# distributed-lock说明文档

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | \*变化状态 | 修订说明 | 日期 | 变更人 |
| **V1.0** | **A** | **文档设计** | **20180522** | **王海洋** |

## 系统说明

fjf-distributed-lock 是一个分布式锁的基础服务，采用redis实现，目前支持以下锁方式：

* **可重入锁（Reentrant Lock），**同时还支持自动过期解锁。
* **联锁（MultiLock）**对象可以将多个RLock对象关联为一个联锁，每个RLock对象实例可以来自于不同的Redisson实例,注意：所有的锁都上锁成功才算成功
* 异步锁，异步执行锁(提供验证和可重入锁[Reentrant Lock]设计,同时还支持自动过期解锁)

## Maven 依赖

<dependency>

<groupId>com.foriseland.fjf</groupId>

<artifactId>fjf-distributed-lock</artifactId>

<version>1.0.0-SNAPSHOT</version>

</dependency>

## 依赖说明

DistributedLock  分布式锁基础服务

**必要依赖：**

* fjf-cache 缓存基础服务类
* fjf-common 基础工具类

特别备注：没有这两个依赖无法使用锁服务。

## 接口服务说明

备注：进入业务方法时候需要先判断是否加锁public boolean pressingReturnLock(String key) throws CacheException

返回Boolean 如果为true代表锁被持有，如果返回false 则表示未加锁

### 可重入锁（lock）

#### 说明

对象实现了java.util.concurrent.locks.Lock接口，同时还支持自动过期解锁。

#### 方法实例

public boolean lock(String key,Integer leaseTime)

throws com.foriseland.fjf.cache.exception.CacheException

**接口描述：对外服务加锁(提供验证和可重入锁[Reentrant Lock]设计,同时还支持自动过期解锁)**

**参数:**

**key**: redis key 满足redis的key设计规范[object-model-key]，否则会抛出distributed to Lock is error,this key not effective,key value-> [" + key+ "],redis key be similar to [object-model-key]

**leaseTime**:锁失效时间，释放时间，以秒作为计量单位，如果传入null 则默认为10秒。

**返回:**

Boolean {true:success | false:error or CacheException}

**抛出:**

com.foriseland.fjf.cache.exception.CacheException

### 解锁

#### 说明

unlock 提供对锁资源释放功能，无返回值.

#### 方法实例

public void unLock(String key)throws com.foriseland.fjf.cache.exception.CacheException

**参数：**key:generate - lock key redis key 满足redis的key设计规范[object-model-key]，否则会抛出distributed to Lock is error,this key not effective,key value-> [" + key+ "],redis key be similar to [object-model-key]

**抛出：**com.foriseland.fjf.cache.exception.CacheException

### 异步锁（asyncReentrantLock）

#### 说明

#### 方法实例

public java.util.concurrent.Future<java.lang.Boolean> asyncReentrantLock(String key,Integer leaseTime) throws com.foriseland.fjf.cache.exception.CacheException

**参数：**

key: Lock key满足redis的key设计规范[object-model-key]，否则会抛出distributed to Lock is error,this key not effective,key value-> [" + key+ "],redis key be similar to [object-model-key]

leaseTime:加锁时间,默认支持过期解锁功能,10秒钟以后自动解锁,无需调用unlock方法手动解锁

**返回:：**

Boolean {true:success | false:error or CacheException}

**抛出:：**

com.foriseland.fjf.cache.exception.CacheException

锁使用完毕后需要调用**unlock** 去释放锁，释放锁需要在方法中手动释放，如果未手动释放，系统会在失效期后自动释放。

### 联锁（multiLock）

#### 说明

对象可以将多个RLock对象关联为一个联锁，每个RLock对象实例可以来自于不同的Redisson实例。

#### 方法实例

public boolean multiLock(java.lang.String... lockKeys)

throws com.foriseland.fjf.cache.exception.CacheException

MultiLock对象可以将多个RLock对象关联为一个联锁，每个RLock对象实例可以来自于不同的Redisson实例。

**参数:**

lockKeys - key:generate - lock key redis key 满足redis的key设计规范[object-model-key]，否则会抛出distributed to Lock is error,this key not effective,key value-> [" + key+ "],redis key be similar to [object-model-key]

**返回:** Boolean {true:success | false:error or CacheException}

**抛出:**

com.foriseland.fjf.cache.exception.CacheException

## 备注

所有锁系统会默认增加锁前缀，前缀信息为：DISTRIBUTED&LOCK@XXX(KEY)为统一格式。

加锁日志统一格式为：[#Foriseland-Distributed-Lock#] -> 统一通过logback的方式打到日志文件中。