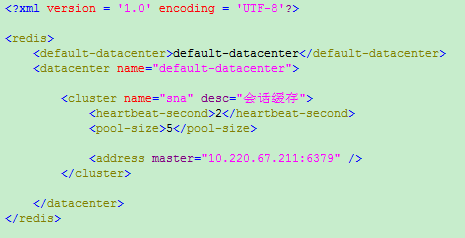


统一会话管理过滤器要配置在其它WEB过滤器特别是SessionVerifyFilter过滤器之前。filter-mapping 配置需要做统一会话处理的servlet-name。

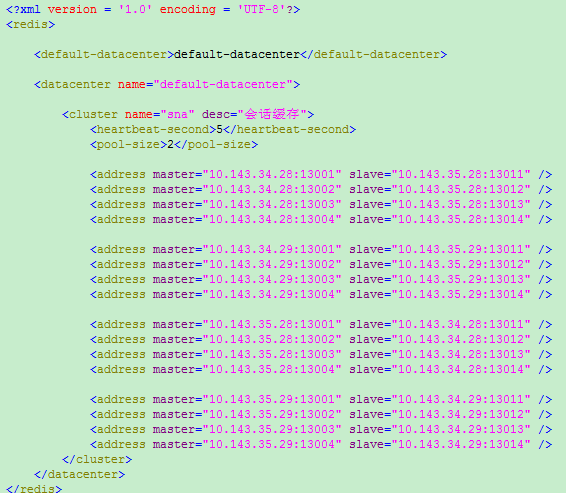
3.缓存服务连接配置

缓存服务连接配置在WEB应用的redis.xml配置文件中

开发测试环境配置：



生产环境配置：



# 文件上传下载

## 功能说明



* 客户端上传下载时与服务端FileServlet交互
* FileSerlvet反射调用IFileAction接口的实现类，完成文件的上传、下载、查询、删除等操作
* IFileAction接口的实现类读取数据库表中的FTP站点配置，将文件上传到FTP站点或对FTP站点的文件进行下载、删除等操作。
* IFileAction接口的实现类对文件记录表进行操作，维护上传文件的信息。

## 相关配置

1.Web Servlet配置

web.xml 中配置FileServlet



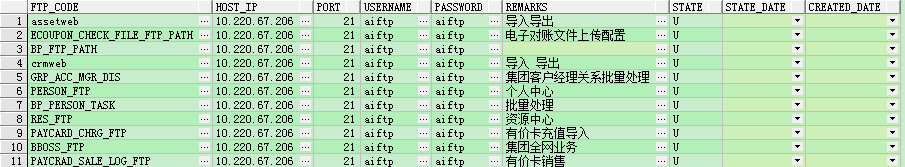
2.应用接口实现类配置

global.properties中配置：

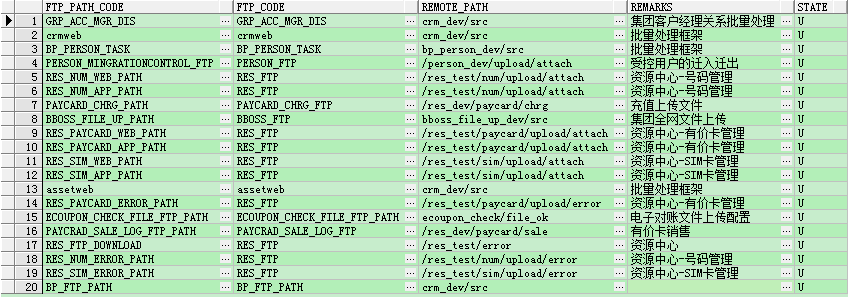
fileman.fileaction=com.ai.baseframe.ipaas.ftpmgr.FtpFileAction

