Spring Data MongoDB : Update document

By mkyong (http://www.mkyong.com/author/mkyong/) | May 18, 2011 | Updated : April 7, 2013 | Viewed : 96,332 times +410 pv/w

In Spring data – MongoDB, you can use following methods to update documents.

1. save – Update the whole object, if "_id" is present, perform an update, else insert it.
2. updateFirst – Updates the first document that matches the query.
3. updateMulti – Updates all documents that match the query.
4. Upserting – If no document that matches the query, a new document is created by combining the query and update object.
5. findAndModify – Same with updateMulti, but it has an extra option to return either the old or newly updated document.

P.S All examples are tested under `mongo-java-driver-2.11.0.jar` *and* `spring-data-mongodb-1.2.0.RELEASE.jar`

1. saveOrUpdate – part 1 example

Assume below json data is inserted into MongoDB.

```
{
    "_id" : ObjectId("id"),
    "ic" : "1001",
    "name" : "appleA",
    "age" : 20,
    "createdDate" : ISODate("2013-04-06T23:17:35.530Z")
}
```

Find the document, modify and update it with `save()` method.

```
Query query = new Query();
query.addCriteria(Criteria.where("name").is("appleA"));

User userTest1 = mongoOperation.findOne(query, User.class);

System.out.println("userTest1 - " + userTest1);

//modify and update with save()
userTest1.setAge(99);
mongoOperation.save(userTest1);

//get the updated object again
User userTest1_1 = mongoOperation.findOne(query, User.class);

System.out.println("userTest1_1 - " + userTest1_1);
```

Output

```
userTest1 - User [id=id, ic=1001, name=appleA, age=20, createdDate=Sat Apr 06 23:17:35 MYT 2013]
userTest1_1 - User [id=id, ic=1001, name=appleA, age=99, createdDate=Sat Apr 06 23:17:35 MYT 2013]
```

Note
See example 2, it shows a common mistake made by most of the developers.

2. saveOrUpdate – part 2 example

This is a failed example, read carefully, a really common mistake.

Assume below json data is inserted into MongoDB.

```
{
    "_id" : ObjectId("id"),
    "ic" : "1002",
    "name" : "appleB",
    "age" : 20,
    "createdDate" : ISODate("2013-04-06T15:22:34.530Z")
}
```

In `Query`, you get the document returned with a single “name” field value only, it did happened often to save the object returned size. The returned “User” object has null value in the fields : age, ic and createdDate, if you modify the ‘age’ field and update it, it will override everything instead of update the modified field – ‘age’.

```
Query query = new Query();
query.addCriteria(Criteria.where("name").is("appleB"));
query.fields().include("name");

User userTest2 = mongoOperation.findOne(query, User.class);
System.out.println("userTest2 - " + userTest2);

userTest2.setAge(99);

mongoOperation.save(userTest2);

// ooppss, you just override everything, it caused ic=null and
// createdDate=null

Query query1 = new Query();
query1.addCriteria(Criteria.where("name").is("appleB"));

User userTest2_1 = mongoOperation.findOne(query1, User.class);
System.out.println("userTest2_1 - " + userTest2_1);
```

Output

```
userTest2 - User [id=51603dba3004d7fffc202391, ic=null, name=appleB, age=0, createdDate=null]
userTest2_1 - User [id=51603dba3004d7fffc202391, ic=null, name=appleB, age=99, createdDate=null]
```

After the save(), the field ‘age’ is updated correctly, but ic and createdDate are both set to null, the entire “user” object is updated. To update a single field / key value, don’t use save(), use updateFirst() or updateMulti() instead.

