**IUAP平台使用指南**

**缓存组件**

用友网络科技股份有限公司

二○一八年五月

**此文档为保密文档，未经用友软件股份有限公司书面同意，不得向任何单位或个人提供、转让本文档中的任何内容，用友软件股份有限公司将保留对泄漏文档内容的起诉权利。**

**IUAP平台使用指南**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [**√**]正在修改  [] 正式发布 | 文件标识： |  |
| 当前版本： |  |
| 作者： |  |
| 完成日期： |  |

版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 作者 | 参与者 | 起止日期 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

审核记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 姓名 | 职位 | 审阅日期 | 备注 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[第一章 概述 1](#_Toc515617010)

[1.1 业务场景 1](#_Toc515617011)

[1.2 开发方案 1](#_Toc515617012)

[第二章 示例配置 3](#_Toc515617013)

[2.1 开发环境配置 3](#_Toc515617014)

[2.2 常用缓存接口服务 4](#_Toc515617015)

[第三章 代码开发示例 6](#_Toc515617016)

[3.1 代码见com.yonyou.iuap.example.cache.support.CodeCache 6](#_Toc515617017)

# 概述

## 业务场景

为了提高业务应用的性能，避免对数据库资源的频繁访问，应用中常常引入缓存技术，将需要访问的资源或初次调用后的结果缓存起来，针对后续的相同访问直接返回缓存结果，以支持应用的高并发。

## 开发方案

iuap缓存组件主要采用Redis中间件提供缓存解决方案，并针对Redis提供统一的连接以及基本Java API封装，更利于方便业务开发简便快速的实现对业务数据的缓存操作，提高系统的运行效率。

此外，iuap缓存组件还实现了对连接池的管理，并支持Redis以主从方式或者分片方式进行部署。分布式的架构保证了缓存服务的高可用性，在主从模式下支持主节点宕机后的自动切换，业务服务无感知。

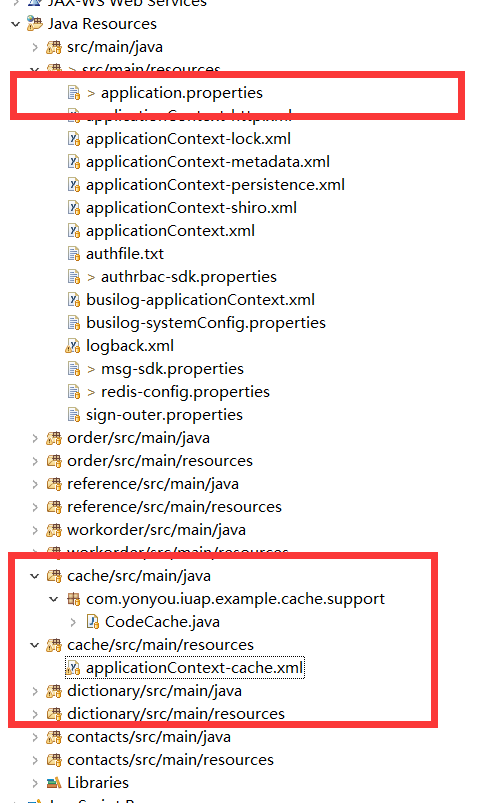
iuap缓存服务组件提供了如下功能：

* 支持对缓存的读写操作以及过期时间设置；
* 支持复杂类型数据的操作，如HashMap，Set，List；
* 支持Redis主从和分片集群的不同配置；
* 支持Redis的密码授权访问；
* 支持Redis连接池配置管理；