3.数据库表参数配置

base.bs\_ftp表配置站点信息



base.bs\_ftp\_path表配置路径配置



## 文件记录

文件上传后会在base.wd\_f\_ftpfile表中插入文件信息数据，其中：

FILE\_ID 为文件标识

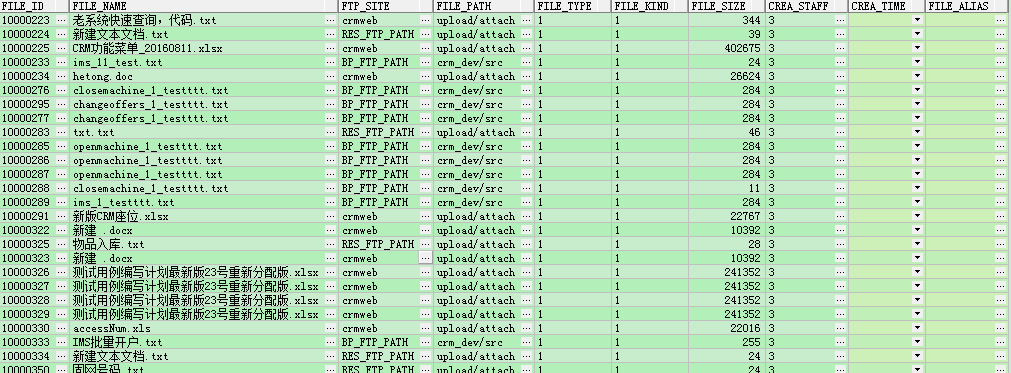
FILE\_NAME 为真实文件名

FTP\_SITE 对应 base.bs\_ftp\_path表中的路径配置编码

FILE\_PATH 为文件上传相对于站点路径的相对路径

FILE\_SIZE 为文件大小 单位为byte

base.wd\_f\_ftpfile表数据



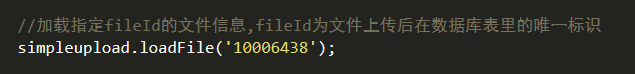
## 组件使用

单文件上传组件：



ftpCode设置为bs\_ftp\_path中的路径编码

单文件上传组件加载已上传文件信息：



多文件上传组件：



ftpCode设置为bs\_ftp\_path中的路径编码

多文件上传组件加载已上传文件信息：



更多详细用法参考在线文档

# 导入导出

## 异步任务调度

### 功能说明



* 异步任务调度是指进程间通过消息队列服务通信，实现由一个进程发起对其它进程上的异步任务执行调用。
* 消息队列服务由MQ-Server守护进程和Redis缓存服务构成。
* 客户端进程以JVM参数 –Dwade.server.name=xxx 的值作为进程名标识。
* MQ-Server守护进程从数据库加载异步任务配置到Redis缓存、从cluster-definition配置文件加载客户端进程的集群配置到Redis缓存。
* 客户端进程根据集群配置监听Redis缓存上的集群消息数据队列和私有消息数据队列。
* 消息生产者进程发起任务调用时，发起进程将要调用的任务标识和调用参数插入到Redis缓存目标消息队列数据中。
* 消息消费者进程的监听读取到消息队列的任务调用消息，根据任务标识从Redis缓存中获取对应的任务配置信息，发起对任务类的反射调用。
* 异步任务执行过程中，任务执行进程将异步任务执行的状态和异常信息发送到Redis缓存的日志队列，MQ-Server守护进程读取队列数据进行异步任务执行状态日志记录和状态更新。

### 服务部署

消息队列服务应用目录结构：

|-app

|---server

|------bin

|---------start-redis.sh //启动redis缓存脚本

|---------stop-redis.sh //停止redis缓存脚本

|---------start-server.sh //启动MQ-Server进程脚本

|---------stop-server.sh //停止MQ-Server进程脚本

|------log //日志目录

|-db //redis缓存持久化存储目录

|-etc //配置文件目录

|---acl.conf //Redis缓存访问控制配置

|---redis.conf //Redis缓存服务配置

|---cluster-definition.xml //客户端进程集群配置

|---database.xml //MQ-Server数据库连接配置

|---redis.xml //MQ-Server Redis缓存连接配置

|---log4j.properties //MQ-Server 日志输出配置

|---

|-lib //MQ-Server服务用到的jar包目录

|-sbin //应用程序和公用脚本目录

|---cronolog //日志切割程序

|---redis-server //Redis缓存服务程序

|---redis-cli //Redis缓存客户端程序

|---monitor\_process.sh //应用进程检查脚本

|---setEnv.sh //应用环境变量设置脚本

消息队列服务的守护应用只能启动一个实例，在生产环境通常做一主一备的冷备模式。

Redis服务部署

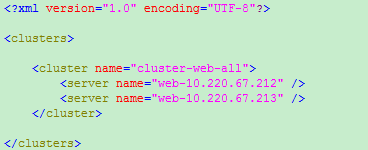
消息队列服务的Redis缓存可以和后台守护进程部署在一起也可以单独部署，生产环境应该使用主备模式。