3. updateFirst example

Updates the first document that matches the query. In this case, only the single field “age” is updated.

```
{
    "_id" : ObjectId("id"),
    "ic" : "1003",
    "name" : "appleC",
    "age" : 20,
    "createdDate" : ISODate("2013-04-06T23:22:34.530Z")
}
```

```
//returns only 'name' field
Query query = new Query();
query.addCriteria(Criteria.where("name").is("appleC"));
query.fields().include("name");

User userTest3 = mongoOperation.findOne(query, User.class);
System.out.println("userTest3 - " + userTest3);

Update update = new Update();
update.set("age", 100);

mongoOperation.updateFirst(query, update, User.class);

//returns everything
Query query1 = new Query();
query1.addCriteria(Criteria.where("name").is("appleC"));

User userTest3_1 = mongoOperation.findOne(query1, User.class);
System.out.println("userTest3_1 - " + userTest3_1);
```

Output

```
userTest3 - User [id=id, ic=null, name=appleC, age=0, createdDate=null]
userTest3_1 - User [id=id, ic=1003, name=appleC, age=100, createdDate=Sat Apr 06 23:22:34 MYT 2013]
```

4. updateMulti example

Updates all documents that matches the query.

```
{
    "_id" : ObjectId("id"),
    "ic" : "1004",
    "name" : "appleD",
    "age" : 20,
    "createdDate" : ISODate("2013-04-06T15:22:34.530Z")
}
{
    "_id" : ObjectId("id"),
    "ic" : "1005",
    "name" : "appleE",
    "age" : 20,
    "createdDate" : ISODate("2013-04-06T15:22:34.530Z")
}
```

```
//show the use of $or operator
Query query = new Query();
query.addCriteria(Criteria
        .where("name").exists(true)
        .orOperator(Criteria.where("name").is("appleD"),
                Criteria.where("name").is("appleE")));
Update update = new Update();

//update age to 11
update.set("age", 11);

//remove the createdDate field
update.unset("createdDate");

// if use updateFirst, it will update 1004 only.
// mongoOperation.updateFirst(query4, update4, User.class);

// update all matched, both 1004 and 1005
mongoOperation.updateMulti(query, update, User.class);

System.out.println(query.toString());

List<User> usersTest4 = mongoOperation.find(query4, User.class);

for (User userTest4 : usersTest4) {
    System.out.println("userTest4 - " + userTest4);
}
```

Output

```
Query: { "name" : { "$exists" : true} ,
    "$or" : [ { "name" : "appleD"} , { "name" : "appleE"}]}, Fields: null, Sort: null

userTest4 - User [id=id, ic=1004, name=appleD, age=11, createdDate=null]
userTest4 - User [id=id, ic=1005, name=appleE, age=11, createdDate=null]
```

5. Upsert example

If document is matched, update it, else create a new document by combining the query and update object, it's works like `findAndModifyElseCreate()`:)

```
{
    //no data
}
```

```
//search a document that doesn't exist
Query query = new Query();
query.addCriteria(Criteria.where("name").is("appleZ"));

Update update = new Update();
update.set("age", 21);

mongoOperation.upsert(query, update, User.class);

User userTest5 = mongoOperation.findOne(query, User.class);
System.out.println("userTest5 - " + userTest5);
```

Output, a new document is created by combining both query and update object.

```
userTest5 - User [id=id, ic=null, name=appleZ, age=21, createdDate=null]
```

6. findAndModify example

Find and modify and get the newly updated object from a single operation.

```
{
    "_id" : ObjectId("id"),
    "ic" : "1006",
    "name" : "appleF",
    "age" : 20,
    "createdDate" : ISODate("2013-04-07T13:11:34.530Z")
}
```

```
Query query6 = new Query();
query6.addCriteria(Criteria.where("name").is("appleF"));

Update update6 = new Update();
update6.set("age", 101);
update6.set("ic", 1111);

//FindAndModifyOptions().returnNew(true) = newly updated document
//FindAndModifyOptions().returnNew(false) = old document (not update yet)
User userTest6 = mongoOperation.findAndModify(
        query6, update6,
        new FindAndModifyOptions().returnNew(true), User.class);
System.out.println("userTest6 - " + userTest6);
```

Output

```
userTest6 - User [id=id, ic=1111, name=appleF, age=101, createdDate=Sun Apr 07 13:11:34 MYT 2013]
```

