**二维码中间件设计**

# 系统目标及解决方案

## 系统目标

1、统一公司内部各个系统中物理世界对象的二维码内容结构；

2、精简二位码中的内容，提高二维码的辨识度；

3、提高二维码的稳定性与可扩展性，随着业务的发展，即使二维码内容发生变化，也无需重新生成二维码；

## 解决方案

将二维码与其内容分离，二位码中只记录一个key，额外用一张表记录二维码key与二维码内容的关系，这样当二维码内容有变化时，二维码的key不会变，已生成的二维码无需修改；

做一个二维码中间件，提供二维码key与二维码内容的记录与查询服务；

二维码：url?key=\*\*\*

# 系统设计

## 数据库表设计

|  |  |  |  |
| --- | --- | --- | --- |
| 二维码内容表：QRCode | | | |
| 字段 | 类型 | 描述 | 备注 |
| key | String | 主键ID | 二维码主键值 |
| objId | String | 对象id | 建筑，楼层，空间，系统，设备，虚拟设备（例如：信标）等 |
| proId | String | 所属项目id |  |
| create\_time | Timestamp | 创建时间 | yyyyMMddhhmmss |
| update\_time | DateTime | 更新时间 | yyyyMMddhhmmss |
| valid | Boolean | 有效状态 |  |
| fmId | String | 资产id | 数字化交付用 |
| pointId | String | 点位标签 | 数字化交付用 |
| 主键 | key |  |  |
| 索引 | valid，objId；valid，fmId；valid，proId； | | |

## 相关接口设计

### 添加二维码

http://localhost:8080/qrcode/restQRCodeService/addQRCode

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

"key":"\*\*\*", //二维码key , 必须（建议用无业务含议的UUID）

"objId":"\*\*\*", //对象id，非必须

"fmId":"\*\*\*", //资产id，非必须

"proId":"\*\*\*", //项目id ,必须

" pointId ":"\*\*\*", //点位标签 ,非必须

}

成功返回例子：

{

"Result": "success",

"ResultMsg": "",

}

失败返回例子：

{

"Result": "failure",

"ResultMsg": ""

}

接口说明：

1、key要求唯一，且不可重复，当重复时，不做添加操作，建议用无业务含议的UUID；

### 根据key查询二维码信息

http://localhost:8080/qrcode/restQRCodeService/queryQRCodeByKey

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

"key":"\*\*\*" //二维码key ,必须

}

返回例子：

{

"Result": "success",

"Item":{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

}

}

### 根据对象id查询二维码

http://localhost:8080/qrcode/restQRCodeService/queryQRCodeByObjIds

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

"objIds":["\*\*\*","\*\*\*"] //对象id数组，必须

}

成功返回格式：

{

"Result": "success",

"Content": [

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

},

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

}

],

"Count": 2,

}

### 根据字段查询二维码

http://localhost:8080/qrcode/restQRCodeService/queryQRCodeList

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

" proId ": "\*\*\*", //所属项目id，非必须

"objId":"\*\*\*", //对象id，非必须

"fmId":"\*\*\*", //资产id ，非必须

" pointId ":"\*\*\*", //点位标签，非必须

"valid":true //有效状态 ，true-有效，false-失效，非必须

}

成功返回格式：

{

"Result": "success",

"Content": [

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

},

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

}

],

"Count": 2,

}

接口说明：

1、返回结果按照创建时间倒序排序；

2、只支持字段值等于的查询，不支持范围查询；

3、查询条件中的字段支持以后的扩充；

### 增量查询二维码

http://localhost:8080/qrcode/restQRCodeService/queryQRCodeBytime

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

"from\_time":"20170620000000", //修改时间，yyyyMMddHHmmss，必须

" proId ": "\*\*\*", //所属项目id，非必须

"valid":true //有效状态 ，true-有效，false-失效，非必须

}

成功返回格式：

{

"Result": "success",

"Content": [

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

},

{

" key ":"\*\*\*", //二维码key

"objId":"\*\*\*", //对象id

"fmId":"\*\*\*", //资产id

"proId":"\*\*\*", //项目id

" pointId ":"\*\*\*", //点位标签

"create\_time":"20170620093000", //创建时间，yyyyMMddHHmmss

"update\_time":"20170620093000", //最后更新时间，yyyyMMddHHmmss

"valid":true //有效状态 ，true-有效，false-失效

}

],

"Count": 2,

}

接口说明：

1、返回结果按照修改时间正序排序；

2、只查询修改时间大于from\_time的数据；

### 修改二维码

http://localhost:8080/qrcode/restQRCodeService/updateQRCodeByKey

post请求

参数例子：

{

"user\_id":"\*\*\*", //账号id-当前登录人的账号id，必须

"key":"\*\*\*", //二维码key ,必须

"objId":"\*\*\*", //对象id，非必须

"fmId":"\*\*\*", //资产id，非必须

"proId":"\*\*\*", //项目id ,非必须

" pointId ":"\*\*\*", //点位标签 ,非必须

}

成功返回例子：

{

"Result": "success",

"ResultMsg": "",

}

失败返回例子：

{

"Result": "failure",

"ResultMsg": ""

}

# 博锐修改内容

### 生成二维码

修改生成二维码的方法，生成二维码同时，调用前面中间件的接口，添加二维码key和二维码内容的关系数据；

因以前的二维码内容与现状的格式不一致，建议调整之后，统一重新生成二维码；

### App端解析

App端，工单执行过程扫码后，根据二维码中的key，调用前面中间件的接口，查询二维码中的数据，主要有objId和proId；

### web解析

web解析，系统外部程序扫描二维码跳转到解析页后，根据二维码中的key，调用前面中间件的接口，查询二维码中的数据，主要有objId和proId，再根据上边两个参数查找对象相关信息；

# 问题

问题：

1. 决定采用什么数据库（HBase/mysql）存储该二维码内容数据；
2. 本系统不支持二维码内容的解析。