### 相关配置

1.MQ-Server服务进程配置

a) 应用进程集群配置

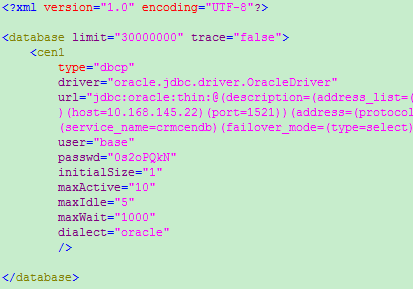
cluster-definition.xml配置客户端进程归属集群。



其中：server节点的name属性为客户端进程的-Dwade.server.name JVM参数值

b) 数据库连接配置

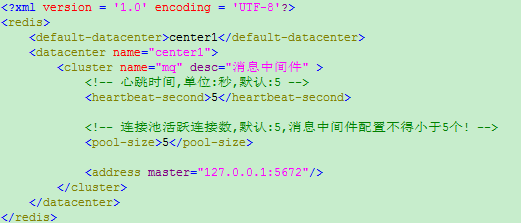
database.xml配置MQ-Server进程的数据库连接



其中： url为oracle jdbc连接串， user为 数据库用户名，passwd是数据库用户密码，initialSize为初始连接数，maxActive为最大连接数。

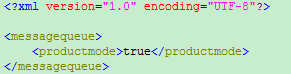
c) Redis服务器连接配置

redis.xml配置MQ-Server进程的Redis缓存连接配置



1. 应用进程配置
2. MQ客户端配置

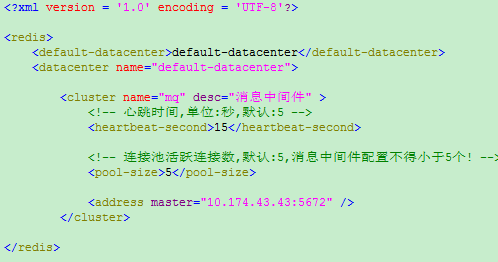
mq-client.xml 中配置是否为生产模式



测试、生产环境配置为true,本机开发配置为false,配置为false时候，所有的异步任务调用均会回调到发起进程，便于开发调试。

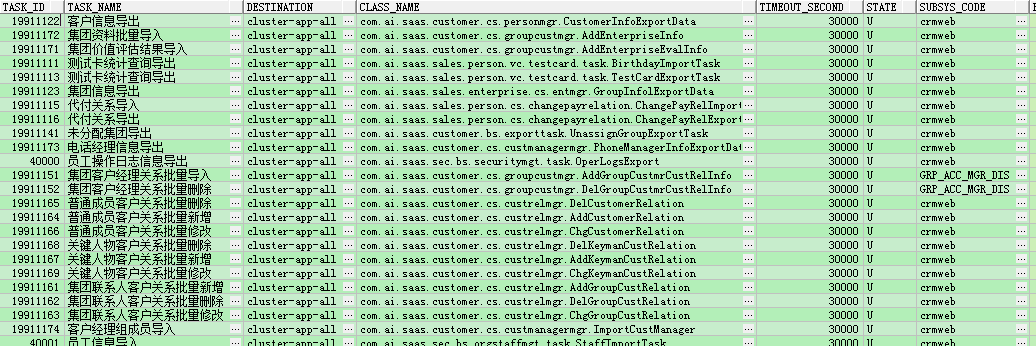
1. Redis服务器连接配置

redis.xml配置应用到消息队列服务Redis缓存的链接配置



1. 数据库表参数配置

base.wd\_m\_asynctask表配置



base.wd\_m\_asynctask表配置异步任务信息，包括：

TASK\_ID: 异步任务标识

TASK\_NAME: 异步任务名称

DESTINATION: 异步任务执行目标应用集群

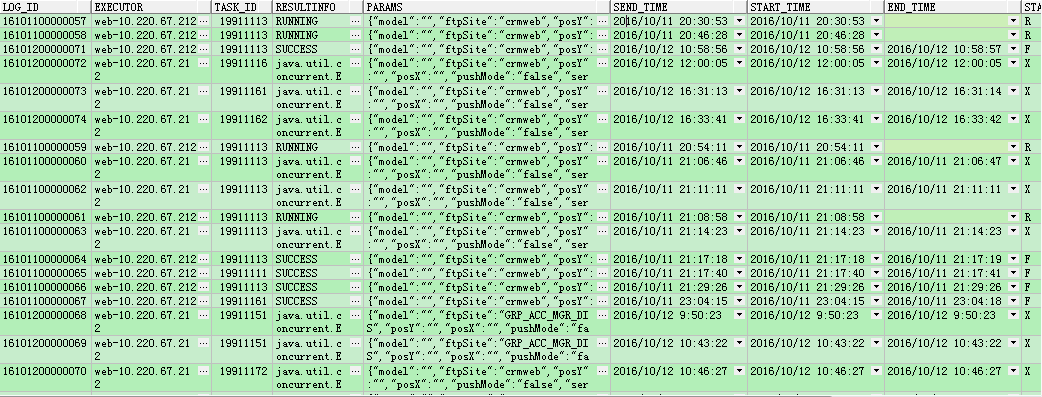
CLASS\_NAME: 异步任务实现类

TIMEOUT\_SECEND: 异步任务执行超时时间，单位秒

新增异步任务配置需要配置到该表，MQ-Server守护进程每5分钟重新加载一次任务配置到Redis缓存。

### 日志记录

base.wd\_m\_asynctask\_log异步任务执行日志：



异步任务执行状态记录到base.wd\_m\_asynctask\_log表中，其中：

LOG\_ID是日志流水

EXECUTOR 是具体执行任务进程的进程标识

TASK\_ID是对应任务标识

RESULTINFO 是执行状态，包括以下几种状态：

ACCEPT: 消息队列服务已经接收到任务调度消息

RUNNING: 任务调度消息已经发送给执行进程

SUCCESS: 任务执行成功完成

错误堆栈： 任务执行异常

PARAMS 为任务执行的参数集，为JSON格式字符串

