

Day16

分布式锁模拟实现 12306 售票

锁

Curator实现分布式锁API

实现步骤

ZooKeeper 集群搭建实战

集群介绍

Leader选举

搭建要求

搭建实战

配置集群

集群启动

模拟集群异常

Zookeeper 核心理论

分布式特性补充 (Dubbo)

序列化问题

超时问题

多版本

灰度发布

负载均衡

集群容错

服务降级

砺锋健康项目

需求说明

技术架构图

功能架构

软件开发流程

需求分析

概要设计

[详细设计](#)

[编码](#)

[测试](#)

[软件交付](#)

[验收](#)

[维护](#)

[项目环境搭建分析](#)

[模块化开发](#)

[各模块职责定位](#)

[seehope-health-parent](#)

[seehope-health-common](#)

[seehope-health-interface](#)

[seehope-health-service_provider](#)

[seehope-health-backend](#)

[seehope-health-mobile](#)

[项目搭建](#)

[搭建 seehope-health-parent 模块](#)

[搭建 seehope-health-common 模块](#)

[搭建 seehope-health-interface 模块](#)

[搭建 seehope-health-service-provider 模块](#)

[搭建 seehope-health-backend 模块](#)

[ElementUI](#)

[介绍](#)

[如何使用](#)

[常用组件](#)

[Container 布局容器](#)

[Dropdown 下拉菜单](#)

[NavMenu 导航菜单](#)

[Table 表格](#)

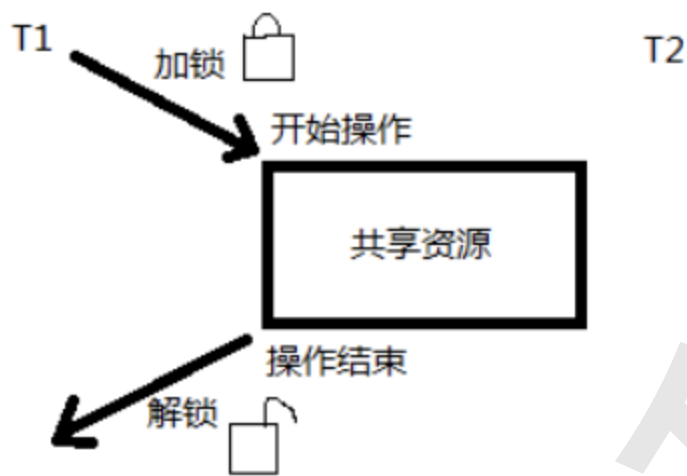
[Pagination 分页](#)

[Message 消息提示](#)

[Tabs 标签页](#)

分布式锁模拟实现 12306 售票

锁



Curator实现分布式锁API

在Curator中有五种锁方案

- InterProcessSemaphoreMutex：分布式排它锁（非可重入锁）
- InterProcessMutex：分布式可重入排它锁
- InterProcessReadWriteLock：分布式读写锁
- InterProcessMultiLock：将多个锁作为单个实体管理的容器
- InterProcessSemaphoreV2：共享信号量

实现步骤

这里介绍 InterProcessMutex，实现步骤：

1. 在根节点（持久化节点，eg: /lock）下创建临时有序节点。
2. 获取根节点下所有子节点，判断上一步中创建的临时节点是否是最小的节点。
 - a. 是：加锁成功
 - b. 不是：加锁失败，对前一个节点加watch，收到删除消息后，加锁成功。
3. 执行被锁住的代码块。
4. 删除自己在根节点下创建的节点，解锁成功。

```
1 package com.rushuni.curator;
2
3 import org.apache.curator.RetryPolicy;
4 import org.apache.curator.framework.CuratorFramework;
5 import org.apache.curator.framework.CuratorFrameworkFactory;
6 import org.apache.curator.framework.recipes.locks.InterProcessMutex;
7 import org.apache.curator.retry.ExponentialBackoffRetry;
8
9 import java.util.concurrent.TimeUnit;
10
11 /**
12  * @author rushuni
13  * @date 2021年07月30日 9:20 上午
14  */
15 public class Ticket12306 implements Runnable {
16
17     // 数据库中的票数
18     private int tickets = 10;
19     private InterProcessMutex lock ;
20
21     @Override
22     public void run() {
23         while(true){
24             //获取锁
25             try {
26                 lock.acquire(3L, TimeUnit.SECONDS);
27                 if(tickets > 0){
28                     System.out.println(Thread.currentThread() + ":" +
tickets);
29                     Thread.sleep(100L);
30                     tickets--;
31                 }
32             } catch (Exception e) {
33                 e.printStackTrace();
34             } finally {
35                 //释放锁
36                 try {
37                     lock.release();
38                 } catch (Exception e) {
39                     e.printStackTrace();
40                 }
41             }
42         }
43     }
44 }
45
46 public Ticket12306() {
47     // 重试策略
```

```

48     RetryPolicy retryPolicy = new ExponentialBackoffRetry(3000, 10);
49     // 初始化zk客户端
50     CuratorFramework client = CuratorFrameworkFactory.builder()
51         .connectString("10.211.55.5:2181")
52         .sessionTimeoutMs(60 * 1000)
53         .connectionTimeoutMs(15 * 1000)
54         .retryPolicy(retryPolicy)
55         .build();
56
57     // 开启连接
58     client.start();
59
60     lock = new InterProcessMutex(client, "/lock");
61 }
62 }

```

测试:

```

1  public static void main(String[] args) {
2      Ticket12306 ticket12306 = new Ticket12306();
3
4      //创建客户端
5      Thread t1 = new Thread(ticket12306, "携程");
6      Thread t2 = new Thread(ticket12306, "飞猪");
7
8      t1.start();
9      t2.start();
10 }

```

Java

复制代码

结果:

```

LockTest x
log4j:WARN See http://www.
Thread[携程,5,main]:10
Thread[飞猪,5,main]:9
Thread[携程,5,main]:8
Thread[飞猪,5,main]:7
Thread[携程,5,main]:6
Thread[飞猪,5,main]:5
Thread[携程,5,main]:4
Thread[飞猪,5,main]:3
Thread[携程,5,main]:2
Thread[飞猪,5,main]:1

```

ZooKeeper 集群搭建实战

集群介绍

Leader选举

- Serverid：服务器ID
 - 比如有三台服务器，编号分别是1，2，3。
 - 编号越大在选择算法中的权重越大。
- Zxid：数据ID
 - 服务器中存放的最大数据ID，值越大说明数据越新，在选举算法中数据越新权重越大。
- 在Leader选举的过程中，如果某台 ZooKeeper 获得了超过半数的选票，则此ZooKeeper就可以成为Leader了。

搭建要求

真实的集群是需要部署在不同的服务器上的，但是在我们测试时同时启动很多个虚拟机内存会吃不消，所以我们通常会搭建伪集群，也就是把所有的服务都搭建在一台虚拟机上，用端口进行区分。

我们这里搭建一个三个节点的 Zookeeper集群（伪集群）。

搭建实战

重新部署一台虚拟机作为我们搭建集群的测试服务器。

- 安装JDK。
- Zookeeper压缩包上传到服务器
- 将Zookeeper解压，建立/usr/local/zookeeper-cluster目录，将解压后的Zookeeper复制到以下三个目录

Shell  复制代码

```
1 /usr/local/zookeeper-cluster/zookeeper-1
2 /usr/local/zookeeper-cluster/zookeeper-2
3 /usr/local/zookeeper-cluster/zookeeper-3
```

Shell | 复制代码

```
1 [root@localhost ~]# mkdir /usr/local/zookeeper-cluster
2 [root@localhost ~]# cp -r apache-zookeeper-3.x.x-bin /usr/local/zookeeper-cluster/zookeeper-1
3 [root@localhost ~]# cp -r apache-zookeeper-3.x.x-bin /usr/local/zookeeper-cluster/zookeeper-2
4 [root@localhost ~]# cp -r apache-zookeeper-3.x.x-bin /usr/local/zookeeper-cluster/zookeeper-3
```

创建data目录，并且将 conf下zoo_sample.cfg 文件改名为 zoo.cfg

Shell | 复制代码

```
1 mkdir /usr/local/zookeeper-cluster/zookeeper-1/data
2 mkdir /usr/local/zookeeper-cluster/zookeeper-2/data
3 mkdir /usr/local/zookeeper-cluster/zookeeper-3/data
4
5 mv /usr/local/zookeeper-cluster/zookeeper-1/conf/zoo_sample.cfg
  /usr/local/zookeeper-cluster/zookeeper-1/conf/zoo.cfg
6 mv /usr/local/zookeeper-cluster/zookeeper-2/conf/zoo_sample.cfg
  /usr/local/zookeeper-cluster/zookeeper-2/conf/zoo.cfg
7 mv /usr/local/zookeeper-cluster/zookeeper-3/conf/zoo_sample.cfg
  /usr/local/zookeeper-cluster/zookeeper-3/conf/zoo.cfg
```

