百知教育
Baizhi Education

redis 实战

讲师：蒋中洲

Redis简介

- **NoSQL技术简介**
  1. Schemaless
  2. In-Memory
  3. 弱化事务
  4. 适用于Cluster环境
  5. 没有复杂的连接查询操作
  6. 脚本语言支持

Redis简介

Redis是什么？

Redis 是一个开源、支持网络、基于内存、键值对型的NOSQL数据库。

Redis的特点

1. Redis是一个高性能的Key/Value数据库
2. 基于内存
3. 数据类型丰富
4. 持久化
5. 单线程
6. 订阅/发布模型

Redis简介

Redis与Memcached对比

1. Redis不仅仅支持简单的k/v类型的数据，同时还提供list，set，hash等数据结构存储
2. Redis支持数据的主从复制，即master-slave模式的数据备份；
3. Redis支持数据的持久化，可以将内存中的数据保存到磁盘中，重启的时候可以再次加载原有数据；
4. Redis单个value的最大限制是1GB， memcached只能保存1MB的数据。

Redis简介

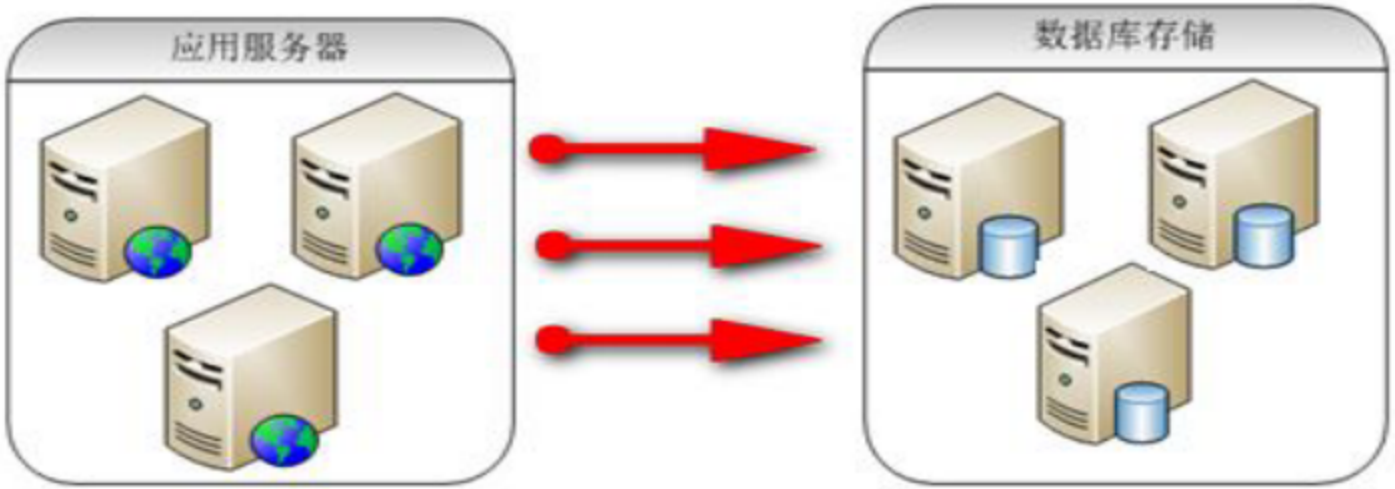
- 谁在使用Redis?