SEND\_TIME 任务调度消息发送时间

START\_TIME 任务开始执行时间

END\_TIME 任务执行完成时间

STATE 任务执行状态，R –正在执行, X - 失败, F - 完成

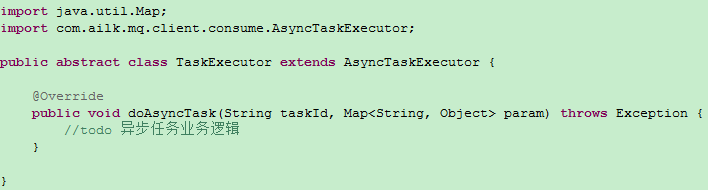
### 任务开发

1. 实现异步任务逻辑类

异步任务开发只需要继承com.ailk.mq.client.consume.AsyncTaskExecutor类，实现服务类的抽象方法:

doAsyncTask(String taskId, Map<String, Object> param) throws Exception

即可。



1. 将异步任务信息配置到异步任务配置表

需要将异步任务信息配置到 base.wd\_m\_asynctask 异步任务配置表

并等待MQ-Server进程加载新的任务配置数据

1. 发起异步任务调用

异步任务调用类：

com.ailk.mq.client.produce.AsyncTaskMultiCast

组播，集群中所有成员都会收到消息并执行

com.ailk.mq.client.produce.AsyncTaskClusterCast

按照轮询规则，下发给集群下一实例执行

com.ailk.mq.client.produce.AsyncTaskUniCast

给指定实例下发异步任务

API:

AsyncTaskMultiCast.sendAsyncTask(String taskId, Map<String, Object> param)

AsyncTaskClusterCast.sendAsyncTask(String taskId, Map<String, Object> param)

AsyncTaskUniCast.sendAsyncTask(String taskId, Map<String, Object> param)

## 导入导出功能

导入导出功能是在异步任务调度的技术上实现的二次开发和封装，它结合了文件上传下载，异步任务调度，Excel解析生成的功能。

### 功能说明



* 前台用户将数据文件上传到文件服务器，并生成file表的文件记录
* 上传完成后，前台进程根据对应的导出任务标识发起导入异步任务调用，同时将上传的数据文件标识、数据解析配置xml文件路径和其它业务参数等信息作为异步任务执行参数发送给导入任务执行进程。
* 导入任务执行进程响应任务执行，并获得传入的参数，从文件服务器获取数据文件，并根据数据解析配置文件的配置解析数据文件中的数据，得到要执行导入的数据集。
* 导入任务执行导入的业务逻辑，调用业务服务将数据集入库或进行其它处理。
* 导入任务执行完成后，更新导入日志状态，如果有部分数据导入失败，则生成导入失败的数据文件并上传到文件服务器。
* 前台检测导入任务执行完成，提示用户成功或失败，并将部分失败的数据文件提供给用户下载。