示例应用将从以上几个方面来对使用iuap缓存服务进行介绍。

示例应用为cache，主要代码工作为“配置”和“Java调用”两部分，代码结构如下：

* 代码结构：



# 示例配置

## 开发环境配置

1. 下载、安装Redis中间件

从redis官网下载redis中间件，并参考官方文档进行安装；

redis官网地址：<https://redis.io>

iuap平台一键部署中已包含redis服务，本示例将直接使用该redis进行缓存应用示例开发。

1. 在工程中添加iuap-cache组件的Maven依赖，具体如下：

<dependency>

<groupId>com.yonyou.iuap</groupId>

<artifactId>iuap-cache</artifactId>

<version>${iuap.modules.version}</version>

</dependency>

其中，${iuap.modules.version} 为平台在maven私服上发布的组件的version

1. 配置Redis中间件地址信息，主要提供三种方式：

* 单机方式

redis.url=direct://10.1.1.2:6379?poolSize=50&poolName=mypool&masterName=mymaster&password=yonyou123&database=0

* 主从集群方式

redis.url=sentinel://10.1.1.2:16379, 10.1.1.3:16379, 10.1.1.4:16379? PoolName=mypool&masterName=mymaster&poolSize=100

* 分片方式

redis.shardedurl=direct:// 10.1.1.2:6379, 10.1.1.3:6379, 10.1.1.3:6379?poolName=mypool&masterName=mymaster&poolSize=100

注：10.1.1.\*为Redis中间件安装服务器IP

1. 配置applicationContext-cache.xml，使用Spring注册加载iuap-cache服务组件

<!-- redis连接池配置 -->

<bean id=*"redisPool"* class=*"com.yonyou.iuap.cache.redis.RedisPoolFactory"*

scope=*"prototype"* factory-method=*"createJedisPool"*>

<constructor-arg value=*"${redis.url}"* />

</bean>

<bean id=*"jedisTemplate"* class=*"org.springside.modules.nosql.redis.JedisTemplate"*>

<constructor-arg ref=*"redisPool"*></constructor-arg>

</bean>

<bean id=*"cacheManager"* class=*"com.yonyou.iuap.cache.CacheManager"*>

<property name=*"jedisTemplate"* ref=*"jedisTemplate"*/>

</bean>

## 常用缓存接口服务

CacheManager提供了缓存服务常用接口

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数 | 返回值 | 说明 |
| set(final String key, final T value) | String key（缓存key）, T value（缓存对象，允许序列化） | void | 放置键值对对应的缓存 |
| setex(final String key, final T value，final int timeout) | String key（缓存key）, T value（缓存对象，允许序列化），int timeout（缓存的有效期） | void | 放置键值对对应的缓存并设置缓存有效期，单位为秒 |
| expire(final String key, final int timeout) | final String key（缓存key）, final int timeout（缓存的有效期） | void | 修改缓存信息的有效期 |
| exists(final String key) | final String key（缓存key） | Boolean（是否存在的标志） | 判断键值为key的缓存是否存在 |
| get(final String key) | final String key（缓存key） | T extends Serializable（声明的返回类型对象） | 获取对应键值的缓存对象 |
| hget(final String key, final String fieldName) | final String key（缓存Map的key），final String fieldName（Map下某个属性的key） | T extends Serializable（声明的返回类型对象） | 获取HashMap中对应的子key的缓存值 |
| hmget(final String key, final String... fieldName) | final String key（缓存Map的key）, final String... fieldName（Map下若干属性的key） | List | 获取HashMap中对应的多个子key的缓存值集合 |
| hexists(final String key, final String field) | final String key（缓存Map的key），final String field（Map下某个属性的key） | Boolean（是否存在的标志） | 判断HashMap中对应的key是否存在 |
| hset(final String key, final String fieldName, final T value) | final String key（缓存Map的key），final String fieldName（Map下某个属性的key），final T value（要放置的缓存对象） | void | 放置HashMap中的属性和值 |
| removeCache(String key) | String key（缓存的key） | void | 删除key对应的缓存信息 |
| hdel(String key, String field) | String key（缓存Map的key），String field（Map下某个属性的key） | void | 删除Map下键值为filed的属性 |

# 代码开发示例

## 代码见com.yonyou.iuap.example.cache.support.CodeCache

代码内容主要是：在系统启动时加载档案字典数据，并以Map存入缓存，并提供查询获取方法。