Instagram

flickr™

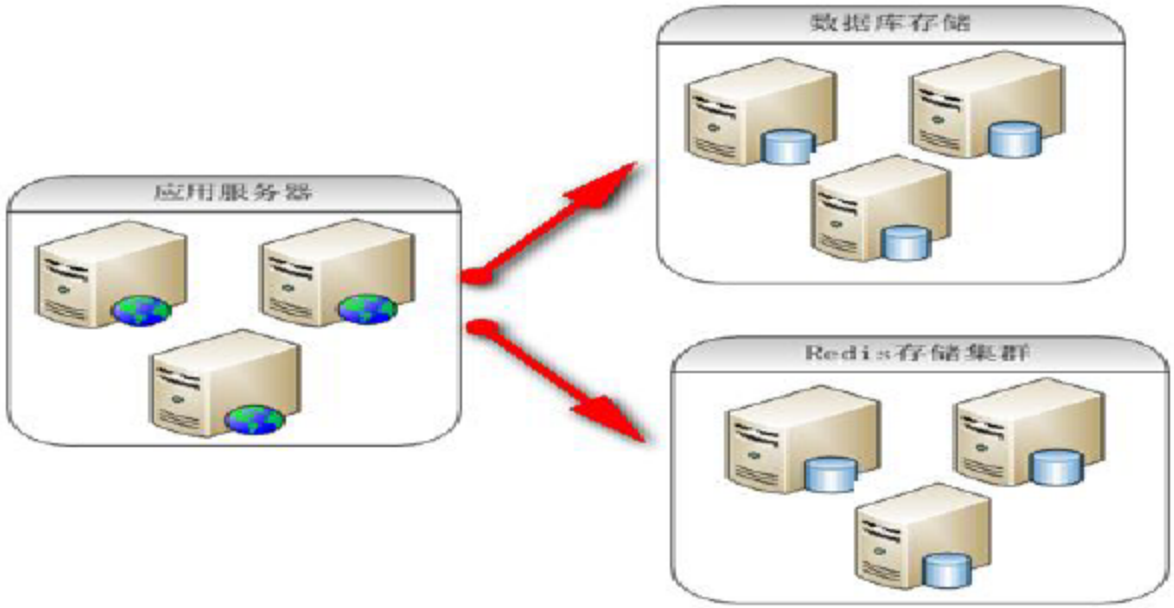
Redis简介

- 传统架构



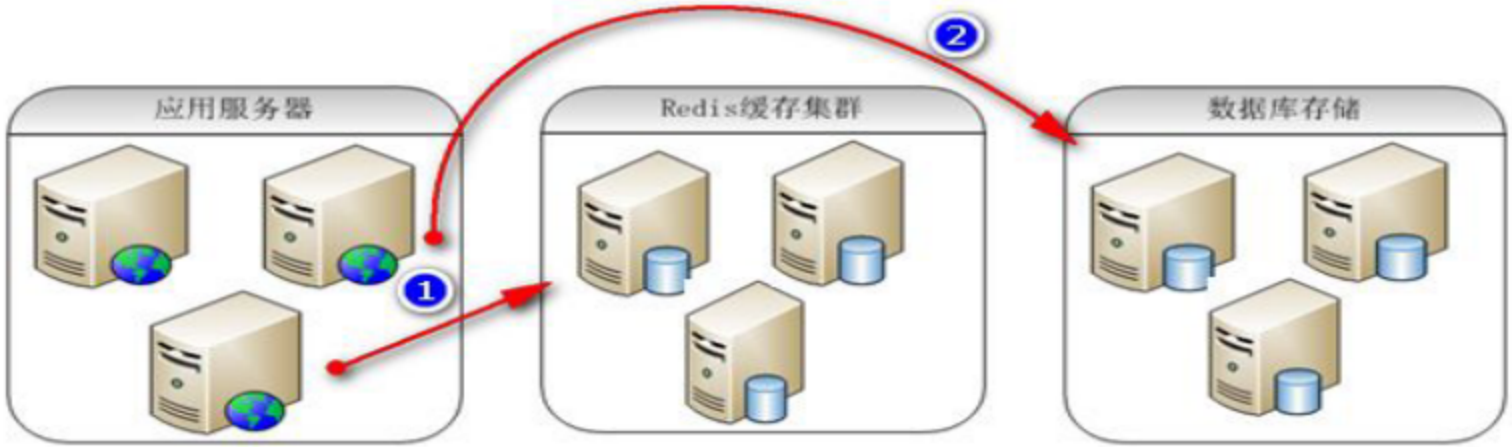
Redis简介

- Redis存储非关系型数据



Redis简介

- Redis充当分布式缓存架构



Redis的安装

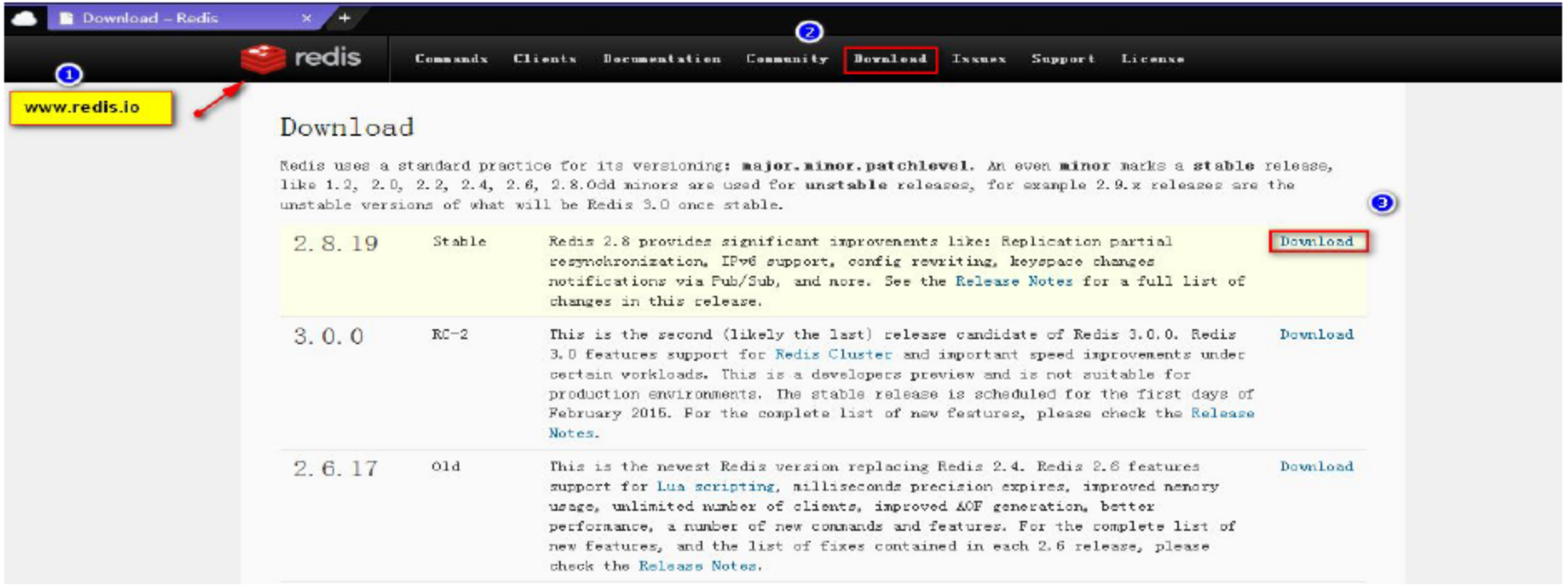
- **环境准备**
  1. VMWare/Virtualbox
  2. CentOS6.5
  3. gcc
  4. SECURECRT
  5. WinSCP

- **访问Redis页**

  主页地址：www.redis.io

百知教育
Baizhi Education

Redis的安装



Redis的安装

- **启动服务**
  1. ./redis-server redis.conf 默认监听端口 6379

- **启动客户端shell**
  2. ./redis-cli

数据类型String

命令	说明
set	设置一个key/value
get	根据key获得对应的value
mset	一次设置多个key value
mget	一次获得多个key的value
getset	获得原始key的值，同时设置新值
strlen	获得对应key存储value的长度
append	为对应key的value追加内容
getrange	截取value的内容