配置每一个Zookeeper 的dataDir 和 clientPort 分别为 2181 2182 2183

修改/usr/local/zookeeper-cluster/zookeeper-1/conf/zoo.cfg

Shell | 复制代码

```
1 vim /usr/local/zookeeper-cluster/zookeeper-1/conf/zoo.cfg
2
3 clientPort=2181
4 dataDir=/usr/local/zookeeper-cluster/zookeeper-1/data
```

修改/usr/local/zookeeper-cluster/zookeeper-2/conf/zoo.cfg

Shell | 复制代码

```
1 vim /usr/local/zookeeper-cluster/zookeeper-2/conf/zoo.cfg
2
3 clientPort=2182
4 dataDir=/usr/local/zookeeper-cluster/zookeeper-2/data
```

修改/usr/local/zookeeper-cluster/zookeeper-3/conf/zoo.cfg

Shell 复制代码

```
1 vim /usr/local/zookeeper-cluster/zookeeper-3/conf/zoo.cfg
2
3 clientPort=2183
4 dataDir=/usr/local/zookeeper-cluster/zookeeper-3/data
```

配置集群

在每个zookeeper的 data 目录下创建一个 myid 文件，内容分别是1、2、3。这个文件就是记录每个服务器的ID

Shell 复制代码

```
1 echo 1 > /usr/local/zookeeper-cluster/zookeeper-1/data/myid
2 echo 2 > /usr/local/zookeeper-cluster/zookeeper-2/data/myid
3 echo 3 > /usr/local/zookeeper-cluster/zookeeper-3/data/myid
```

在每一个 zookeeper 的 zoo.cfg 配置客户端访问端口（clientPort）和集群服务器IP列表。

集群服务器IP列表如下

Shell 复制代码

```
1 vim /usr/local/zookeeper-cluster/zookeeper-1/conf/zoo.cfg
2 vim /usr/local/zookeeper-cluster/zookeeper-2/conf/zoo.cfg
3 vim /usr/local/zookeeper-cluster/zookeeper-3/conf/zoo.cfg
4
5 # 每个配置文件一样
6 server.1=192.168.149.135:2881:3881
7 server.2=192.168.149.135:2882:3882
8 server.3=192.168.149.135:2883:3883
```

解释：server.服务器ID=服务器IP地址：服务器之间通信端口：服务器之间投票选举端口

集群启动

启动集群就是分别启动每个实例。


```
1 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh start
2 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh start
3 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh start
```

```
[parallels@CentOS-8 zookeeper-cluster]$ ./zookeeper-2/bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-2/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[parallels@CentOS-8 zookeeper-cluster]$ ./zookeeper-3/bin/zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-3/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
```

启动后我们查询一下每个实例的运行状态

```
1 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh status
2 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
3 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh status
```

```
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-1/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-1/bin/../conf/zoo.cfg
Client port found: 2181. Client address: localhost. Client SSL: false.
Mode: follower
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-2/bin/../conf/zoo.cfg
Client port found: 2182. Client address: localhost. Client SSL: false.
Mode: leader
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-3/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-3/bin/../conf/zoo.cfg
Client port found: 2183. Client address: localhost. Client SSL: false.
Mode: follower
```

Mode为follower表示是跟随者（从）

再查询第二个服务Mod 为leader表示是领导者（主）

模拟集群异常

首先我们测试如果是从服务器挂掉，会怎么样。

把3号服务器停掉，观察1号和2号，发现状态并没有变化

```
1 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh stop
2
3 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh status
4 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
```

```
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-3/bin/zkServer.sh stop
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-3/bin/../conf/zoo.cfg
Stopping zookeeper ... STOPPED
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-1/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-1/bin/../conf/zoo.cfg
Client port found: 2181. Client address: localhost. Client SSL: false.
Mode: follower
[parallels@CentOS-8 logs]$ sh /home/parallels/Documents/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /home/parallels/Documents/zookeeper-cluster/zookeeper-2/bin/../conf/zoo.cfg
Client port found: 2182. Client address: localhost. Client SSL: false.
Mode: leader
[parallels@CentOS-8 logs]$
```

由此得出结论，3个节点的集群，从服务器挂掉，集群正常

我们再把1号服务器（从服务器）也停掉，此时集群中只有一台服务器了。
查看2号（主服务器）的状态，发现已经停止运行了。

```
1 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh stop
2
3 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
```

由此得出结论，3个节点的集群，2个从服务器都挂掉，主服务器也无法运行。

因为可运行的机器没有超过集群总数量的半数。

我们再次把1号服务器启动起来，发现2号服务器又开始正常工作了。而且依然是领导者。

```
1 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh start
2
3 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
```

我们把3号服务器也启动起来，把2号服务器停掉,停掉后观察1号和3号的状态。

```
1 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh start
2 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh stop
3
4 /usr/local/zookeeper-cluster/zookeeper-1/bin/zkServer.sh status
5 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh status
```

发现新的leader产生了~

由此我们得出结论，当集群中的主服务器挂了，集群中的其他服务器会自动进行选举状态，然后产生新的 leader

我们再次测试，当我们把 2 号服务器重新启动起来启动后，会发生什么？2号服务器会再次成为新的领导吗？我们看结果

```
1 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh start
2
3 /usr/local/zookeeper-cluster/zookeeper-2/bin/zkServer.sh status
4 /usr/local/zookeeper-cluster/zookeeper-3/bin/zkServer.sh status
```

我们会发现，2号服务器启动后依然是跟随者（从服务器），3号服务器依然是领导者（主服务器），没有撼动3号服务器的领导地位。

由此我们得出结论，当领导者产生后，再次有新服务器加入集群，不会影响到现任领导者。

Zookeeper 核心理论

Zookeepe集群中的角色是其核心。

在ZooKeeper集群服务中有三个角色：

- Leader 领导者
 - 处理事务请求
 - 集群内部各服务器的调度者
- Follower 跟随者：
 - 处理客户端非事务请求，转发事务请求给Leader服务器
 - 参与Leader选举投票
- Observer 观察者：

- 处理客户端非事务请求，转发事务请求给Leader服务器

分布式特性补充（Dubbo）

序列化问题

1. dubbo 内部已经将序列化和反序列化的过程内部进行了封装，所以，我们在定义pojo类时，一定要实现serializable接口。
2. 一般会定义一个公共的pojo模块，让生产者和消费者都依赖该模块。

如：

```
1 public User implements Serializable { ... }
```

Java

复制代码

超时问题

- 服务消费者在调用服务提供者时发生了阻塞、等待的情形，这个时候，服务消费者会一直等待下去。
- 在某个峰值时刻，大量的请求都在同时请求服务提供者，会造成线程的大量堆积，势必会造成雪崩。
- dubbo 利用超时机制来解决这个问题，设置一个超时时间，在这个时间段内，无法完成服务访问，则自动断开连接。

使用 timeout 属性配置超时时间，默认值1000，单位毫秒。

```
1 //timeout 超时时间 单位毫秒 retries 重试次数  
2 @Service(timeout = 3000, retries=2)
```

Java

复制代码

和超时一起的，还有重试（retries），用于设置重试次数：

1. 设置了超时时间，在这个时间段内，无法完成服务访问，则自动断开连接。
2. 如果出现网络抖动，则这一次请求就会失败。
3. Dubbo提供重试机制来避免类似问题的发生。
4. 通过retries属性来设置重试次数。默认为2次。

多版本

灰度发布

当出现新功能时，会让一部分用户先使用新功能，用户反馈没问题时，再将所有用户迁移到新功能。

dubbo 中可以使用 version 属性来设置和调用同一个接口的不同版本。

生产者配置

```
1 @Service(version="v2.0")
2 public class UserServiceImp12 implements UserService { ... }
3 @Service(version="v1.0")
4 public class UserServiceImp13 implements UserService { ... }
```

Java

复制代码

消费者配置

```
1 // 远程注入时指定版本
2 @Reference(version = "v2.0")
3 private UserService userService;
```

Java

复制代码

负载均衡

负载均衡策略：

- Random
 - 按权重随机，默认值。按权重设置随机概率。
- RoundRobin
 - 按权重轮询。
- LeastActive
 - 最少活跃调用数,相同活跃数的随机。
- ConsistentHash
 - 一致性Hash,相同参数的请求总是发到同一提供者。

服务提供者配置

```
1 @Service(weight = 100)
2 public class UserServiceImp12 implements UserService { ... }
```