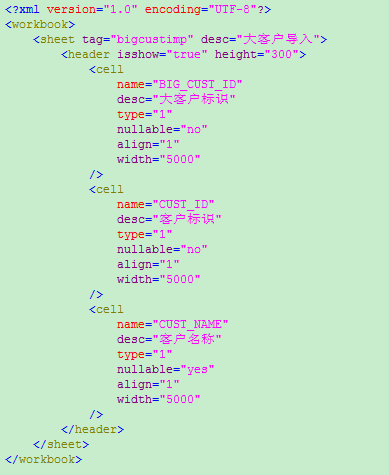
* 前台用户输入要导出生成的文件名并选择文件类型，并执行导出。
* 前台进程发起导出异步任务的调用，同时将数据解析配置xml文件路径和其它业务参数信息作为异步任务执行参数发送给导出执行进程。
* 导出执行进程响应异步任务执行，运行导出的业务逻辑，从数据源获得要导出的数据集。
* 导出任务结合数据集和数据解析配置xml文件，生成导出后的数据文件，并上传到文件服务器，生成文件记录。
* 导出任务执行完成后生成数据文件下链接，并更新导出日志的状态。
* 前台检测导出任务执行完成，将数据文件下载链接提供给用户下载。

### 相关配置

导入导出功能涉及的文件上传下载配置，异步任务配置参考前面相关功能说明部分的配置说明

数据解析配置xml文件：

数据解析配置xml文件是解析和生成Excel数据文件的关键配置，它定义了数据文件中的列和数据集中的列的字段对应关系，以及数据类型，数据校验规则。



sheet对应Excel文件的一个sheet标签，sheet的tag用来区分不同的标签数据，sheet的desc用于生成sheet的名称。

header 对应Excel中的表头，header下的每一个cell表示一列

cell的属性配置如下：

name表示字段名，和数据集中的字段名对应

desc 表示列名，生成Excel文件时以此作为表头的列名

type 为列数据类型，1 – String, 2 – Numeric, 3 – DateTime

nullable 为导入时的校验属性，指定该列数据是否允许为空

align 为单元格横向排列方式, 1 – left，2 – center， 3 – right

format 表格列数据格式（如果是数字，格式为 0.00,#.## 类似数据格式形式，如果为时间，格式为 yyyy-MM-dd 类似形式）

equsize 输入值的定长位数限制，如 equsize="10"

minsize 输入值的最小位数限制，如 minsize="10"

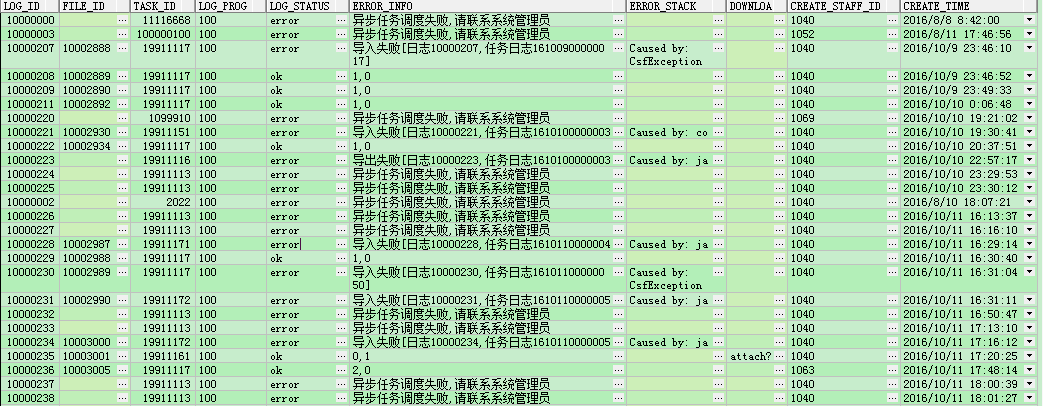
maxsize 输入值的最大位数限制，如 maxsize="100"

数据解析配置文件位于应用工程的 config/export和config/import目录下，应用发布后位于 WEB-INF/classes/export 和 WEB-INF/classes/import目录下。