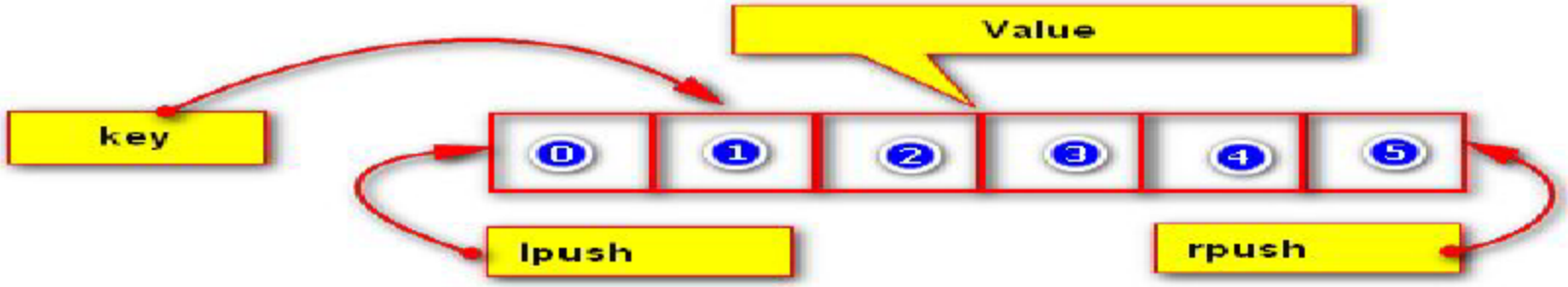
数据类型String

命令	说明
setex	设置一个key存活的有效期（秒）
psetex	设置一个key存活的有效期（豪秒）
setnx	只有当这个key不存在时等效set操作
msetnx	可以同时设置多个key
decr	进行数值类型的-1操作
decrby	根据提供的数据进行减法操作
lncr	进行数值类型的+1操作
incrby	根据提供的数据进行加法操作

数据类型String

命令	说明
Incrbyfloat	根据提供的数据加入浮点数

数据类型List



命令	说明
lpush	将某个值加入到一个key列表头部
lpushx	同lpush,但是必须要保证这个key存在
rpush	将某个值加入到一个key列表末尾
rpushx	同rpush,但是必须要保证这个key存在

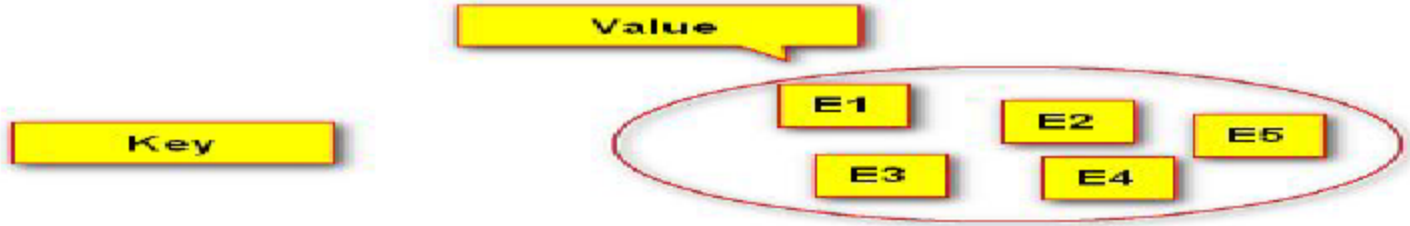
数据类型List

命令	说明
lpop	返回和移除列表的第一个元素
rpop	返回和移除列表的第一个元素
lrange	获取某一个下标区间内的元素
llen	获取列表元素个数
lset	设置某一个位置的元素
lindex	获取某一个位置的元素
lrem	删除重复元素
ltrim	保留列表中特定区间内的元素

数据类型List

命令	说明
linsert	在某一个元素之前，之后插入新元素

数据类型Set

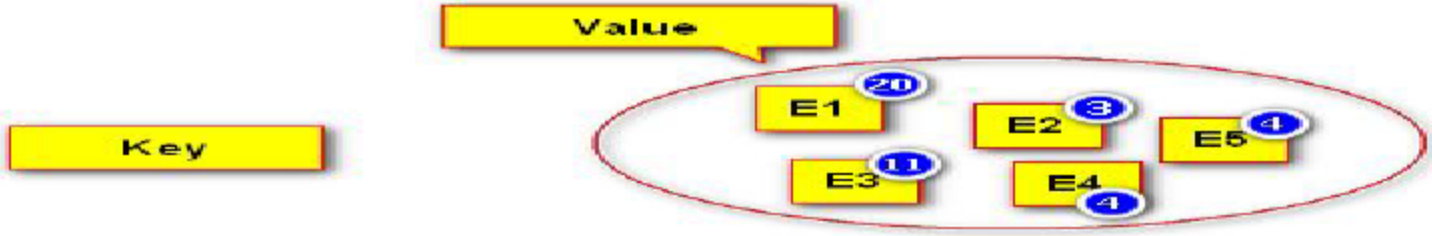


命令	说明
sadd	为集合添加元素
smembers	显示集合中所有元素 无序
scard	返回集合中元素的个数
spop	随机返回一个元素

数据类型Set

命令	说明
smove	从一个集合中向另一个集合移动元素
srem	从集合中删除一个元素
sismember	判断一个集合中是否含有这个元素
srandmember	随机返回元素
sdiff	减去两个集合中共有的元素
sinsert	求并集
sunion	求和集

数据类型ZSet

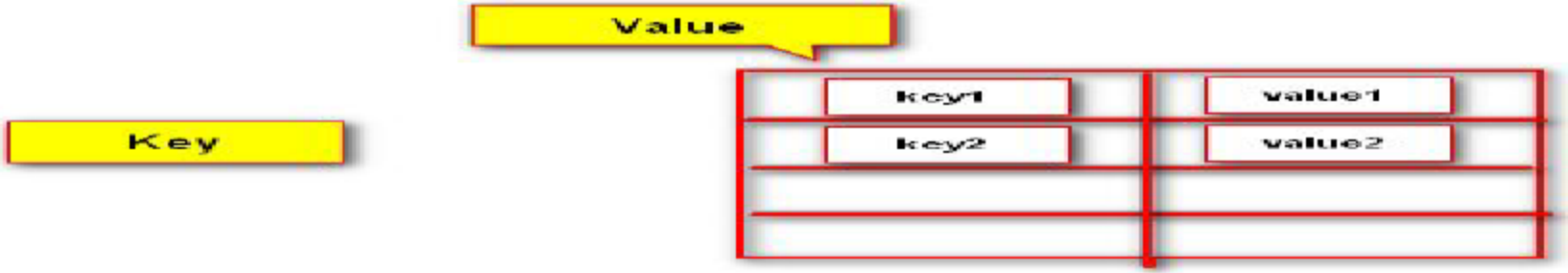


命令	说明
zadd	添加一个有序集合元素
zcard	返回集合的元素
zrange	返回一个范围内的元素
zrangebyscore	按照分数查找一个范围内的元素