消费者配置

```
1 // @Reference(loadbalance = "roundrobin")
2 // @Reference(loadbalance = "leastactive")
3 // @Reference(loadbalance = "consistenthash")
4 @Reference(loadbalance = "random") // 默认 按权重随机
5 private UserService userService;
```

集群容错

集群容错模式

- Failover Cluster
 - 失败重试。默认值。当出现失败，重试其它服务器，默认重试2次，使用retries配置。
 - 一般用于读操作
- Failfast Cluster
 - 快速失败,发起一次调用，失败立即报错。
 - 通常用于写操作。
- Failsafe Cluster
 - 失败安全，出现异常时，直接忽略。返回一个空结果。
- Failback Cluster
 - 失败自动恢复,后台记录失败请求,定时重发。
- Forking Cluster
 - 并行调用多个服务器，只要一个成功即返回。
- Broadcast Cluster
 - 广播调用所有提供者,逐个调用，任意一台报错则报错。

消费者配置

```
1 // 远程注入
2 @Reference(cluster = "failover")
3 private UserService userService;
```

服务降级

当服务器压力剧增的情况下，根据实际业务情况及流量，对一些服务和页面有策略的不处理或换种简单的方式处理，从而释放服务器资源以保证核心业务正常运作或高效运作。

服务降级方式

- mock= force:return null
 - 表示消费方对该服务的方法调用都直接返回 null 值,不发起远程调用。
 - 用来屏蔽不重要服务不可用时对调用方的影响。
- mock=fail:return null
 - 表示消费方对该服务的方法调用在失败后，再返回 null 值,不抛异常。
 - 用来容忍不重要服务不稳定时对调用方的影响

消费方配置

```
1 //远程注入
2 @Reference(mock ="force: return null") // 不再调用userService的服务
3 private UserService userService;
```

Java

复制代码

砺锋健康项目

需求说明

健康管理系统是一款应用于健康管理机构的业务系统，系统集成了如下能力：

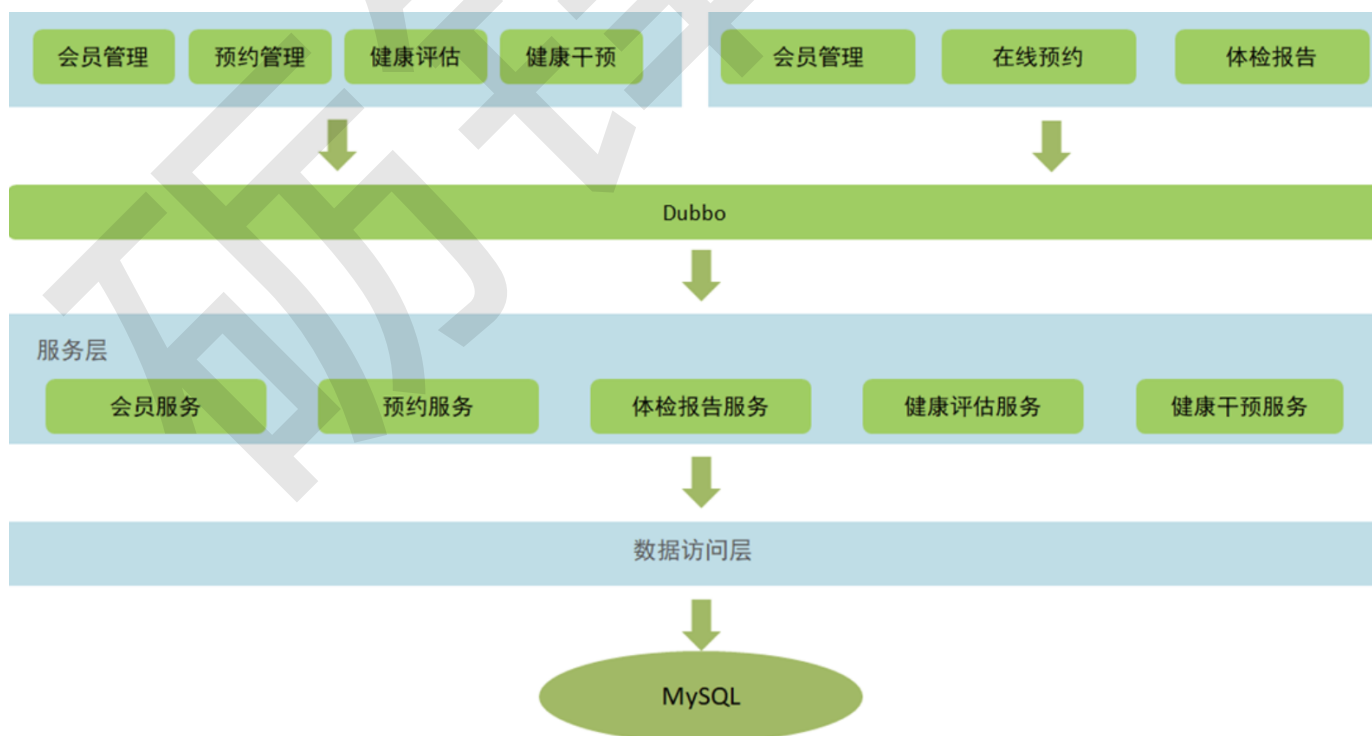
- 健康管理机构工作内容可视化
- 会员管理专业化
- 健康评估数字化
- 健康干预流程化
- 知识库集成化

系统目标是提高健康管理师的工作效率，加强与会员间的互动，增强管理者对健康管理机构运营情况的了解。

技术架构图



功能架构



软件开发流程

需求分析

1. 相关系统分析员向用户初步了解需求，然后用相关的工具软件列出要开发的系统的大功能模块，每个大功能模块有哪些小功能模块，对于有些需求比较明确相关的界面时，在这一步里面可以初步定义好少量的界面。
2. 系统分析员深入了解和分析需求，根据自己的经验和需求用WORD或相关的工具再做出一份文档系统的功能需求文档。这次的文档会清楚列出系统大致的大功能模块，大功能模块有哪些小功能模块，并且还列出相关的界面和界面功能。
3. 系统分析员向用户再次确认需求。

概要设计

首先，开发者需要对软件系统进行概要设计，即系统设计。概要设计需要对软件系统的设计进行考虑，包括系统的基本处理流程、系统的组织结构、模块划分、功能分配、接口设计、运行设计、数据结构设计和出错处理设计等，为软件的详细设计提供基础。

详细设计

在概要设计的基础上，开发者需要进行软件系统的详细设计。

在详细设计中，描述实现具体模块所涉及到的主要算法、数据结构、类的层次结构及调用关系，需要说明软件系统各个层次中的每一个程序(每个模块或子程序)的设计考虑，以便进行编码和测试。

应当保证软件的需求完全分配给整个软件。

详细设计应当足够详细，能够根据详细设计报告进行编码。

编码

在软件编码阶段，开发者根据《软件系统详细设计报告》中对数据结构、算法分析和模块实现等方面的设计要求，开始具体的编写程序工作，分别实现各模块的功能，从而实现对目标系统的功能、性能、接口、界面等方面的要求。

在规范化的研发流程中，编码工作在整个项目流程里最多不会超过1/2，通常在1/3的时间，所谓磨刀不误砍柴功，设计过程完成的好，编码效率就会极大提高，编码时不同模块之间的进度协调和协作是最需要小心的，也许一个小模块的问题就可能影响了整体进度，让很多程序员因此被迫停下工作等待，这种问题在很多研发过程中都出现过。

编码时的相互沟通和应急的解决手段都是相当重要的，对于程序员而言，bug永远存在，你必须永远面对这个问题！

测试

测试编写好的系统。交给用户使用，用户使用后一个一个的确认每个功能。

软件测试有很多种：按照测试执行方，可以分为内部测试和外部测试；按照测试范围，可以分为模块测试和整体联调；按照测试条件，可以分为正常操作情况测试和异常情况测试；按照测试的输入范围，可以分为全覆盖测试和抽样测试。

以上都很好理解，不多解释。

总之，测试同样是项目研发中一个相当重要的步骤，对于一个大型软件，3个月到1年的外部测试都是正常的，因为永远都会有不可预料的问题存在。

完成测试后，完成验收并完成最后的一些帮助文档，整体项目才算告一段落，当然日后少不了升级，修补等工作，只要不是想通过一锤子买卖骗钱，就要不停的跟踪软件的运营状况并持续修补升级，直到这个软件被彻底淘汰为止。

软件交付

在软件测试证明软件达到要求后，软件开发者应向用户提交开发的目标安装程序、数据库的数据字典、《用户安装手册》、《用户使用指南》、需求报告、设计报告、测试报告等双方合同约定的产物。