### 日志记录

导入导出任务执行会将日志记录在 base.wd\_m\_impexp\_log表中：

base.wd\_m\_impexp\_log表



其中：

LOG\_ID是日志标识

FILE\_ID 是导入的数据文件或导出生成的数据文件标识

TASK\_ID 是对应的导入导出异步任务标识

LOG\_PROG 是导入导出执行进度百分比值

LOG\_STATUS 是导入导出任务执行状态，ok为成功， error为异常

ERROR\_INFO 是导入导出的异常信息，若为导入任务，且执行成功，则会记录成功的数据条数和失败的数据条数，用逗号分隔。

ERROR\_STACK 为异常堆栈记录

CREA\_STAFF\_ID 为任务调度发起操作员工号

CREA\_TIME 为任务调度发起时间

### 组件使用

导入组件：



其中：

taskId 为执行导入的异步任务标识

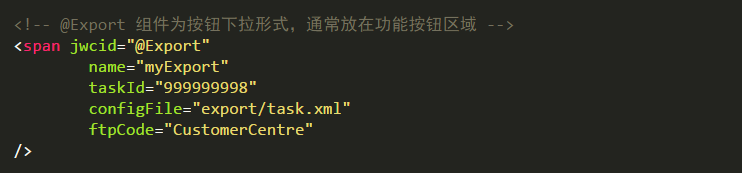
cond 为业务参数区域id

configFile 为数据解析配置xml文件，使用相对路径配置

templaeFile 为目标下载路径

ftpCode 为文件上传下载FTP配置

导出组件



其中：

taskId 为执行导出的异步任务标识

configFile 为数据解析配置xml文件，使用相对路径配置

ftpCode 为文件上传下载FTP配置

更多组件详细用法，请参考在线开发文档

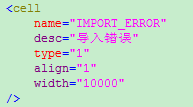
### 常见问题

日期时间格式问题：

导入导出对于日期类型的数据，对于年月日部分只支持yyyy-MM-dd格式日期的读写，导入的数据文件中日期类型单元格数据和数据解析配置xml文件中的日期时间列配置要注意单元格格式和日期format格式配置。

数据校验错误信息：

对于导入过程中数据行的校验错误信息输出，需要在数据解析配置xml文件中定义固定一列IMPORT\_ERROR来生成到导入失败的数据文件中

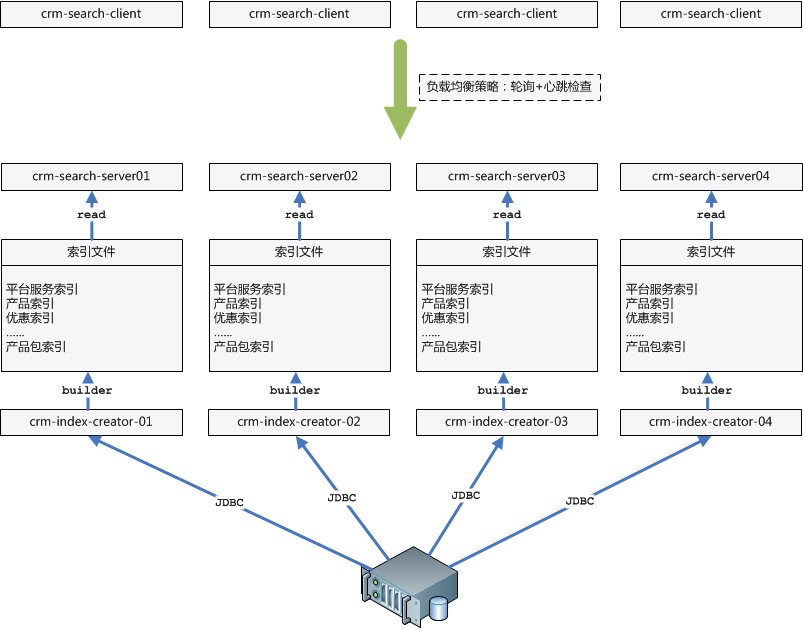


# 分布式搜索

## 为什么采用搜索引擎

传统的关系数据库索引不是为模糊搜索专门设计的，所以在处理模糊匹配时，数据库索引是不起作用的。对于含有模糊查询的数据库服务来说，like对性能的危害极大，只是这种性能损耗在数据量很少，数据库并发数少的情况下，不易暴露出来，所以也经常被开发人员和系统维护人员所忽视。实际业务开发过程中，为了系统使用的方便性，通常需要提供模糊匹配查询功能，希望既能以很快的速度获得想要的搜索结果，又不会导致系统性能受损的目的。为此WADE4.0借助开源lucene搜索引擎技术，提供一套适用于BOSS/CRM业务系统的搜索框架，经测试，此框架在预热完成以后，平均搜索响应时间为10毫秒级，相比数据库的Like多则一两秒，少则几百毫秒要好很多。