数据类型ZSet

命令	说明
zrank	返回排名
zrevrank	倒序排名
zscore	显示某一个元素的分数
zrem	移除某一个元素
zincrby	给某个特定元素加分

数据类型Hash



命令	说明
hset	设置一个key/value对
hget	获得一个key对应的value
hgetall	获得所有的key/value对
hdel	删除某一个key/value对

数据类型Hash

命令	说明
hexists	判断一个key是否存在
hkeys	获得所有的key
hvals	获得所有的value
hmset	设置多个key/value
hmget	获得多个key的value
hsetnx	设置一个不存在的key的值
hincrby	为value进行加法运算
hincrbyfloat	为value加入浮点值

Redis的持久化

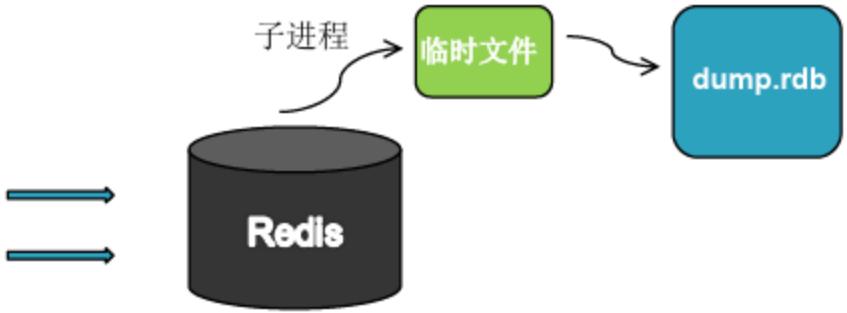
- **Redis持久化的概念**
  Redis是一个支持持久化的内存数据库，也就是说Redis需要经常将内存中的数据同步到磁盘来保证持久化。Redis支持两种持久化方式，一种是Snapshotting(快照)也是默认方式，另一种是Append-Only File(缩写AOF)的方式。

- **Snapshotting(RDB)机制**
  在某个时间点保存一个完整的数据快照。

Redis的持久化

Snapshotting(RDB)机制的运行原理

1. Redis通过fork产生子进程；
2. 父进程继续处理client请求，子进程负责将快照写入临时文件；
3. 子进程写完后，用临时文件替换原来的快照文件，然后子进程退出。



Redis的持久化

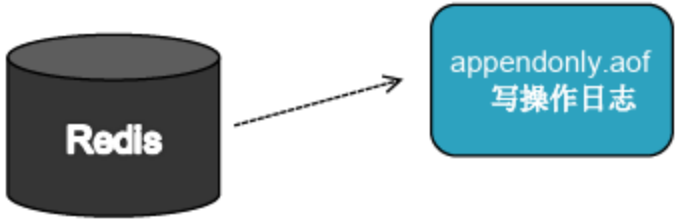
- **Snapshotting(RDB)机制的开发步骤**
  编辑redis.conf文件

```
save  900  1       #900秒超过1个key被修改
save  300  10      #300秒超过10个key被修改
save  60   10000   #60 秒超过一万条被修改
```

```
[1424] 06 Apr 01:21:18.102 * 1 changes in 900 seconds. Saving...
[1424] 06 Apr 01:21:18.104 * Background saving started by pid 1441
[1441] 06 Apr 01:21:18.110 * DB saved on disk
[1441] 06 Apr 01:21:18.111 * RDB: 0 MB of memory used by copy-on-write
```

Redis的持久化

- Append-Only File(AOF)机制
  每一个写操作，将写入log文件，用于数据恢复



- Append-Only File(AOF)机制的运行原理
  1. Redis 通过fork一个子进程
  2. 父进程继续处理client请求，子进程把AOF内容写入缓冲区；
  3. 子进程写完退出，父进程接收退出消息，将缓冲区AOF写入临时文件；
  4. 临时文件重命名成appendonly.aof，原来文件被覆盖，整个过程完成。

Redis的持久化

- Append-Only File(AOF)机制的开发步骤

  编辑redis.conf文件

```
appendonly  yes          #启动AOF机制
appendfsync always       #每次收到写命令就立即强制 写入磁盘，最慢的，但是
                           保证完全的持久化，不推荐使用
appendfsync everysec     #每秒钟强制写入磁盘一次，在性能和持久化方面做了
                           很好的折中，推荐
appendfsync no           #完全依赖os，性能最好,持久化没保证。
```

Redis的持久化

Snapshotting与AOF的对比

持久化技术	优势	缺陷
Snapshotting	1、RDB产生的文件小。 2、RDB恢复快，并且简单，例如你可以快速的将RDB文件传输到其他主机，做数据的恢复。 3、在进行RDB备份的时候，主进程仅仅需要创建一个子进程，所有的I/O操作都由子进程完成	1、不能完全保证数据安全，在两个备份点之间可能会发生数据丢失 2、当数据量很大时，创建子进程可能会是一个非常耗时的操作，甚至可能需要1秒，在这个期间，Redis无法向客户端提供服务。
aop	1、数据的备份粒度更小，数据安全性更高。 2、AOF只会对日志文件进行追加操作，不会修改已经写好的内容。即使在掉电的情况下，AOF日志仍然是可用的	1、AOF文件通常比相同的数据集的RDB文件更大。 2、AOF写日志可能会很慢，这跟fsync的机制有关。

Redis的内存分配管理

- **Redis的内存管理**
  1、与memcached不同，没有实现自己的内存池
  2、在v2.4.4以前，默认使用标准的内存分配函数(libc)，可以选择使tcmalloc
  3、在v2.4.4以后，jemalloc成为代码的一部分

- **Redis中各种内存管理的对比**
  使用Redis自带的redis-benchmark写入等量数据进行测试

百知教育 Baizhi Education

Redis的内存分配管理

Redis中各种内存管理的对比

小数据量对比

```
used_memory:708391440
used_menory_human:675.57M
used_memory_rss:715169792
used_memory_peak:708814040
used_memory_peak_human:675.98M
mem_fragmentation_ratio:1.01
mem_allocator:tcmalloc-1.7
```

```
used_memory:708381168
used_menory_human:675.56M
used_memory_rss:723587072
used_memory_peak:708803768
used_memory_peak_human:675.97M
mem_fragmentation_ratio:1.02
mem_allocator:jemalloc-2.2.1
```

```
used_memory:869000400
used_menory_human:828.74M
used_memory_rss:1136689152
used_memory_peak:868992208
used_memory_peak_human:828.74M
mem_fragmentation_ratio:1.31
mem_allocator:libc
```