《用户安装手册》应详细介绍安装软件对运行环境的要求、安装软件的定义和内容、在客户端、服务器端及中间件的具体安装步骤、安装后的系统配置。

《用户使用指南》应包括软件各项功能的使用流程、操作步骤、相应业务介绍、特殊提示和注意事项等方面的内容，在需要时还应举例说明。

验收

用户验收。

维护

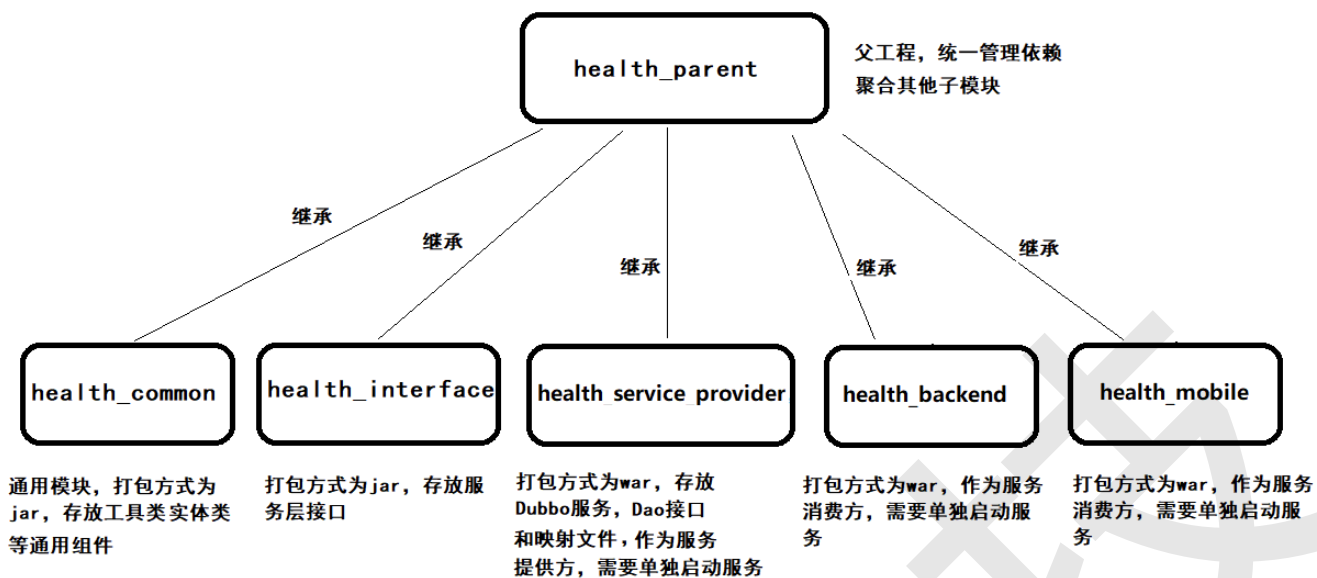
根据用户需求的变化或环境的变化，对应用程序进行全部或部分的修改。

项目环境搭建分析

模块化开发

采用maven模块化开发方式，即对整个项目拆分为几个maven工程，每个maven工程存放特定的一类代码。

具体如下图：



各模块职责定位

seehope-health-parent

- 父工程
- 统一锁定依赖的版本，同时聚合其他子模块便于统一执行 maven 命令
- 打包方式为pom

seehope-health-common

- 通用模块
- 存放项目中使用到的一些工具类、实体类、返回结果和常量类
- 打包方式为jar

seehope-health-interface

- 存放服务接口
- 打包方式为jar

seehope-health-service_provider

- Dubbo 服务模块
- 存放服务实现类、Dao接口、Mapper映射文件等
- 打包方式为war
- 作为服务提供方，需要部署到服务器中运行

seehope-health-backend

- 砺锋健康管理后台

- 作为 Dubbo 服务消费方，存放 Controller、HTML页面、js、css、spring配置文件等
- 打包方式为war
- 需要部署到服务器中运行

seehope-health-mobile

- 移动端前台

项目搭建

通过前面的项目功能架构图可以知道本项目分为砺锋健康管理后台和砺锋健康前台（微信端）

搭建 seehope-health-parent 模块

创建 seehope-health-parent，父工程，打包方式为pom，用于统一管理依赖版本

pom.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>com.rushuni</groupId>
8      <artifactId>seehope-health-parent</artifactId>
9      <version>1.0-SNAPSHOT</version>
10     <packaging>pom</packaging>
11
12     <properties>
13         <maven.compiler.source>11</maven.compiler.source>
14         <maven.compiler.target>11</maven.compiler.target>
15         <junit.version>4.12</junit.version>
16         <spring.version>5.1.9.RELEASE</spring.version>
17         <pagehelper.version>4.1.4</pagehelper.version>
18         <servlet-api.version>2.5</servlet-api.version>
19         <dubbo.version>2.7.7</dubbo.version>
20         <zookeeper.version>3.4.7</zookeeper.version>
21         <zkclient.version>0.1</zkclient.version>
22         <mybatis.version>3.4.5</mybatis.version>
23         <mybatis.spring.version>1.3.1</mybatis.spring.version>
24         <mybatis.paginator.version>1.2.15</mybatis.paginator.version>
25         <mysql.version>5.1.32</mysql.version>
26         <druid.version>1.0.9</druid.version>
27         <commons-fileupload.version>1.3.1</commons-fileupload.version>
28         <spring.security.version>5.0.5.RELEASE</spring.security.version>
29         <poi.version>3.14</poi.version>
30         <jedis.version>2.9.0</jedis.version>
31         <quartz.version>2.2.1</quartz.version>
32     </properties>
33     <!-- 依赖管理标签 必须加 -->
34     <dependencyManagement>
35         <dependencies>
36             <!-- Spring -->
37             <dependency>
38                 <groupId>org.springframework</groupId>
39                 <artifactId>spring-context</artifactId>
40                 <version>${spring.version}</version>
41             </dependency>
42             <dependency>
43                 <groupId>org.springframework</groupId>
44                 <artifactId>spring-beans</artifactId>
45                 <version>${spring.version}</version>
46             </dependency>
47             <dependency>

```

```

48         <groupId>org.springframework</groupId>
49         <artifactId>spring-web</artifactId>
50         <version>${spring.version}</version>
51     </dependency>
52     <dependency>
53         <groupId>org.springframework</groupId>
54         <artifactId>spring-webmvc</artifactId>
55         <version>${spring.version}</version>
56     </dependency>
57     <dependency>
58         <groupId>org.springframework</groupId>
59         <artifactId>spring-jdbc</artifactId>
60         <version>${spring.version}</version>
61     </dependency>
62     <dependency>
63         <groupId>org.springframework</groupId>
64         <artifactId>spring-aspects</artifactId>
65         <version>${spring.version}</version>
66     </dependency>
67     <dependency>
68         <groupId>org.springframework</groupId>
69         <artifactId>spring-jms</artifactId>
70         <version>${spring.version}</version>
71     </dependency>
72     <dependency>
73         <groupId>org.springframework</groupId>
74         <artifactId>spring-context-support</artifactId>
75         <version>${spring.version}</version>
76     </dependency>
77     <dependency>
78         <groupId>org.springframework</groupId>
79         <artifactId>spring-test</artifactId>
80         <version>${spring.version}</version>
81     </dependency>
82     <!-- dubbo相关 -->
83     <dependency>
84         <groupId>com.alibaba</groupId>
85         <artifactId>dubbo</artifactId>
86         <version>${dubbo.version}</version>
87     </dependency>
88     <dependency>
89         <groupId>org.apache.zookeeper</groupId>
90         <artifactId>zookeeper</artifactId>
91         <version>${zookeeper.version}</version>
92     </dependency>
93     <dependency>
94         <groupId>com.github.sgroschupf</groupId>
95         <artifactId>zkclient</artifactId>
96         <version>${zkclient.version}</version>
97     </dependency>

```