## 搜索引擎部署图示



整个搜索引擎采用分层结构部署。分为三个层次：搜索客户端、搜索服务端、索引创建者。

* 搜索客户端

接受搜索关键词，和服务端通信，返回搜索结果。客户端本身不做搜索逻辑处理，它通过RPC远程调用服务端，由服务器端返回对应的搜索结果。

* 搜索服务端

接受客户端传过来的关键字，读取索引文件，返回搜索结果。

* 索引创建者

功能：创建和维护索引文件。

## 配置说明

* 搜索配置主表：UOP\_CEN1.WD\_SEARCH\_CFG

SEARCH\_CODE：搜索编码

SEARCH\_NAME：搜索名称

DATA\_SOURCE：数据源名称，到哪个库取数据，与database.xml对应。

QUERY\_SQL：查询SQL

INDEX\_DATA\_PATH：索引数据目录

INDEX\_TEMP\_PATH：索引临时目录

INDEX\_BACKUP\_PATH：索引备份目录

LIMIT\_HITS：搜索结果数量限制

MAX\_BACKUP：索引最大备份数

STATE：有效性标志位

* 搜索配置子表：UOP\_CEN1.WD\_SEARCH\_FIELDS

SEARCH\_CODE：搜索编码，与主表对应

FIELD\_NAME：索引字段名

NEED\_PINYIN\_INDEX：是否需要支持拼音首字母

STATE：有效性标志位

* 搜索客户端配置文件：search-client.properties

# 搜索服务器地址，可以配置多个地址，用英文逗号分隔

search.server.address=10.154.63.121:11000

## 开发示例

假设我们需要对优惠表的DISCNT\_NAME字段，进行中文+拼音首字母搜索。

* 定义唯一的搜索编码，WD\_SEARCH\_CFG.SEARCH\_CODE
* 定义数据源名称，WD\_SEARCH\_CFG.DATA\_SOURCE
* 定义查询SQL，WD\_SEARCH\_CFG.QUERY\_SQL

简言之，想要搜索出什么结果，就定义什么查询语句，可以关联多表。

SELECT指定的字段尽可能少，原则是够用就好，以减小索引空间成本，提高搜索效率。

SELECT DISCNT\_CODE,

DISCNT\_NAME,

DISCNT\_EXPLAIN,

C\_DISCNT\_CODE,

B\_DISCNT\_CODE,

A\_DISCNT\_CODE,

OBJ\_TYPE\_CODE,

DEFINE\_MONTHS,

MONTHS,

TAG\_SET,

EPARCHY\_CODE

FROM TD\_B\_DISCNT

WHERE SYSDATE BETWEEN START\_DATE AND END\_DATE

* 指定索引目录（数据目录、临时目录、备份目录）

WD\_SEARCH\_CFG.INDEX\_DATA\_PATH

E:/search-index/TD\_B\_DISCNT/data

WD\_SEARCH\_CFG.INDEX\_TEMP\_PATH

E:/search-index/TD\_B\_DISCNT/temp

WD\_SEARCH\_CFG.INDEX\_BACKUP\_PATH

E:/search-index/TD\_B\_DISCNT/backup

* 指定查询结果集大小限制，WD\_SEARCH\_CFG.LIMIT\_HITS

一般1000比较合适，分页参数受这个限制，不建议定义得过大。

* 指定索引备份数量，WD\_SEARCH\_CFG.MAX\_BACKUP
* 指定索引条目的有效性，WD\_SEARCH\_CFG.STATE （‘U’：有效）
* 指定你想对哪些字段进行索引

WD\_SEARCH\_FIELDS.FIELD\_NAME

* 指定你想对哪些字段进行索引

WD\_SEARCH\_FIELDS.NEED\_PINYIN\_INDEX（‘Y’:支持拼音首字母搜索）

依赖jar包清单：(已commit至最新的wadelib目录)

wade-rpc.jar

wade-search.jar

pinyin4j-2.5.0.jar

lucene-core-3.6.2.jar

// 导入搜索客户端

**import** com.ailk.search.client.SearchClient;

/\*\*

\* 参数一：搜索编码，与 UOP\_CEN1.WD\_SEARCH\_CFG.SEARCH\_CODE对应。

\* 参数二：搜索关键字，注：最少要输入两个关键字

\* 参数三：结果集起始位置(用于对搜索结果分页处理)

\* 参数四：结果集终止位置(用于对搜索结果分页处理)

\*/

SearchResponse resp = SearchClient.search(“TD\_B\_DISCNT”, “商务宝”，0，10);

int numTotalHits = resp.getNumTotalHits(); // 匹配总数

IDataset datas = resp.getDatas(); // 当前返回结果，分页后的结果

# 客户端消息推送

## 功能说明



* 客户端通过WebSocket或HTTP长连接连接到消息推送服务器
* 每个客户端有唯一的会话标识，并根据其它业务属性，比如工号，部门，地州，进行分组。
* 应用进程通过http接口调用发对消息推送
* 消息推送服务将消息根据发送目标的分组信息将消息下发到客户端。
* 客户端接收到消息后执行回调处理消息内容