7. Full example

Full application to combine everything from example 1 to 6.

```
package com.mkyong.core;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.data.mongodb.core.FindAndModifyOptions;
import org.springframework.data.mongodb.core.MongoOperations;
import org.springframework.data.mongodb.core.query.Criteria;
import org.springframework.data.mongodb.core.query.Query;
import org.springframework.data.mongodb.core.query.Update;

import com.mkyong.config.SpringMongoConfig;
import com.mkyong.model.User;

public class UpdateApp {

    public static void main(String[] args) {
        // For Annotation
        ApplicationContext ctx =
            new AnnotationConfigApplicationContext(SpringMongoConfig.class);
        MongoOperations mongoOperation =
            (MongoOperations) ctx.getBean("mongoTemplate");

        // insert 6 users for testing
        List<User> users = new ArrayList<User>();

        User user1 = new User("1001", "appleA", 20, new Date());
        User user2 = new User("1002", "appleB", 20, new Date());
        User user3 = new User("1003", "appleC", 20, new Date());
        User user4 = new User("1004", "appleD", 20, new Date());
        User user5 = new User("1005", "appleE", 20, new Date());
        User user6 = new User("1006", "appleF", 20, new Date());
        users.add(user1);
        users.add(user2);
        users.add(user3);
        users.add(user4);
        users.add(user5);
        users.add(user6);
        mongoOperation.insert(users, User.class);

        // Case 1 ... find and update
        System.out.println("Case 1");

        Query query1 = new Query();
        query1.addCriteria(Criteria.where("name").is("appleA"));

        User userTest1 = mongoOperation.findOne(query1, User.class);

        System.out.println("userTest1 - " + userTest1);

        userTest1.setAge(99);
        mongoOperation.save(userTest1);

        User userTest1_1 = mongoOperation.findOne(query1, User.class);

        System.out.println("userTest1_1 - " + userTest1_1);

        // Case 2 ... select single field only
        System.out.println("\nCase 2");

        Query query2 = new Query();
        query2.addCriteria(Criteria.where("name").is("appleB"));
        query2.fields().include("name");

        User userTest2 = mongoOperation.findOne(query2, User.class);
        System.out.println("userTest2 - " + userTest2);

        userTest2.setAge(99);

        mongoOperation.save(userTest2);

        // ooppss, you just override everything, it caused ic=null and
        // createdDate=null
```

```
Query query2_1 = new Query();
query2_1.addCriteria(Criteria.where("name").is("appleB"));

User userTest2_1 = mongoOperation.findOne(query2_1, User.class);
System.out.println("userTest2_1 - " + userTest2_1);

System.out.println("\nCase 3");
Query query3 = new Query();
query3.addCriteria(Criteria.where("name").is("appleC"));
query3.fields().include("name");

User userTest3 = mongoOperation.findOne(query3, User.class);
System.out.println("userTest3 - " + userTest3);

Update update3 = new Update();
update3.set("age", 100);

mongoOperation.updateFirst(query3, update3, User.class);

Query query3_1 = new Query();
query3_1.addCriteria(Criteria.where("name").is("appleC"));

User userTest3_1 = mongoOperation.findOne(query3_1, User.class);
System.out.println("userTest3_1 - " + userTest3_1);

System.out.println("\nCase 4");
Query query4 = new Query();
query4.addCriteria(Criteria
        .where("name")
        .exists(true)
        .orOperator(Criteria.where("name").is("appleD"),
                Criteria.where("name").is("appleE"))));
Update update4 = new Update();
update4.set("age", 11);
update4.unset("createdDate");

// update 1004 only.
// mongoOperation.updateFirst(query4, update4, User.class);

// update all matched
mongoOperation.updateMulti(query4, update4, User.class);

System.out.println(query4.toString());

List<User> usersTest4 = mongoOperation.find(query4, User.class);

for (User userTest4 : usersTest4) {
    System.out.println("userTest4 - " + userTest4);
}

System.out.println("\nCase 5");
Query query5 = new Query();
query5.addCriteria(Criteria.where("name").is("appleZ"));

Update update5 = new Update();
update5.set("age", 21);

mongoOperation.upsert(query5, update5, User.class);

User userTest5 = mongoOperation.findOne(query5, User.class);
System.out.println("userTest5 - " + userTest5);

System.out.println("\nCase 6");
Query query6 = new Query();
query6.addCriteria(Criteria.where("name").is("appleF"));

Update update6 = new Update();
update6.set("age", 101);
update6.set("ic", 1111);

User userTest6 = mongoOperation.findAndModify(query6, update6,
        new FindAndModifyOptions().returnNew(true), User.class);
System.out.println("userTest6 - " + userTest6);

mongoOperation.dropCollection(User.class);
```

```
    }

}
```