```

98     <dependency>
99         <groupId>junit</groupId>
100         <artifactId>junit</artifactId>
101         <version>4.12</version>
102     </dependency>
103     <dependency>
104         <groupId>com.alibaba</groupId>
105         <artifactId>fastjson</artifactId>
106         <version>1.2.47</version>
107     </dependency>
108     <dependency>
109         <groupId>javassist</groupId>
110         <artifactId>javassist</artifactId>
111         <version>3.12.1.GA</version>
112     </dependency>
113     <dependency>
114         <groupId>commons-codec</groupId>
115         <artifactId>commons-codec</artifactId>
116         <version>1.10</version>
117     </dependency>
118     <dependency>
119         <groupId>com.github.pagehelper</groupId>
120         <artifactId>pagehelper</artifactId>
121         <version>${pagehelper.version}</version>
122     </dependency>
123     <!-- Mybatis -->
124     <dependency>
125         <groupId>org.mybatis</groupId>
126         <artifactId>mybatis</artifactId>
127         <version>${mybatis.version}</version>
128     </dependency>
129     <dependency>
130         <groupId>org.mybatis</groupId>
131         <artifactId>mybatis-spring</artifactId>
132         <version>${mybatis.spring.version}</version>
133     </dependency>
134     <dependency>
135         <groupId>com.github.miemiedev</groupId>
136         <artifactId>mybatis-paginator</artifactId>
137         <version>${mybatis.paginator.version}</version>
138     </dependency>
139     <!-- MySql -->
140     <dependency>
141         <groupId>mysql</groupId>
142         <artifactId>mysql-connector-java</artifactId>
143         <version>${mysql.version}</version>
144     </dependency>
145     <!-- 连接池 -->
146     <dependency>
147         <groupId>com.alibaba</groupId>

```

```

148         <artifactId>druid</artifactId>
149         <version>${druid.version}</version>
150     </dependency>
151     <!-- 文件上传组件 -->
152     <dependency>
153         <groupId>commons-fileupload</groupId>
154         <artifactId>commons-fileupload</artifactId>
155         <version>${commons-fileupload.version}</version>
156     </dependency>
157     <dependency>
158         <groupId>org.quartz-scheduler</groupId>
159         <artifactId>quartz</artifactId>
160         <version>${quartz.version}</version>
161     </dependency>
162     <dependency>
163         <groupId>org.quartz-scheduler</groupId>
164         <artifactId>quartz-jobs</artifactId>
165         <version>${quartz.version}</version>
166     </dependency>
167     <dependency>
168         <groupId>com.sun.jersey</groupId>
169         <artifactId>jersey-client</artifactId>
170         <version>1.18.1</version>
171     </dependency>
172     <dependency>
173         <groupId>com.qiniu</groupId>
174         <artifactId>qiniu-java-sdk</artifactId>
175         <version>7.2.0</version>
176     </dependency>
177     <!--POI报表-->
178     <dependency>
179         <groupId>org.apache.poi</groupId>
180         <artifactId>poi</artifactId>
181         <version>${poi.version}</version>
182     </dependency>
183     <dependency>
184         <groupId>org.apache.poi</groupId>
185         <artifactId>poi-ooxml</artifactId>
186         <version>${poi.version}</version>
187     </dependency>
188     <dependency>
189         <groupId>redis.clients</groupId>
190         <artifactId>jedis</artifactId>
191         <version>${jedis.version}</version>
192     </dependency>
193     <!-- 安全框架 -->
194     <dependency>
195         <groupId>org.springframework.security</groupId>
196         <artifactId>spring-security-web</artifactId>
197         <version>${spring.security.version}</version>

```



```

198         </dependency>
199         <dependency>
200             <groupId>org.springframework.security</groupId>
201             <artifactId>spring-security-config</artifactId>
202             <version>${spring.security.version}</version>
203         </dependency>
204         <dependency>
205             <groupId>org.springframework.security</groupId>
206             <artifactId>spring-security-taglibs</artifactId>
207             <version>${spring.security.version}</version>
208         </dependency>
209         <dependency>
210             <groupId>com.github.penggle</groupId>
211             <artifactId>kaptcha</artifactId>
212             <version>2.3.2</version>
213             <exclusions>
214                 <exclusion>
215                     <groupId>javax.servlet</groupId>
216                     <artifactId>javax.servlet-api</artifactId>
217                 </exclusion>
218             </exclusions>
219         </dependency>
220         <dependency>
221             <groupId>dom4j</groupId>
222             <artifactId>dom4j</artifactId>
223             <version>1.6.1</version>
224         </dependency>
225         <dependency>
226             <groupId>xml-apis</groupId>
227             <artifactId>xml-apis</artifactId>
228             <version>1.4.01</version>
229         </dependency>
230     </dependencies>
231 </dependencyManagement>
232 <dependencies>
233     <dependency>
234         <groupId>javax.servlet</groupId>
235         <artifactId>servlet-api</artifactId>
236         <version>${servlet-api.version}</version>
237         <scope>provided</scope>
238     </dependency>
239 </dependencies>
240 <build>
241     <plugins>
242         <!-- java编译插件 -->
243         <plugin>
244             <groupId>org.apache.maven.plugins</groupId>
245             <artifactId>maven-compiler-plugin</artifactId>
246             <version>3.2</version>
247             <configuration>

```

```
248         <source>1.8</source>
249         <target>1.8</target>
250         <encoding>UTF-8</encoding>
251     </configuration>
252 </plugin>
253 </plugins>
254 </build>
255 </project>
```

搭建 seehope-health-common 模块

pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <parent>
6         <artifactId>seehope-health-parent</artifactId>
7         <groupId>com.rushuni</groupId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10    <modelVersion>4.0.0</modelVersion>
11
12    <artifactId>seehope-health-common</artifactId>
13
14    <properties>
15        <maven.compiler.source>11</maven.compiler.source>
16        <maven.compiler.target>11</maven.compiler.target>
17    </properties>
18
19    <dependencies>
20        <dependency>
21            <groupId>com.github.pagehelper</groupId>
22            <artifactId>pagehelper</artifactId>
23        </dependency>
24        <!-- Mybatis -->
25        <dependency>
26            <groupId>org.mybatis</groupId>
27            <artifactId>mybatis</artifactId>
28        </dependency>
29        <dependency>
30            <groupId>org.mybatis</groupId>
31            <artifactId>mybatis-spring</artifactId>
32        </dependency>
33        <dependency>
34            <groupId>com.github.miemiedev</groupId>
35            <artifactId>mybatis-paginator</artifactId>
36        </dependency>
37        <!-- MySql -->
38        <dependency>
39            <groupId>mysql</groupId>
40            <artifactId>mysql-connector-java</artifactId>
41        </dependency>
42        <!-- 连接池 -->
43        <dependency>
44            <groupId>com.alibaba</groupId>
45            <artifactId>druid</artifactId>
46        </dependency>
47        <dependency>
```

```
48         <groupId>commons-fileupload</groupId>
49         <artifactId>commons-fileupload</artifactId>
50     </dependency>
51     <!-- Spring -->
52     <dependency>
53         <groupId>org.springframework</groupId>
54         <artifactId>spring-context</artifactId>
55     </dependency>
56     <dependency>
57         <groupId>org.springframework</groupId>
58         <artifactId>spring-beans</artifactId>
59     </dependency>
60     <dependency>
61         <groupId>org.springframework</groupId>
62         <artifactId>spring-web</artifactId>
63     </dependency>
64     <dependency>
65         <groupId>org.springframework</groupId>
66         <artifactId>spring-webmvc</artifactId>
67     </dependency>
68     <dependency>
69         <groupId>org.springframework</groupId>
70         <artifactId>spring-jdbc</artifactId>
71     </dependency>
72     <dependency>
73         <groupId>org.springframework</groupId>
74         <artifactId>spring-aspects</artifactId>
75     </dependency>
76     <dependency>
77         <groupId>org.springframework</groupId>
78         <artifactId>spring-jms</artifactId>
79     </dependency>
80     <dependency>
81         <groupId>org.springframework</groupId>
82         <artifactId>spring-context-support</artifactId>
83     </dependency>
84     <dependency>
85         <groupId>org.springframework</groupId>
86         <artifactId>spring-test</artifactId>
87     </dependency>
88     <!-- dubbo相关 -->
89     <dependency>
90         <groupId>com.alibaba</groupId>
91         <artifactId>dubbo</artifactId>
92     </dependency>
93     <dependency>
94         <groupId>org.apache.zookeeper</groupId>
95         <artifactId>zookeeper</artifactId>
96     </dependency>
97     <dependency>
```

```
98         <groupId>com.github.sgroschupf</groupId>
99         <artifactId>zkclient</artifactId>
100     </dependency>
101     <dependency>
102         <groupId>junit</groupId>
103         <artifactId>junit</artifactId>
104     </dependency>
105     <dependency>
106         <groupId>com.alibaba</groupId>
107         <artifactId>fastjson</artifactId>
108     </dependency>
109     <dependency>
110         <groupId>javassist</groupId>
111         <artifactId>javassist</artifactId>
112     </dependency>
113     <dependency>
114         <groupId>commons-codec</groupId>
115         <artifactId>commons-codec</artifactId>
116     </dependency>
117     <dependency>
118         <groupId>org.apache.poi</groupId>
119         <artifactId>poi</artifactId>
120     </dependency>
121     <dependency>
122         <groupId>redis.clients</groupId>
123         <artifactId>jedis</artifactId>
124     </dependency>
125     <dependency>
126         <groupId>com.qiniu</groupId>
127         <artifactId>qiniu-java-sdk</artifactId>
128     </dependency>
129     <dependency>
130         <groupId>com.sun.jersey</groupId>
131         <artifactId>jersey-client</artifactId>
132     </dependency>
133     <dependency>
134         <groupId>org.apache.poi</groupId>
135         <artifactId>poi-ooxml</artifactId>
136     </dependency>
137     <dependency>
138         <groupId>org.springframework.security</groupId>
139         <artifactId>spring-security-web</artifactId>
140     </dependency>
141     <dependency>
142         <groupId>org.springframework.security</groupId>
143         <artifactId>spring-security-config</artifactId>
144     </dependency>
145     <dependency>
146         <groupId>org.springframework.security</groupId>
147         <artifactId>spring-security-taglibs</artifactId>
```

```
148         </dependency>
149     </dependencies>
150
151 </project>
```