百知教育 Baizhi Education

Redis的内存分配管理

Redis中各种内存管理的对比

1k数据量对比

used_memory:830573680	used_memory:915911024	used_memory:771963304
used_memory_human:792.10M	used_memory_human:873.48M	used_memory_human:736.20M
used_memory_rss:849068032	used_memory_rss:927047680	used_memory_rss:800583680
used_memory_peak:831436048	used_memory_peak:916773392	used_memory_peak:772784056
used_memory_peak_human:792.92M	used_memory_peak_human:874.30M	used_memory_peak_human:736.98M
mem_fragmentation_ratio:1.02	mem_fragmentation_ratio:1.01	mem_fragmentation_ratio:1.04
mem_allocator:tcmalloc-1.7	mem_allocator:jemalloc-2.2.1	mem_allocator:libc

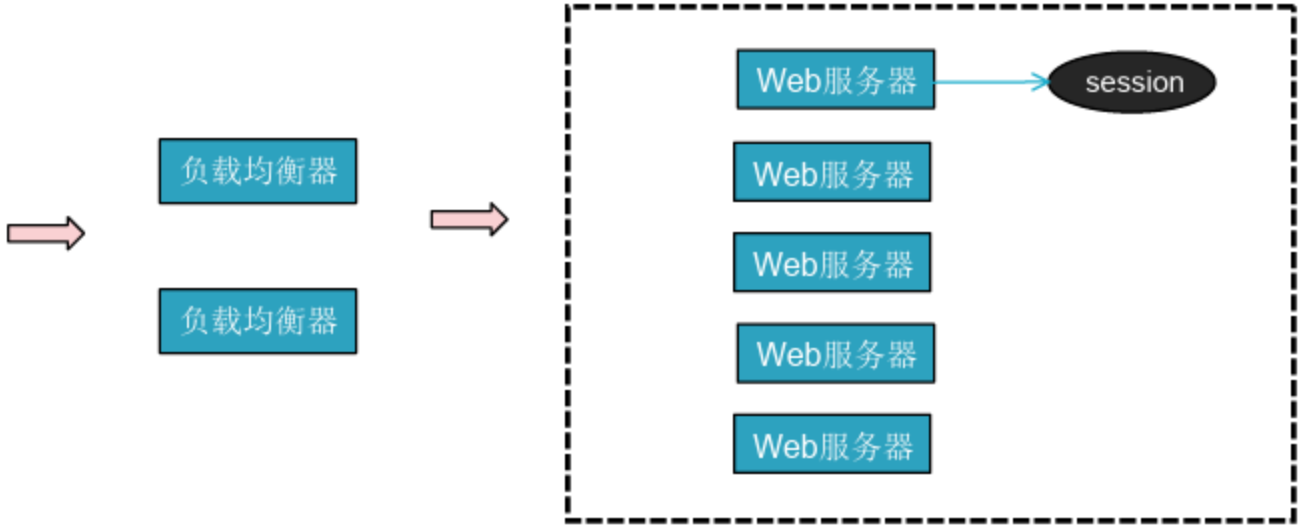
Redis的内存分配管理

- **Redis如何使用内存管理器**
  进行make的过程中使用：

```
make MALLOC = libc
make MALLOC = jemalloc
```

集群环境下的Session管理

- 集群环境下管理HttpSession所遇到的问题

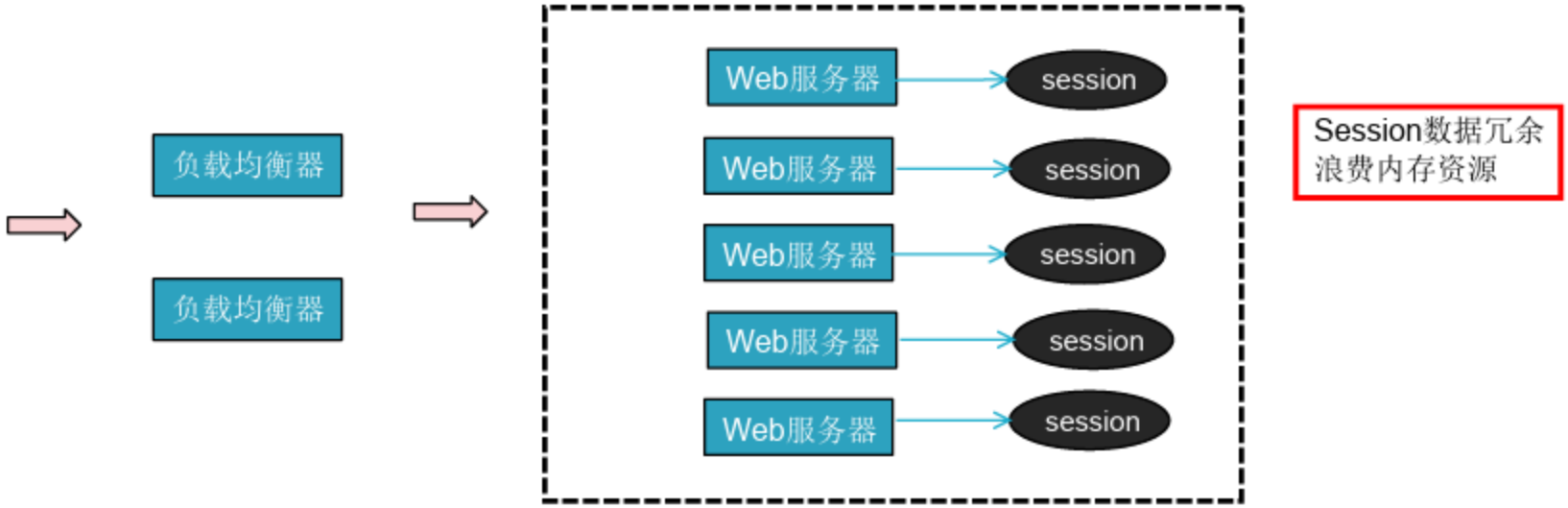


集群环境下的Session管理

- **集群环境下管理Session的4种方式**
  1、Session的Replication
  2、Session的Sticky
  3、Cookie保存状态
  4、通过Redis或者Memcache等分布式缓存集中管理Session

