**Java操作mongodb简单介绍**

**MONGODB的官网：**[**http://www.mongodb.org/**](http://www.mongodb.org/)

**MONGODB的下载：**[**http://www.mongodb.org/download/**](http://www.mongodb.org/download/)

**MONGODB的java驱动：**[**https://github.com/mongodb/mongo-java-driver/downloads**](https://github.com/mongodb/mongo-java-driver/downloads)

**MONGODB针对java的api文档：**[**http://api.mongodb.org/java/2.6.2/**](http://api.mongodb.org/java/2.6.2/)

## Java与mongodb的连接

### 1.1连单台mongodb

Mongo mg = **new** Mongo();//默认连本机127.0.0.1 端口为27017

Mongo mg = **new** Mongo(ip);//可以指定ip 端口默认为27017

Mongo mg = **new** Mongo(ip,port);//也可以指定ip及端口号

### 1.2连双台mongodb

//ip为主机ip地址，port为端口号，dataBaseName相当于数据库名

DBAddress left = new DBAddress("ip:port/dataBaseName");

DBAddress right = new DBAddress("ip:port/dataBaseName ");

//若一个mongodb出现问题，会自动连另外一台

Mongo mongo = new Mongo(left, right);

### 1.3连多台mongodb

List<ServerAddress> mongoHostList = **new** ArrayList<ServerAddress>();

mongoHostList.add(**new** ServerAddress("ip",port));

mongoHostList.add(**new** ServerAddress("ip",port));

mongoHostList.add(**new** ServerAddress("ip",port));

Mongo mg = **new** Mongo(mongoHostList);

## Java获取mongodb的数据库名

### 2.1获取mongodb的db

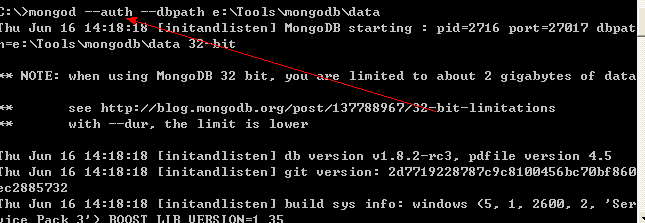
//dataBaseName相当于关系数据库里的数据库名，mongodb中若没有该数据库名也//不会报错，默认mongodb会建立这个数据库名，为空。

DB db = mg.getDB(dataBaseName);

**注意：mongodb区分大小写，程序中一定要注意**

### 2.2mongodb的db安全认证

//如果mongodb服务器运行时加了用户名及密码安全认证 如下图



若这个时候要想访问db中的collection（相当于关系数据库里的表），就必须通过安全认证才可以访问，否则后台会报您没有通过安全认证。

//安全认证java代码 返回true表示通过，false表示没通过不能进行操作

**boolean** auth = db.authenticate("userName", "password".toCharArray());

if(auth){……}

## 三．Java对mongodb的collection进行crud操作

### 3.1得到mongodb中的db的collection

//参数tableName相当于关系数据库里的表名,

//若mongodb中没有该tableName，默认会创建该tableName,为空

DBCollection users = db.getCollection(tableName);

### 3.2mongodb中的db的collection自增长主键

Mongodb中也像传统的关系数据库里表一样，有主键（\_id）概念，用来唯一标识他们。当用户往collection中插入一条新记录的时候，如果没有指定\_id属性，那么mongodb会自动生成一个ObjectId类型的值，保存为\_id的值。

\_id的值可以为任何类型，除了数组，在实际应用中，鼓励用户自己定义\_id值，但是要保证他的唯一性。

传统的数据库中，通常用一个递增的序列来提供主键，在Mongodb中用ObjectId来代替，我们可以通过如下的方法来得到主键。

**public** **class** Tools {

/\*\*

\* 实现mongodb主键自增长的功能

\* **@param** users

\* **@param** tableName

\* **@return**

\*/

**public** **static** **long** getNext(DBCollection users,String tableName){

**long** incId = 0;

**try** {

DBObject ret = users.findAndModify(

**new** BasicDBObject("\_id", tableName), **null**, **null**, **false**,

**new** BasicDBObject("$inc", **new** BasicDBObject("next", 1)),

**true**, **true**);

incId = Long.*valueOf*(ret.get("next").toString());

} **catch** (Exception e) {

e.printStackTrace();

}

**return** incId;

}

}

### 3.3java对collection进行插入操作

//第一种插入方式

DBObject data1 = **new** BasicDBObject();

data1.put("\_id", Tools.*getNext*(users, "webcall\_notice\_gx"));

data1.put("cust\_Id", "123456");

data1.put("cust\_Name", "");

data1.put("cust\_BranchNo", "12345");

data1.put("notice\_Id", 2);

data1.put("is\_Show", 1);

data1.put("start\_time", **new** Date());

data1.put("end\_time", **new** Date());

data1.put("province\_Id", "22");

data1.put("notice\_time\_number", 20110616);

users.insert(data1);//等同于users.save(data1);

//第二中插入方式

BasicDBObjectBuilder data1 = BasicDBObjectBuilder.*start*()

.add("\_id", Tools.*getNext*(users, "webcall\_notice\_gx"))

.add(… );

users.insert(data1.get());//等同于users.save(data1.get());