搭建 seehope-health-interface 模块

pom.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5          http://maven.apache.org/xsd/maven-4.0.0.xsd">
6      <parent>
7          <artifactId>health-parent</artifactId>
8          <groupId>com.rushuni</groupId>
9          <version>1.0-SNAPSHOT</version>
10     </parent>
11     <modelVersion>4.0.0</modelVersion>
12     <artifactId>health-interface</artifactId>
13
14     <properties>
15         <maven.compiler.source>11</maven.compiler.source>
16         <maven.compiler.target>11</maven.compiler.target>
17     </properties>
18
19     <dependencies>
20         <dependency>
21             <groupId>com.rushuni</groupId>
22             <artifactId>health-common</artifactId>
23             <version>1.0-SNAPSHOT</version>
24         </dependency>
25     </dependencies>
26
27
28
29 </project>
```

XML

复制代码

搭建 seehope-health-service-provider 模块

注意打包方式为war，作为服务单独部署，存放服务类、Dao接口和Mapper映射文件等

码点科技

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6   http://maven.apache.org/xsd/maven-4.0.0.xsd">
7   <modelVersion>4.0.0</modelVersion>
8
9   <groupId>com.rushuni</groupId>
10  <artifactId>health_service_provider</artifactId>
11  <version>1.0-SNAPSHOT</version>
12  <packaging>war</packaging>
13
14  <name>health_service_provider Maven Webapp</name>
15  <!-- FIXME change it to the project's website -->
16  <url>http://www.example.com</url>
17
18  <properties>
19    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
20    <maven.compiler.source>11</maven.compiler.source>
21    <maven.compiler.target>11</maven.compiler.target>
22  </properties>
23
24  <dependencies>
25    <dependency>
26      <groupId>com.rushuni</groupId>
27      <artifactId>health-interface</artifactId>
28      <version>1.0-SNAPSHOT</version>
29    </dependency>
30  </dependencies>
31
32  <build>
33    <plugins>
34      <plugin>
35        <groupId>org.apache.tomcat.maven</groupId>
36        <artifactId>tomcat7-maven-plugin</artifactId>
37        <configuration>
38          <!-- 指定端口 -->
39          <port>81</port>
40          <!-- 请求路径 -->
41          <path>/</path>
42        </configuration>
43      </plugin>
44    </plugins>
45  </build>
46</project>
```


web.xml

XML | 复制代码

```
1 <!DOCTYPE web-app PUBLIC
2     "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3     "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5 <web-app>
6     <display-name>Archetype Created Web Application</display-name>
7     <!-- 加载spring容器 -->
8     <context-param>
9         <param-name>contextConfigLocation</param-name>
10        <param-value>classpath*:spring*.xml</param-value>
11    </context-param>
12    <listener>
13        <listener-
14            class>org.springframework.web.context.ContextLoaderListener</listener-class>
15    </listener>
16 </web-app>
```

log4j.properties

Java | 复制代码

```
1 ### direct log messages to stdout ###
2 log4j.appender.stdout=org.apache.log4j.ConsoleAppender
3 log4j.appender.stdout.Target=System.err
4 log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
5 log4j.appender.stdout.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L -
6 %m%n
7
8 ### direct messages to file mylog.log ###
9 log4j.appender.file=org.apache.log4j.FileAppender
10 #log4j.appender.file.File=c:\\mylog.log
11 log4j.appender.file.File=../logs/mylog.log
12 log4j.appender.file.layout=org.apache.log4j.PatternLayout
13 log4j.appender.file.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L - %m%n
14
15 ### set log levels - for more verbose logging change 'info' to 'debug' ###
16 log4j.rootLogger=debug, stdout
```

spring-dao.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3      xmlns:context="http://www.springframework.org/schema/context"
4      xmlns:p="http://www.springframework.org/schema/p"
5      xmlns:aop="http://www.springframework.org/schema/aop"
6      xmlns:tx="http://www.springframework.org/schema/tx"
7      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
8      xsi:schemaLocation="http://www.springframework.org/schema/beans
9          http://www.springframework.org/schema/beans/spring-beans-
10             4.2.xsd
11             http://www.springframework.org/schema/context
12             http://www.springframework.org/schema/context/spring-
13             context.xsd
14             http://www.springframework.org/schema/aop
15             http://www.springframework.org/schema/aop/spring-aop.xsd
16             http://www.springframework.org/schema/tx
17             http://www.springframework.org/schema/tx/spring-tx.xsd
18             http://www.springframework.org/schema/util
19             http://www.springframework.org/schema/util/spring-util.xsd">
20      <!--数据源-->
21      <bean id="dataSource"
22          class="com.alibaba.druid.pool.DruidDataSource" destroy-
23          method="close">
24          <property name="username" value="rush" />
25          <property name="password" value="rush123!" />
26          <property name="driverClassName" value="com.mysql.cj.jdbc.Driver" />
27          <property name="url" value="jdbc:mysql://localhost:3306/health" />
28      </bean>
29      <!--spring和mybatis整合的工厂bean-->
30      <bean id="sqlSessionFactory"
31          class="org.mybatis.spring.SqlSessionFactoryBean">
32          <property name="dataSource" ref="dataSource" />
33          <property name="configLocation" value="classpath:SqlMapConfig.xml" />
34      </bean>
35      <!--批量扫描接口生成代理对象-->
36      <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
37          <!--指定接口所在的包-->
38          <property name="basePackage" value="com.rushuni.dao" />
39      </bean>
40  </beans>

```

spring-service.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
6       xmlns:mvc="http://www.springframework.org/schema/mvc"
7       xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9       http://www.springframework.org/schema/beans/spring-beans.xsd
10      http://www.springframework.org/schema/mvc
11      http://www.springframework.org/schema/mvc/spring-
12      mvc.xsd
13      http://code.alibabatech.com/schema/dubbo
14      http://code.alibabatech.com/schema/dubbo/dubbo.xsd
15      http://www.springframework.org/schema/context
16      http://www.springframework.org/schema/context/spring-context.xsd">
17   <!-- 指定应用名称 -->
18   <dubbo:application name="health_service_provider"/>
19   <!--指定暴露服务的端口，之前没有指定，也就是使用默认的20880-->
20   <dubbo:protocol name="dubbo" port="20887"/>
21   <!--指定服务注册中心地址-->
22   <dubbo:registry address="zookeeper://10.211.55.5:2181"/>
23   <!--批量扫描，发布服务-->
24   <dubbo:annotation package="com.rushuni.service"/>
25 </beans>
```

spring-tx.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xmlns:context="http://www.springframework.org/schema/context"
5         xmlns:tx="http://www.springframework.org/schema/tx"
6         xmlns:mvc="http://www.springframework.org/schema/mvc"
7         xsi:schemaLocation="http://www.springframework.org/schema/beans
8                             http://www.springframework.org/schema/beans/spring-
9                             beans.xsd
10                             http://www.springframework.org/schema/mvc
11                             http://www.springframework.org/schema/mvc/spring-
12                             mvc.xsd
13                             http://www.springframework.org/schema/tx
14                             http://www.springframework.org/schema/tx/spring-
15                             tx.xsd
16                             http://www.springframework.org/schema/context
17                             http://www.springframework.org/schema/context/spring-
18                             context.xsd">
19      <!-- 事务管理器 -->
20      <bean id="transactionManager"
21            class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
22          <property name="dataSource" ref="dataSource"/>
23        </bean>
24      <!--
25      开启事务控制的注解支持
26      注意：此处必须加入proxy-target-class="true",
27      需要进行事务控制，会由Spring框架产生代理对象，
28      Dubbo需要将Service发布为服务，要求必须使用cglib创建代理对象。
29      -->
30      <tx:annotation-driven transaction-manager="transactionManager"
31                            proxy-target-class="true"/>
32    </beans>

```

SqlMapConfig.xml