## 服务部署

客户端消息推送应用目录结构：

|-app

|---bin //启停脚本目录

|------monitor-comet-server.sh //HTTP长连接消息推送服务进程检查脚本

|------monitor-websocket-server.sh //Websocket消息推送服务进程检查脚本

|------start-comet-server.sh //HTTP长连接消息推送服务进程启动脚本

|------stop-comet-server.sh //HTTP长连接消息推送服务进程停止脚本

|------start-websocket-server.sh //Websocket消息推送服务进程启动脚本

|------stop-websocket-server.sh //Websocket消息随同服务进程停止脚本

|---log //日志目录

|-etc //配置文件目录

|---acl.conf //推送接口访问控制

|---comet-server.properties //HTTP长连接消息推送服务配置

|---websocket-server.properties //Websocket消息推送服务配置

|---log4j.properties //日志配置

|-lib //相关jar包目录

|-sbin //应用程序和公用脚本目录

|---cronolog //日志切割程序

|---monitor\_process.sh //应用进程检测脚本

|---setEnv.sh //应用环境变量设置脚本

## 相关配置

服务端配置：

acl.conf配置



配置允许调用HTTP推送接口的IP段

comet-server.properties配置：

该配置为HTTP长连接消息推送服务配置



其中：

comet.server.worker.clazz为Worker类实例，不许要修改

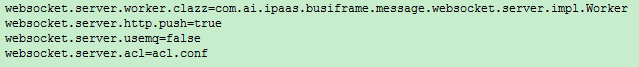
comet.server.http.push 为是否打开HTTP推送接口，需要配置为true

comet.server.usemq 为是否使用消息队列服务接收推送消息，配置为false

comet.server.acl 为访问控制配置文件名，不需要修改

websocket-server.properties配置：

该配置为Websocket消息推送服务配置



其中：

websocket.server.worker.clazz 为Worker类实例，不需要修改

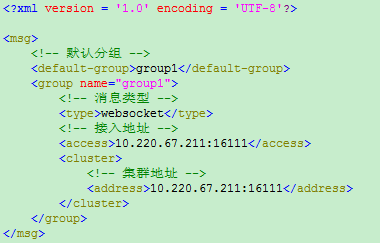
websocket.server.http.push 为是否打开HTTP推送接口，需要配置为true

websocket.server.usemq 为是否使用消息队列服务接收推送消息，配置为false

websocket.server.acl 为访问控制配置文件名，不需要修改

应用配置：

应用配置为msg.xml配置文件，配置内容如下：



其中：

default-group 为默认分组配置，通常只有一组

type 为客户端消息推送服务类型，可选：websocket, comet(HTTP长连接)

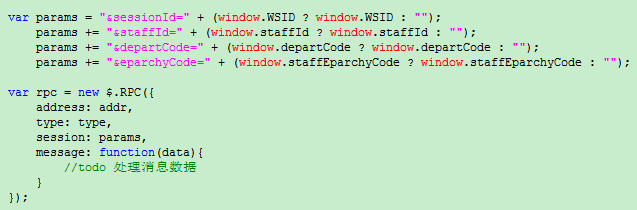
access 为客户端消息推送服务客户端接入地址，如果有前置负载均衡，则此处需要配置为负载均衡的地址

cluster 为客户端消息推送服务集群地址，每个服务实例地址配置一条

## 连接和推送

客户端连接API(JS):

示例：



其中，addr和type分别为通过 服务端JAVA API的

MessageFactory.getDefaultAccess();

和

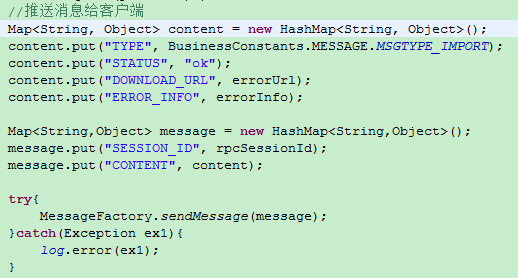
MessageFactory.getDefautType();

获取msg.xml中的配置

sessionId, staffId, departCode,eparchyCode 为固定参数，分别为会话标识，工号，部门编码，地州编码。

message 方法为消息接收的回调函数，形参 data 为接收到的消息内容，数据为JSON对象

消息推送API（JAVA）：



content为消息内容, message为消息体，SESSION\_ID为接收消息的客户端连接会话标识

支持的发送维度：

SESSION\_ID: 发送给特定会话标识的客户端连接

STAFF\_ID: 发送给指定工号的所有客户端连接

DEPART\_CODE： 发送给指定部门标识的客户端连接

EPARCHY\_CODE: 发送给指定地州编码的客户端连接