//第三种插入方式

Map<String, Object> data1 = **new** HashMap<String, Object>();

data1.put("\_id", Tools.*getNext*(users, "webcall\_notice\_gx"));

……

users.insert(**new** BasicDBObject(data1));

//等同于users.save(**new** BasicDBObject(data1)

);

//第四中插入方式

String json = "{'database' : 'mkyongDB','table' : 'hosting'," +

"'detail' : {'records' : 99, 'index' : 'vps\_index1', 'active' : 'true'}}}";

DBObject data1 = (DBObject)JSON.*parse*(json);

users.insert(data1);//等同于users.save(data1);

### 3.4java对collection的查询操作

//查询所有的数据 find() 注意他返回的是DBCursor

DBCursor cur = users.find();

**while**(cur.hasNext()){

System.*out*.println(cur.next());

}

Find()里面可以加很多参数，相当于关系数据库里where条件一样 如：

//查询 id的值为30000的记录 相当于where \_id=30000

DBCursor cur = users.find(**new** BasicDBObject("\_id",30000));

**while**(cur.hasNext()){

System.*out*.println(cur.next());

}

//查询id大于等于30000的记录 相当于where \_id>=30000

DBCursor cur = users.find(**new** BasicDBObject("\_id",**new** BasicDBObject("$gte",30000)));

**while**(cur.hasNext()){

System.*out*.println(cur.next());

}

//查询id大于等于1的记录，并且只取10条记录

DBCursor cur = users.find(**new** BasicDBObject("\_id",**new** BasicDBObject("$gte",1))).limit(10);

**while**(cur.hasNext()){

System.*out*.println(cur.next());

}

//查询id大于等于1的记录，并按id进行降序 -1表示降序，1升序。

DBCursor cur = users.find(new BasicDBObject("\_id",new BasicDBObject("$gte",1))).sort(new BasicDBObject("\_id",-1));

while(cur.hasNext()){

System.out.println(cur.next());

}

//查询id大于等于1的记录，并跳过前10条记录显示 相当于id>10

DBCursor cur = users.find(**new** BasicDBObject("\_id",**new** BasicDBObject("$gte",1))).skip(10);

**while**(cur.hasNext()){

System.*out*.println(cur.next());

}

//查询id大于等于1的记录，跳过前10条记录并且只显示10条记录。相当//于分页功能where id>10 and id<=20

DBCursor cur = users.find(new BasicDBObject("\_id",new BasicDBObject("$gte",1))).skip(10).limit(10);

while(cur.hasNext()){

System.out.println(cur.next());

}

//查询id大于等于1的所有记录数 返回int型

users.find(**new** BasicDBObject("\_id",**new** BasicDBObject("$gte",1))).count()

//只查询一条数据，多条取第一条 findOne(),他返回的是DBObject

users.findOne();//查询一条记录

//查询\_id为30000的一条记录

users.findOne(**new** BasicDBObject("\_id",30000));

//查询\_id为30000 返回cust\_Id的值 ，**mongodb查询都会返回主键的值**

System.*out*.println(users.findOne(**new** BasicDBObject("\_id",30000),**new** BasicDBObject("cust\_Id",**true**)));

//findAndRemove() 查询\_id=30000的数据，并且删除

users.findAndRemove(**new** BasicDBObject("\_id", 30000));

//findAndModify介绍

users.findAndModify(

**new** BasicDBObject("\_id", 28), //查询\_id=28的数据

**new** BasicDBObject("cust\_Id", **true**), //查询cust\_Id属性

**new** BasicDBObject("notice\_Id", **true**), //按照notice\_Id排序

**false**, //查询到的记录是否删除，true表示删除

**new** BasicDBObject("province\_Id", "100"), //将province\_id的值改为100

**true**, //是否返回新记录 true返回，false不返回

**true** //如果没有查询到该数据是否插collection true入库，false不入

));

### 3.5java对collection的更新操作

//查询id为300的记录，将cust\_Id的值更新为6533615，一定要注意大小写，以及数据//类型，返回值为int表示所影响的记录条数

//可以用users.findOne(**new** BasicDBObject("\_id",300));查看下，会发现这条记录//只返回两个字段了，分别为\_id,cust\_Id，别的字段都删除了。

users.update(**new** BasicDBObject("\_id",300), **new** BasicDBObject ("cust\_Id","6533615")).getN();

//这种写法可以实现只更新id为300的cust\_Id值为6533615，而不删除这条记录的别的字//段属性

users.update(**new** BasicDBObject("\_id",300), **new** BasicDBObject("$set",**new** BasicDBObject("cust\_Id","6533615"))).getN();

users.update(**new** BasicDBObject("cust\_Id","6533615"),

**new** BasicDBObject("$set",**new** BasicDBObject("cust\_Name","夏新火")),

**false**, //如果数据库不存在，是否添加

**true** //多条修改，false只修改一条记录

).getN());

### 3.6java对collection的删除操作

//移除cust\_Id为6533615的数据。注意 用remove方法不释放磁盘空间，

//mongodb只在collection中做了标志，没有正在删除。

users.remove(**new** BasicDBObject("cust\_Id","6533615")).getN();

//移除id>=1的数据

users.remove(new BasicDBObject("\_id",new BasicDBObject("$gte",1))).getN();

//移除整个collection，drop也不释放磁盘空间

users.drop();