```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
3     "http://mybatis.org/dtd/mybatis-3-config.dtd">
4 <configuration>
5     <plugins>
6         <!-- com.github.pagehelper 为 PageHelper 类所在包名 -->
7         <plugin interceptor="com.github.pagehelper.PageHelper">
8             <!-- 设置数据库类型 Oracle,MySQL,MariaDB,SQLite,Hsqldb,PostgreSQL 六
种数据库-->
9             <property name="dialect" value="mysql"/>
10        </plugin>
11    </plugins>
12 </configuration>
```

搭建 seehope-health-backend 模块

注意是 war 打包方式。

pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6     http://maven.apache.org/xsd/maven-4.0.0.xsd">
7   <modelVersion>4.0.0</modelVersion>
8
9   <groupId>com.rushuni</groupId>
10  <artifactId>seehope-health-backend</artifactId>
11  <version>1.0-SNAPSHOT</version>
12  <packaging>war</packaging>
13
14  <name>health-backend Maven Webapp</name>
15  <!-- FIXME change it to the project's website -->
16  <url>http://www.example.com</url>
17
18  <properties>
19    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
20    <maven.compiler.source>11</maven.compiler.source>
21    <maven.compiler.target>11</maven.compiler.target>
22  </properties>
23
24  <dependencies>
25    <dependency>
26      <groupId>com.rushuni</groupId>
27      <artifactId>health-interface</artifactId>
28      <version>1.0-SNAPSHOT</version>
29    </dependency>
30  </dependencies>
31
32  <build>
33    <plugins>
34      <plugin>
35        <groupId>org.apache.tomcat.maven</groupId>
36        <artifactId>tomcat7-maven-plugin</artifactId>
37        <configuration>
38          <!-- 指定端口 -->
39          <port>81</port>
40          <!-- 请求路径 -->
41          <path>/</path>
42        </configuration>
43      </plugin>
44    </plugins>
45  </build>
46</project>
```

```
1  <!DOCTYPE web-app PUBLIC
2      "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3      "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5  <web-app>
6      <display-name>Archetype Created Web Application</display-name>
7      <!-- 解决乱码 -->
8      <filter>
9          <filter-name>CharacterEncodingFilter</filter-name>
10         <filter-
11             class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
12             <init-param>
13                 <param-name>encoding</param-name>
14                 <param-value>utf-8</param-value>
15             </init-param>
16             <init-param>
17                 <param-name>forceEncoding</param-name>
18                 <param-value>true</param-value>
19             </init-param>
20         </filter>
21         <filter-mapping>
22             <filter-name>CharacterEncodingFilter</filter-name>
23             <url-pattern>/*</url-pattern>
24         </filter-mapping>
25         <servlet>
26             <servlet-name>springmvc</servlet-name>
27             <servlet-
28                 class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
29                 <!-- 指定加载的配置文件，通过参数contextConfigLocation加载 -->
30                 <init-param>
31                     <param-name>contextConfigLocation</param-name>
32                     <param-value>classpath:springmvc.xml</param-value>
33                 </init-param>
34                 <load-on-startup>1</load-on-startup>
35             </servlet>
36             <servlet-mapping>
37                 <servlet-name>springmvc</servlet-name>
38                 <url-pattern>/</url-pattern>
39             </servlet-mapping>
40         </web-app>
```

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xmlns:context="http://www.springframework.org/schema/context"
5         xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
6         xmlns:mvc="http://www.springframework.org/schema/mvc"
7         xsi:schemaLocation="http://www.springframework.org/schema/beans
8                             http://www.springframework.org/schema/beans/spring-beans.xsd
9                             http://www.springframework.org/schema/mvc
10                            http://www.springframework.org/schema/mvc/spring-mvc.xsd
11                            http://code.alibabatech.com/schema/dubbo
12                            http://code.alibabatech.com/schema/dubbo/dubbo.xsd
13                            http://www.springframework.org/schema/context
14                            http://www.springframework.org/schema/context/spring-
context.xsd">
15    <mvc:annotation-driven> </mvc:annotation-driven>
16    <!-- 指定应用名称 -->
17    <dubbo:application name="health_backend" />
18    <!--指定服务注册中心地址-->
19    <dubbo:registry address="zookeeper://10.211.55.5:2181"/>
20    <!--批量扫描-->
21    <dubbo:annotation package="com.rushuni.controller" />
22    <!--
23        超时全局设置 10分钟
24        check=false 不检查服务提供方，开发阶段建议设置为false
25        check=true 启动时检查服务提供方，如果服务提供方没有启动则报错
26    -->
27    <dubbo:consumer timeout="600000" check="false"/>
28    <!--文件上传组件-->
29    <bean id="multipartResolver"
30
class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
31        <property name="maxUploadSize" value="104857600" />
32        <property name="maxInMemorySize" value="4096" />
33        <property name="defaultEncoding" value="UTF-8"/>
34    </bean>
35
36    <mvc:resources mapping="/elementuidemo/**" location="/elementuidemo/" />
37    <!-- <mvc:resources mapping="/images/**" location="/images/" />-->
38    <!-- <mvc:resources mapping="/js/**" location="/js/" />-->
39
40 </beans>

```

ElementUI

介绍

介绍 ElementUI 是一套基于VUE2.0的桌面端组件库，ElementUI 提供了丰富的组件帮助开发人员快速构建功能强大、风格统一的页面。

官网地址：<http://element-cn.eleme.io/#/zh-CN>

如何使用

健康项目后台系统就是使用ElementUI来构建页面，在页面上引入 js 和 css 文件即可开始使用，如下：

HTML | 复制代码

```
1 <!-- 引入ElementUI样式 -->
2 <link rel="stylesheet" href="https://unpkg.com/element-ui/lib/theme-chalk/index.css">
3 <script src="https://unpkg.com/vue/dist/vue.js"></script>
4 <!-- 引入ElementUI组件库 -->
5 <script src="https://unpkg.com/element-ui/lib/index.js"></script>
```

常用组件

Container 布局容器

用于布局的容器组件，方便快速搭建页面的基本结构：

- `<el-container>`：外层容器。当子元素中包含 `<el-header>` 或 `<el-footer>` 时，全部子元素会垂直上下排列，否则会水平左右排列
- `<el-header>`：顶栏容器
- `<el-aside>`：侧边栏容器
- `<el-main>`：主要区域容器
- `<el-footer>`：底栏容器

```
1 <body>
2   <div id="app">
3     <el-container>
4       <el-header>Header</el-header>
5       <el-container>
6         <el-aside width="200px">Aside</el-aside>
7         <el-container>
8           <el-main>Main</el-main>
9           <el-footer>Footer</el-footer>
10        </el-container>
11      </el-container>
12    </el-container>
13  </div>
14  <style>
15    .el-header, .el-footer {
16      background-color: #B3C0D1;
17      color: #333;
18      text-align: left;
19      line-height: 60px;
20    }
21
22    .el-aside {
23      background-color: #D3DCE6;
24      color: #333;
25      text-align: center;
26      line-height: 200px;
27    }
28
29    .el-main {
30      background-color: #E9EEF3;
31      color: #333;
32      text-align: center;
33      line-height: 590px;
34    }
35  </style>
36 </body>
37 <script>
38   new Vue({
39     el: '#app'
40   });
41 </script>
```

Dropdown 下拉菜单

将动作或菜单折叠到下拉菜单中。

```
1 <el-dropdown split-button size="small" trigger="click">
2   个人中心
3   <el-dropdown-menu>
4     <el-dropdown-item>退出系统</el-dropdown-item>
5     <el-dropdown-item divided>修改密码</el-dropdown-item>
6     <el-dropdown-item divided>联系管理员</el-dropdown-item>
7   </el-dropdown-menu>
8 </el-dropdown>
```

NavMenu 导航菜单

为网站提供导航功能的菜单。

```
1 <el-menu>
2   <el-submenu index="1">
3     <template slot="title">
4       <i class="el-icon-location"></i>
5       <span slot="title">导航一</span>
6     </template>
7     <el-menu-item>选项1</el-menu-item>
8     <el-menu-item>选项2</el-menu-item>
9     <el-menu-item>选项3</el-menu-item>
10  </el-submenu>
11  <el-submenu index="2">
12    <template slot="title">
13      <i class="el-icon-menu"></i>
14      <span slot="title">导航二</span>
15    </template>
16    <el-menu-item>选项1</el-menu-item>
17    <el-menu-item>选项2</el-menu-item>
18    <el-menu-item>选项3</el-menu-item>
19  </el-submenu>
20 </el-menu>
```

Table 表格

用于展示多条结构类似的数据，可对数据进行排序、筛选、对比或其他自定义操作。