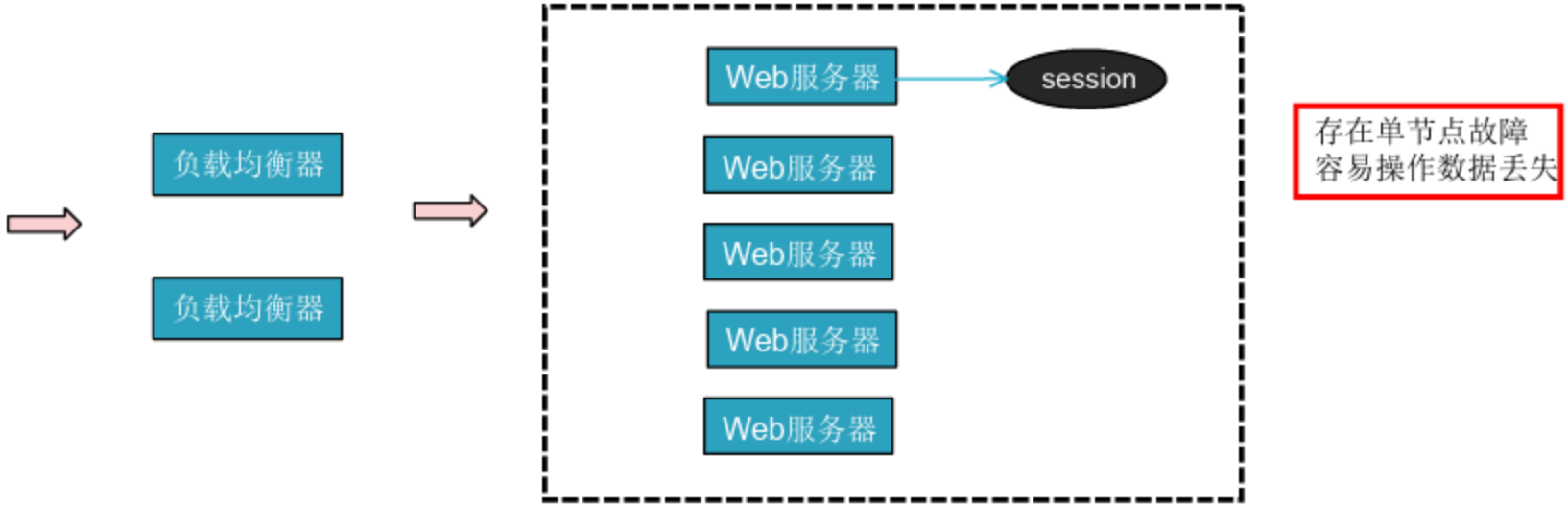
集群环境下的Session管理

- Session的Replication



集群环境下的Session管理

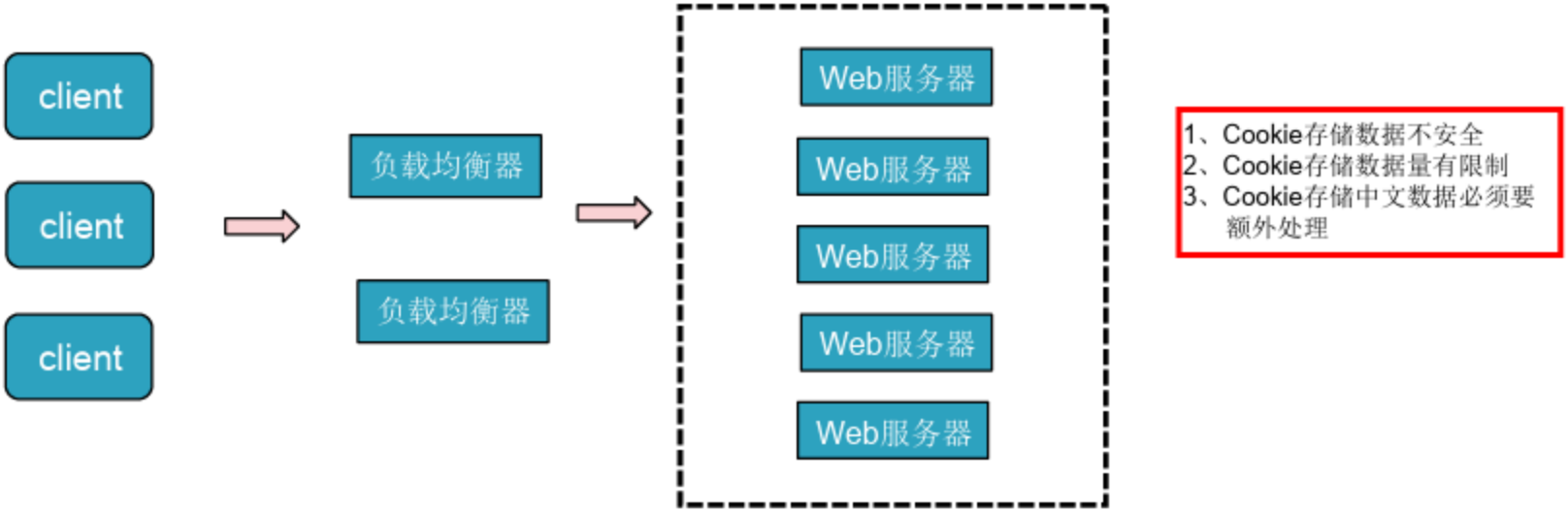
- Session的Sticky



百知教育 Baizhi Education

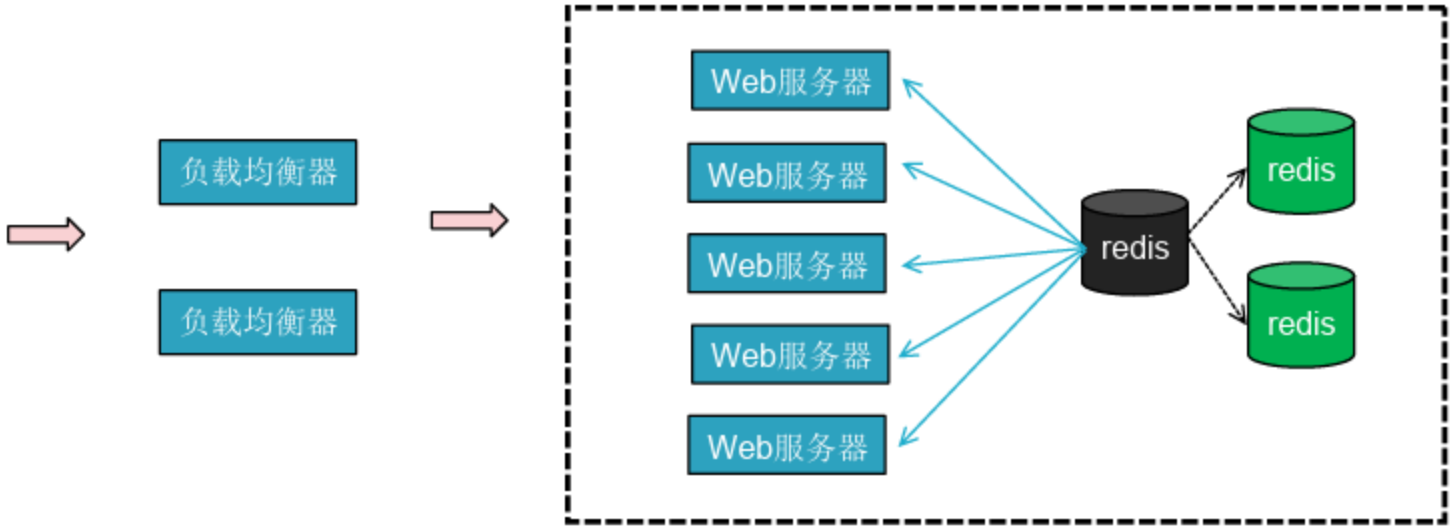
集群环境下的Session管理

- Cookie保存状态



集群环境下的Session管理

- 通过Redis或者Memcached集中管理Session



集群环境下的Session管理

- **通过Redis集中管理Session的开发步骤**

1、在tomcat的lib目录中导入

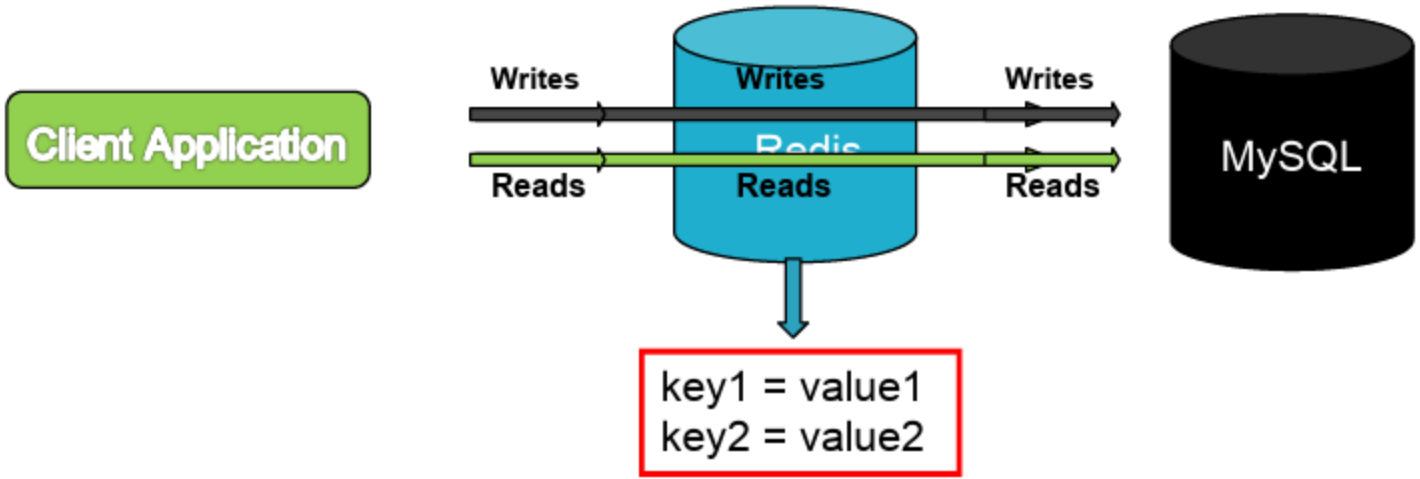
tomcat-redis-session-manager-1.2-tomcat-6.jar
jedis-2.0.0.jar
commons-pool-1.6

2、在tomcat的conf目录中修改context.xml配置文件

```
<Valve className="com.radiadesign.catalina.session.RedisSessionHandlerValve" />
<Manager className="com.radiadesign.catalina.session.RedisSessionManager"
                host="192.168.1.105"
                port="6379"
                maxInactiveInterval="60"/>
```

Redis充当Web缓存

- Redis充当Web缓存的概念



Redis充当Web缓存

- **Redis中的key如何设计？**

1、查询有几种情况？

通过主键查询　id

通过其他条件查询　sql

2、key的设计

```
product:1   {name:"hibernate",price:10}
product:2   {name:"spring",price:10}
product:select * from t_product  1
                                 2
```