Output

```
Case 1
userTest1 - User [id=id, ic=1001, name=appleA, age=20, createdDate=Sun Apr 07 13:22:48 MYT 2013]
userTest1_1 - User [id=id, ic=1001, name=appleA, age=99, createdDate=Sun Apr 07 13:22:48 MYT 2013]

Case 2
userTest2 - User [id=id, ic=null, name=appleB, age=0, createdDate=null]
userTest2_1 - User [id=id, ic=null, name=appleB, age=99, createdDate=null]

Case 3
userTest3 - User [id=id, ic=null, name=appleC, age=0, createdDate=null]
userTest3_1 - User [id=id, ic=1003, name=appleC, age=100, createdDate=Sun Apr 07 13:22:48 MYT 2013]

Case 4
Query: { "name" : { "$exists" : true} , "$or" : [ { "name" : "appleD"} , { "name" : "appleE"}]}, Fields: null, Sort: null
userTest4 - User [id=id, ic=1004, name=appleD, age=11, createdDate=null]
userTest4 - User [id=id, ic=1005, name=appleE, age=11, createdDate=null]

Case 5
userTest5 - User [id=id, ic=null, name=appleZ, age=21, createdDate=null]

Case 6
userTest6 - User [id=id, ic=1006, name=appleF, age=20, createdDate=Sun Apr 07 13:22:48 MYT 2013]
```

Download Source Code

Download it – SpringMongoDB-Update-Example.zip (http://www.mkyong.com/wp-content/uploads/2011/05/SpringMongoDB-Update-Example.zip) (29 KB)

References

1. MongoDB template update documentation (http://static.springsource.org/spring-data/data-document/docs/current/reference/html/#mongodb-template-update)
2. Spring Data MongoDB – Saving, Updating, and Removing Documents (http://static.springsource.org/spring-data/mongodb/docs/current/reference/html/mongo.core.html#mongo-template.save-update-remove)
3. Java MongoDB update example/ (http://www.mkyong.com/mongodb/java-mongodb-update-document/)
4. MongoDB update modifier operations (http://docs.mongodb.org/manual/core/update/)
5. Spring Data MongoDB Hello World Example (http://www.mkyong.com/mongodb/spring-data-mongodb-hello-world-example/)

Tags : mongodb (http://www.mkyong.com/tag/mongodb/) spring-data (http://www.mkyong.com/tag/spring-data/) update (http://www.mkyong.com/tag/update/)

Share this article on

About the Author

mkyong

Founder of Mkyong.com (http://mkyong.com), love Java and open source stuff. Follow him on Twitter (https://twitter.com/mkyong), or befriend him on Facebook (http://www.facebook.com/java.tutorial) or Google Plus (https://plus.google.com/110948163568945735692?rel=author). If you like my tutorials, consider make a donation to these charities (http://www.mkyong.com/blog/donate-to-charity/).

Related Posts

76k Spring Data MongoDB : Insert document (/mongodb/spring-data-mongodb-insert-document/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=0)

337k Spring Data MongoDB hello world example (/mongodb/spring-data-mongodb-hello-world-example/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=1)

47k Spring Data MongoDB : Save binary file, GridFS example (/mongodb/spring-data-mongodb-save-binary-file-gridfs-example/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=2)

24k Spring Data MongoDB remove _class column (/mongodb/spring-data-mongodb-remove-_class-column/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=3)

38k Spring Data MongoDB : get last modified records (date sorting) (/mongodb/spring-data-mongodb-get-last-modified-records-date-sorting/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=4)

3k MongoDB - Update to upper case (/mongodb/mongodb-update-to-upper-case/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=5)

25k Spring Data MongoDB - Select fields to return (/mongodb/spring-data-mongodb-select-fields-to-return/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=6)