```


1 <el-table :data="tableData" stripe>
2   <el-table-column prop="date" label="日期"></el-table-column>
3   <el-table-column prop="name" label="姓名"></el-table-column>
4   <el-table-column prop="address" label="地址"></el-table-column>
5   <el-table-column label="操作" align="center">
6     <!--
7       slot-scope: 作用域插槽，可以获取表格数据
8       scope: 代表表格数据，可以通过scope.row来获取表格当前行数据，scope不是固定写法
9     -->
10    <template slot-scope="scope">
11      <el-button type="primary" size="mini" @click="handleUpdate(scope.row)">
编辑</el-button>
12      <el-button type="danger" size="mini" @click="handleDelete(scope.row)">
删除</el-button>
13    </template>
14  </el-table-column>
15 </el-table>
16 <script>
17   new Vue({
18     el: '#app',
19     data: {
20       tableData: [{
21         date: '2021-05-01',
22         name: '张一',
23         address: '广州市天河区 111 号'
24       }, {
25         date: '2021-05-02',
26         name: '张二',
27         address: '广州市天河区 112 号'
28       }, {
29         date: '2021-05-03',
30         name: '张三',
31         address: '广州市天河区 113 号'
32       }
33     ],
34     methods: {
35       handleUpdate(row) {
36         alert(row.date);
37       },
38       handleDelete(row) {
39         alert(row.date);
40       }
41     }
42   });
43 </script>

```

Pagination 分页

当数据量过多时，使用分页分解数据。

HTML

 复制代码

```
1  <!--
2    current-change: 内置的事件，当前页码改变时会触发，可以获取到改变之后的页码
3  -->
4  <el-pagination
5      @current-change="handleCurrentChange"
6      current-page="5"
7      page-size="10"
8      layout="total, prev, pager, next, jumper"
9      :total="305">
10 </el-pagination>
11 <script>
12   new Vue({
13     el: '#app',
14     methods: {
15       handleCurrentChange(page) {
16         alert(page);
17       }
18     }
19   });
20 </script>
```

Message 消息提示

常用于主动操作后的反馈提示。

```
1 <el-button :plain="true" @click="open1">消息</el-button>
2 <el-button :plain="true" @click="open2">成功</el-button>
3 <el-button :plain="true" @click="open3">警告</el-button>
4 <el-button :plain="true" @click="open4">错误</el-button>
5 <script>
6   new Vue({
7     el: '#app',
8     methods: {
9       open1() {
10         this.$message('这是一条消息提示');
11       },
12       open2() {
13         this.$message({
14           message: '恭喜你，这是一条成功消息',
15           type: 'success'
16         });
17       },
18       open3() {
19         this.$message({
20           message: '警告，这是一条警告消息',
21           type: 'warning'
22         });
23       },
24       open4() {
25         this.$message.error('错了，这是一条错误消息');
26       }
27     }
28   })
29 </script>
```

Tabs 标签页

分隔内容上有关联但属于不同类别的数据集合。

```
1 <h3>基础的、简洁的标签页</h3>
2 <!--
3     通过value属性来指定当前选中的标签页
4 -->
5 <el-tabs value="first">
6   <el-tab-pane label="用户管理" name="first">用户管理</el-tab-pane>
7   <el-tab-pane label="配置管理" name="second">配置管理</el-tab-pane>
8   <el-tab-pane label="角色管理" name="third">角色管理</el-tab-pane>
9   <el-tab-pane label="定时任务补偿" name="fourth">定时任务补偿</el-tab-pane>
10 </el-tabs>
11 <h3>选项卡样式的标签页</h3>
12 <el-tabs value="first" type="card">
13   <el-tab-pane label="用户管理" name="first">用户管理</el-tab-pane>
14   <el-tab-pane label="配置管理" name="second">配置管理</el-tab-pane>
15   <el-tab-pane label="角色管理" name="third">角色管理</el-tab-pane>
16   <el-tab-pane label="定时任务补偿" name="fourth">定时任务补偿</el-tab-pane>
17 </el-tabs>
18 <h3>卡片化的标签页</h3>
19 <el-tabs value="first" type="border-card">
20   <el-tab-pane label="用户管理" name="first">用户管理</el-tab-pane>
21   <el-tab-pane label="配置管理" name="second">配置管理</el-tab-pane>
22   <el-tab-pane label="角色管理" name="third">角色管理</el-tab-pane>
23   <el-tab-pane label="定时任务补偿" name="fourth">定时任务补偿</el-tab-pane>
24 </el-tabs>
25 <script>
26   new Vue({
27     el: '#app'
28   })
29 </script>
```

Form 表单

由输入框、选择器、单选框、多选框等控件组成，用以收集、校验、提交数据。在 Form 组件中，每一个表单域由一个 Form-Item 组件构成，表单域中可以放置各种类型的表单控件，包括 Input、Select、Checkbox、Radio、Switch、DatePicker、TimePicker。

```

1  <!--
2      rules: 表单验证规则
3  -->
4  <el-form ref="form" :model="form" :rules="rules" label-width="80px">
5      <!--
6          prop: 表单域 model 字段, 在使用 validate、resetFields 方法的情况下, 该属性是必填的
7      -->
8      <el-form-item label="活动名称" prop="name">
9          <el-input v-model="form.name"></el-input>
10     </el-form-item>
11     <el-form-item label="活动区域" prop="region">
12         <el-select v-model="form.region" placeholder="请选择活动区域">
13             <el-option label="区域一" value="shanghai"></el-option>
14             <el-option label="区域二" value="beijing"></el-option>
15         </el-select>
16     </el-form-item>
17     <el-form-item label="活动时间">
18         <el-col :span="11">
19             <el-date-picker type="date" placeholder="选择日期" v-model="form.date1"
20             style="width: 100%;"></el-date-picker>
21         </el-col>
22         <el-col class="line" :span="2">--</el-col>
23         <el-col :span="11">
24             <el-time-picker type="fixed-time" placeholder="选择时间" v-
25             model="form.date2" style="width: 100%;"></el-time-picker>
26         </el-col>
27     </el-form-item>
28     <el-form-item label="即时配送">
29         <el-switch v-model="form.delivery"></el-switch>
30     </el-form-item>
31     <el-form-item label="活动性质">
32         <el-checkbox-group v-model="form.type">
33             <el-checkbox label="美食/餐厅线上活动" name="type"></el-checkbox>
34             <el-checkbox label="地推活动" name="type"></el-checkbox>
35             <el-checkbox label="线下主题活动" name="type"></el-checkbox>
36             <el-checkbox label="单纯品牌曝光" name="type"></el-checkbox>
37         </el-checkbox-group>
38     </el-form-item>
39     <el-form-item label="特殊资源">
40         <el-radio-group v-model="form.resource">
41             <el-radio label="线上品牌商赞助"></el-radio>
42             <el-radio label="线下场地免费"></el-radio>
43         </el-radio-group>
44     </el-form-item>
45     <el-form-item label="活动形式">
46         <el-input type="textarea" v-model="form.desc"></el-input>
47     </el-form-item>
48 </el-form>

```



```

47     <el-button type="primary" @click="onSubmit">立即创建</el-button>
48   </el-form-item>
49 </el-form>
50 <script>
51   new Vue({
52     el: '#app',
53     data:{
54       form: {
55         name: '',
56         region: '',
57         date1: '',
58         date2: '',
59         delivery: false,
60         type: [],
61         resource: '',
62         desc: ''
63       },
64       //定义校验规则
65       rules: {
66         name: [
67           { required: true, message: '请输入活动名称', trigger: 'blur' },
68           { min: 3, max: 5, message: '长度在 3 到 5 个字符', trigger: 'blur' }
69         ],
70         region: [
71           { required: true, message: '请选择活动区域', trigger: 'change' }
72         ]
73       }
74     },
75     methods:{
76       onSubmit() {
77         console.log(this.form);
78         //validate: 对整个表单进行校验的方法，参数为一个回调函数。
79         //该回调函数会在校验结束后被调用，并传入两个参数：是否校验成功和未通过校验的字段。
80         this.$refs['form'].validate((valid) => {
81           if (valid) {
82             alert('submit!');
83           } else {
84             console.log('error submit!!');
85             return false;
86           }
87         });
88       }
89     }
90   })
91 </script>

```

课后作业

- 完成分布式锁，集群搭建
- 完成项目搭建
- 熟悉 element-ui