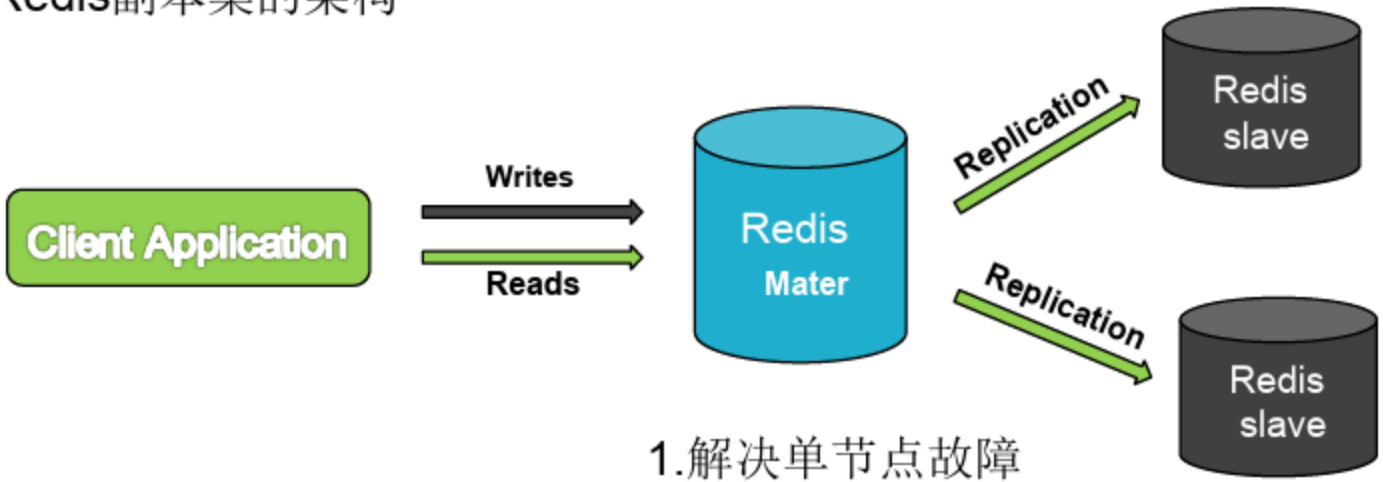
Redis的副本集

- **Redis的副本集**
  1、为什么需要副本集？

防止单节点故障

Redis的副本集

- **Redis的副本集**
  1、Redis副本集的架构



1.解决单节点故障
2.读写分离

Redis的副本集

Redis的副本集

1、Redis副本集的开发步骤

a 准备一台Redis主服务器 一台Redis从服务器

b 在从服务器中配置redis.conf文件

```
slaveof <masterip> <masterport>
```

2、Redis副本集的运行过程

a slave redis进程启动以后，连接到master redis，然后发送sync命令

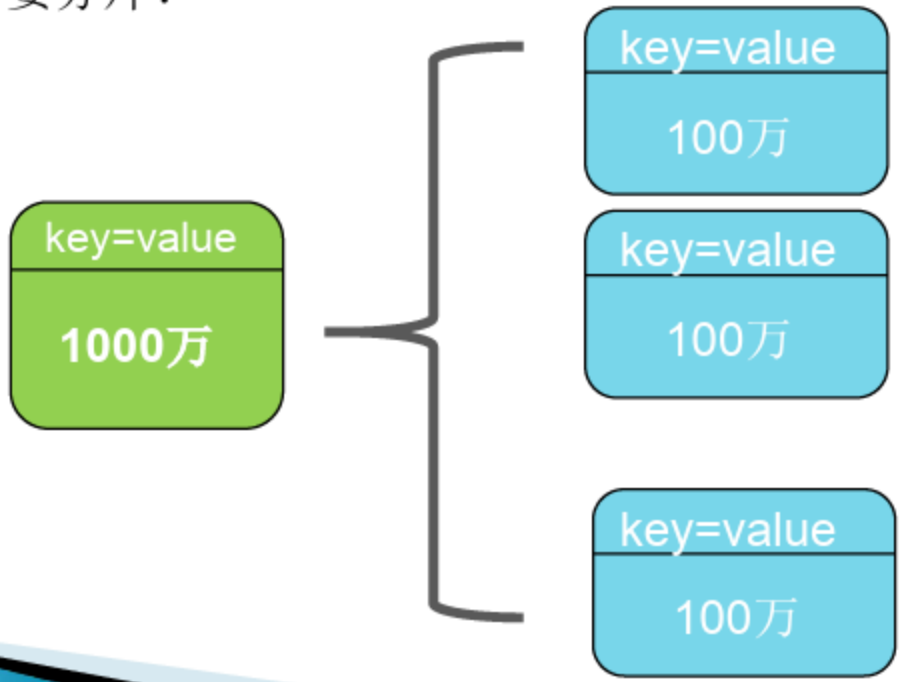
b master redis接到sync命令以后，将所有内存数据写入文件，在这个期间，缓存数据改动的指令。文件完成后，传给slave redis

c slave redis将数据加载进内存

d master redis将后面的涉及到改名数据内容的指令，发送给slave redis

Redis的分片

- **Redis的分片**
  为什么需要分片？



提高查询效率

提高并发

Redis的分片

- **Redis的分片**

  分片的开发步骤

```
Set<HostAndPort> jedisClusterNodes = new HashSet<HostAndPort>();
//Jedis Cluster will attempt to discover cluster nodes automatically
jedisClusterNode.add(new HostAndPort("127.0.0.1", 7379));
JedisCluster jc = new JedisCluster(jedisClusterNode);
jc.set("foo", "bar");
String value = jc.get("foo");
```

Redis充当消息队列

- **消息队列Message Queue(MQ)**
  1、常见的MQ产品

  Active MQ、IBM WebSphere MQ、MetaQ、Kafka

  2、MQ的模型

1、解耦合
2、提高系统的响应时间

订单支付成功的方法(){
  1、修改订单状态
  2、计算会员积分
  3、通知物流进行配送
}

Redis充当消息队列

- **消息队列Message Queue(MQ)**
  - 3、Redis的MQ使用
    - publish
    - subscribe

Redis的JavaDriver

- **Redis的JavaDriver**
  1、通过Maven导入jar

```
<dependency>
    <groupId>redis.clients</groupId>
    <artifactId>jedis</artifactId>
    <version>2.2.1</version>
    <type>jar</type>
    <scope>compile</scope>
</dependency>
```

Redis的JavaDriver

- **Redis的JavaDriver**
  2、使用相关类型进行测试

```
Jedis jedis = new Jedis("localhost");
jedis.set("foo", "bar");
String value = jedis.get("foo");
```

Spring-Data-Redis

- **Spring-Data-Redis框架**
  1、什么是Spring-Data-Redis?
  基于Spring开发的一个简化JedisAPI操作的java框架,其使用方式与Spring一致
  2、Spring-Data-Redis代码

结束

百知教育
Baizhi Education