58k Spring Data MongoDB - Aggregation Grouping Example (/mongodb/spring-data-mongodb-aggregation-grouping-example/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=7)

63k Spring Boot + Spring Data MongoDB example (/spring-boot/spring-boot-spring-data-mongodb-example/?utm_source=mkyong&utm_medium=author&utm_campaign=related-post&utm_content=8)

Popular Posts

764k
Hibernate – One-to-Many example (Annotation) (/hibernate/hibernate-one-to-many-relationship-example-annotation/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=1)

536k
10 Java Regular Expression Examples You Should Know (/regular-expressions/10-java-regular-expression-examples-you-should-know/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=2)

584k
Spring Security form login using database (/spring-security/spring-security-form-login-using-database/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=3)

966k
Maven Tutorial (/tutorials/maven-tutorials/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=4)

632k
How to write to file in Java - FileOutputStream (/java/how-to-write-to-file-in-java-fileoutputstream-example/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=5)

633k
Android WebView example (/android/android-webview-example/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=6)

956k
log4j.properties example (/logging/log4j-log4j-properties-examples/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=7)

632k
Apache HttpClient Examples (/java/apache-httpclient-examples/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=8)

683k

Spring MVC form handling example (/spring-mvc/spring-mvc-form-handling-example/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=9)

549k

ClassNotFoundException : org.springframework.web.context.ContextLoaderListener (/spring/spring-error-classnotfoundexception-org-springframework-web-context-contextloaderlistener/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=10)

906k

How to create a Web Application Project with Maven (/maven/how-to-create-a-web-application-project-with-maven/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=11)

561k

Hibernate Criteria examples (/hibernate/hibernate-criteria-examples/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=12)

1.2m

Spring Security Tutorial (/tutorials/spring-security-tutorials/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=13)

501k

Hibernate - fetching strategies examples (/hibernate/hibernate-fetching-strategies-examples/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=14)

1.1m

JAX-WS Tutorial (/tutorials/jax-ws-tutorials/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=15)

758k

How to read XML file in Java - (SAX Parser) (/java/how-to-read-xml-file-in-java-sax-parser/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=16)

1.1m

How to read and parse CSV file in Java (/java/how-to-read-and-parse-csv-file-in-java/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=17)

2.5m

Java - How to get current date time (/java/java-how-to-get-current-date-time-date-and-calender/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=18)

1.2m

Android spinner (drop down list) example (/android/android-spinner-drop-down-list-example/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=19)

525k

Spring AOP + AspectJ annotation example (/spring3/spring-aop-aspectj-annotation-example/?utm_source=mkyong&utm_medium=author&utm_campaign=top-pv&utm_content=20)

Leave a Reply

Join the discussion

Sort by: newest | oldest | most voted

Carol

very complete

Guest

0 REPLY 1 year 6 months ago

hihi

findAndModify only update 1 document.

Guest

0 REPLY 2 years 4 months ago

Hemant Sharma

Hello, your information is very helpful, I have a query, How to update data from any inner object like:

Guest

{
“student1” : {
“name” : “student_1_name”,
“fees” : {
“01” : {
“ruppees” : “1234”,
“date” : “need to update”
},
“02” : {

“ruppees” : “1234”,
“date” : “need to update”
},
“03” : {
“ruppees” : “1234”,
“date” : “need to update”
},
}
}
}

How to update date?

0 REPLY 2 years 10 months ago

Guest

lsoares13

how to update a field of type date ($date)?

0 REPLY 3 years 5 months ago

Guest

Dipankar Dutta

save method is not working for update.
Its inserting new row.
My Id coming as 2b238579-5ebf-49da-9228-a9dee828895f
but when save method fire then new row inserted
with id 2b238579-5ebf-49da-9228-a9dee828895f,2b238579-5ebf-49da-9228-a9dee828895f.
If I go to update again then new row id becomes
2b238579-5ebf-49da-9228-a9dee828895f,2b238579-5ebf-49da-9228-a9dee828895f,2b238579-5ebf-49da-9228-a9dee828895f

0 REPLY 4 years 3 months ago

Guest

edward

how to update the embedded collections?

0 REPLY 4 years 8 months ago