

# Redis开发实操之春运迁徙页面

演讲人（凡澈）

NoSQL Engineer, Redis Committer, Jedis Reviewer

# Agenda

## 一、开源Redis体验

- 在线体验
- 下载、编译、启动

## 二、云Redis开通到连接

- 创建、白名单与账号、连接

## 三、春运迁徙页面开发

- 使用spring data redis
- 个人信息
- 地理位置
- 新闻通知
- 迁入迁出地排名

# 01 开源Redis使用



# 开源Redis体验

访问 <https://try.redis.io/> 可以在线执行Redis命令，体验Redis。

也可以在Redis官网每个命令文档页面 <https://redis.io/commands/set> 在线执行Redis命令



# 从源码编译启动Redis

Redis GitHub : <https://github.com/redis/redis>

## Linux&MacOS

```
$ git clone https://github.com/redis/redis.git
$ make -j
$ ./src/redis-server
$ ./src/redis-cli // 在另一个终端连接
```

## Windows

<https://github.com/tporadowski/redis>

```
→ redis git:(unstable) ./src/redis-server
60582:C 23 Feb 2021 16:37:36.050 # o000o000o000o Redis is starting o000o000o00
0o
60582:C 23 Feb 2021 16:37:36.050 # Redis version=255.255.255, bits=64, commit=
8e83bcd2, modified=0, pid=60582, just started
60582:C 23 Feb 2021 16:37:36.050 # Warning: no config file specified, using th
e default config. In order to specify a config file use ./src/redis-server /pa
th/to/redis.conf
60582:M 23 Feb 2021 16:37:36.051 * Increased maximum number of open files to 1
0032 (it was originally set to 4864).
60582:M 23 Feb 2021 16:37:36.051 * monotonic clock: POSIX clock_gettime

Redis 255.255.255 (8e83bcd2/0) 64 bit

Running in standalone mode
Port: 6379
PID: 60582

http://redis.io
```

```
→ redis git:(unstable) ./src/redis-cli
127.0.0.1:6379> set key value
OK
127.0.0.1:6379> get key
"value"
127.0.0.1:6379> lpush list a b c d e
(integer) 5
127.0.0.1:6379> lpop list
"e"
127.0.0.1:6379> lpop list
"d"
127.0.0.1:6379> █
```

# 使用客户端程序连接Redis

Redis客户端程序生态繁荣，可参考：

<https://redis.io/clients>，常见的如：

- Java: Jedis, Lettuce, Redisson
- C/C++: hiredis, redis-plus-plus
- Python: redis-py
- Go: Redigo

## Jedis Example

1, 引入Jedis Pom依赖

```
<dependency>
    <groupId>redis.clients</groupId>
    <artifactId>jedis</artifactId>
</dependency>
```

2, 初始化并调用API

```
Jedis jedis = new Jedis("127.0.0.1", 6379);
System.out.println("ping redis: " + jedis.ping());
jedis.set("key", "value");
System.out.println(jedis.get("key"));
```

3, 阿里云官方JedisPool连接池优化

[https://help.aliyun.com/document\\_detail/98726.html](https://help.aliyun.com/document_detail/98726.html)

# 开源Redis参数设置

`./src/redis-server redis.conf // 通过指定参数的方式启动`

- **bind** {default: bind 127.0.0.1}
- **protected-mode** {default: yes}
- **save** {default 3600 1 300 100 60 10000}
- **appendonly** {default no everysec}
- ...

## 02 云Redis开通与设置



# 云Redis开通及设置

- 购买实例
- 设置白名单
- 连接实例
- 账号管理
- 监控与日志

## 03 春运迁徙页面开发

声明：本页面不会实际获取用户地理位置信息，页面数据为测试数据，只做演示使用。

2021-03-11 17:14:16

您的姓名: jonny

您所在的位置: 浙江

您是否经过热门春运地域: 否

新闻通知:

- 2021-03-11 14:48 2021年春运 天水铁路累计发送旅客69.05 万人次
- 2021-03-11 12:45 浙江舟山 40天路畅人安，春运圆满收官
- 2021-03-11 11:57 锦州市交通 集团不放松不懈怠圆满完成春运

迁入迁出排名:

热力值	省份
15.52	广东省
6.95	浙江省
6.87	江苏省
6.47	河南省
6.35	山东省



用户信息存储采用 Hash结构



地理位置信息判断采用Tair性能增强型结构 TairGis，因为Redis原生的GEO不具有点与面关系判断的功能，无法实现此功能。主要用到的API为GIS.CONTAINS，用户判断用户的坐标与在哪个省中；GIS.INTERSECTS，判断用户是否经过热门春运地域。



本地简报功能采用Redis原生结构 Stream，给用户推送实时消息。主要用到的API有两个：一是XADD用来添加消息，二是XREVRANGE用来遍历返回消息。



迁入迁出排名采用Redis原生结构 Sorted Set，因为其本身按照score排序，因此极大的简化了开发逻辑，可以直接按照顺序返回。主要用到的API有两个：一是ZADD用来添加省份及对应的热力值，二是ZREVRANGE用来按照倒序返回信息。



# 功能演示与代码详解

- config: 负责初始化redisTemplate
- controller: 项目的http请求路由
- model: 项目中定义的class
- repo: 封装具体Redis的操作

```
springbootexample
├── config
│   └── BeanConfig
├── controller
│   ├── HotAreaController
│   ├── InsertDataController
│   ├── NewsController
│   ├── PositionController
│   ├── RankController
│   └── UserController
├── model
│   ├── HotArea
│   ├── News
│   ├── Position
│   ├── Rank
│   └── User
├── repo
│   ├── HotAreaRepository
│   ├── NewsRepository
│   ├── PositionRepository
│   ├── RankRepository
│   └── UserRepository
└── SpringBootExampleApplication
```



# TairGis - 轨迹漫游查询

- 典型的判断“线”和“面”的关系

漫游查询：根据一段行程轨迹判断路过哪些地方（如省、市、县等）

- 场景：判断一个人是经过疫区，电子围栏，红绿码。

```
x:6379> GIS.ADD country 河南 'POLYGON ((30 10, 40 40, 20 40,
10 20, 30 10)) '
(integer) 1
x:6379> GIS.ADD country 安徽 'POLYGON ((80 90, 80 80, 90 80,
90 90))'
(integer) 1
x:6379> GIS.ADD country 浙江 'POLYGON ((35 10, 45 45, 10 20,
35 10))'
(integer) 1
x:6379> GIS.INTERSECTS country 'LINESTRING (30 10, 10 30, 40
40) '
1) "0"
2) 1) "浙江"
2) "POLYGON((30 10,40 40,20 40,10 20,30 10))"
```



