Installation and Configuration

- Setup
- Configuration File
- Security
- Roadmap

API

- Tutorial
- Resource URIs
- Representation Format
- Quick Rererence
- Query Documents
- Write Requests
- Files Bucket (GridFS)
- Collection Indexes
- ETag
- Shard Keys

Advanced Topics

- Aggregations
- Request Transformers
- Request Checkers
- Document validation with JSON schema
- Application Logic
- Initializer
- Request Hooks
- Relationships
- How to package custom code

Misc

- CORS Support
- Metrics on Requests

Tutorial

- Introduction
- Start MongoDB and RESTHeart
- Create a Database
- Get the database
- Create a Collection
- Get the collection
- Create two Documents
  - First document
  - Second document
- Get all Documents from the Collection
- GET Document by URL (by id)
- Query documents by properties
- Clean-up
- Additional resources

Introduction

In this tutorial we use RESTHeart to create a **database**, a **collection** and a couple of **documents** in MongoDB. Then we show how to perform simple queries. Here we use **Docker** because it’s the easiest way to run RESTHeart and MongoDB together.

> We use httpie, a command line HTTP client (curl is fine, but httpie is easier to use and produces a colorized and formatted output that’s easier to read). For a primer read httpie: A CLI http client that will make you smile.

Start MongoDB and RESTHeart

If you have Docker properly installed in your machine, you can start RESTHeart in seconds:

```
$ mkdir restheart && cd restheart
$ curl
https://raw.githubusercontent.com/SoftInstigate/restheart/master/docker-
compose.yml --output docker-compose.yml
$ docker-compose up -d
```

That runs RESTHeart and an empty MongoDB container, which we are going to use for the rest of this tutorial.

Optionally, you might want to review to the setup section to get more details on how to run RESTHeart and MongoDB with Docker and Docker Compose, but you can leave it for later. Let’s instead jump into some action.

Create a Database

Now that RESTHeart is up and running and connected to its empty MongoDB instance (just created for you by Docker Compose), the first step is to create a new MongoDB database:

```
$ http -a 'admin:changeit' PUT localhost:8080/db desc='this is my
first db created with restheart'

HTTP/1.1 201 Created
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:09:29.385Z
Connection: keep-alive
Content-Length: 0
Content-Type: application/json
Date: Thu, 26 Apr 2018 13:54:29 GMT
ETag: 5ae1da15a7b11b0005a3c41d
X-Powered-By: restheart.org
```

Get the database

To get the created database:

```
$ http -a 'admin:changeit' GET localhost:8080/db

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:19:07.744Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 149
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:04:07 GMT
ETag: 5ae1da15a7b11b0005a3c41d
X-Powered-By: restheart.org
```

```
{
    "_embedded": [],
    "_etag": {
        "$oid": "5ae1da15a7b11b0005a3c41d"
    },
    "_id": "db",
    "_returned": 0,
    "_size": 0,
    "_total_pages": 0,
    "desc": "this is my first db created with restheart"
}
```

Create a Collection

Now it’s possible to create a collection “coll” in the database “db”:

```
$ http -a 'admin:changeit' PUT localhost:8080/db/coll desc='my first
collection created with restheart'

HTTP/1.1 201 Created
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:25:39.914Z
Connection: keep-alive
Content-Length: 0
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:10:39 GMT
ETag: 5ae1dddfa7b11b0005a3c41f
X-Powered-By: restheart.org
```

Get the collection

To get the created collection:

```
$ http -a 'admin:changeit' GET 127.0.0.1:8080/db/coll

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:26:43.480Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 135
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:11:43 GMT
ETag: 5ae1dddfa7b11b0005a3c41f
X-Powered-By: restheart.org
```

```
{
    "_embedded": [],
    "_etag": {
        "$oid": "5ae1dddfa7b11b0005a3c41f"
    },
    "_id": "coll",
    "_returned": 0,
    "desc": "my first collection created with restheart"
}
```

Create two Documents

Let’s create some documents in MongoDB.

First document

```
$ http -a 'admin:changeit' POST localhost:8080/db/coll
name='RESTHeart' rating='cool'

HTTP/1.1 201 Created
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:28:06.435Z
Connection: keep-alive
Content-Length: 0
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:13:06 GMT
ETag: 5ae1de72a7b11b0005a3c420
Location: http://localhost:8080/db/coll/5ae1de72586f80fc867131f4
X-Powered-By: restheart.org
```

Note the `Location` header, as it contains a link to the newly created document! To get the document you can directly copy that link and use it in a subsequent query, like this:

```
$ http -a 'admin:changeit' GET
http://localhost:8080/db/coll/5ae1de72586f80fc867131f4

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:38:48.212Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 119
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:23:48 GMT
ETag: 5ae1de72a7b11b0005a3c420
X-Powered-By: restheart.org
```

```
{
    "_etag": {
        "$oid": "5ae1de72a7b11b0005a3c420"
    },
    "_id": {
        "$oid": "5ae1de72586f80fc867131f4"
    },
    "name": "RESTHeart",
    "rating": "cool"
}
```

Second document

Cool, now let’s create a second document:

```
$ http -a 'admin:changeit' POST localhost:8080/db/coll name='MongoDB'
rating='super cool'

HTTP/1.1 201 Created
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:29:24.215Z
Connection: keep-alive
Content-Length: 0
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:14:24 GMT
ETag: 5ae1dec0a7b11b0005a3c421
Location: http://localhost:8080/db/coll/5ae1dec0586f80fc86713200
X-Powered-By: restheart.org
```

As before, note the `Location` header. As before, you can GET the newly created document by requesting that link.

Get all Documents from the Collection

Now let’s get all documents in a row. For this, we send a GET request to the whole collection (named “coll” in this example).

```
$ http -a 'admin:changeit' GET localhost:8080/db/coll

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:30:18.109Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 223
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:15:18 GMT
ETag: 5ae1dddfa7b11b0005a3c41f
X-Powered-By: restheart.org
```

```
{
    "_embedded": [
        {
            "_etag": {
                "$oid": "5ae1dec0a7b11b0005a3c421"
            },
            "_id": {
                "$oid": "5ae1dec0586f80fc86713200"
            },
            "name": "MongoDB",
            "rating": "super cool"
        },
        {
            "_etag": {
                "$oid": "5ae1de72a7b11b0005a3c420"
            },
            "_id": {
                "$oid": "5ae1de72586f80fc867131f4"
            },
            "name": "RESTHeart",
            "rating": "cool"
        }
    ],
    "_etag": {
        "$oid": "5ae1dddfa7b11b0005a3c41f"
    },
    "_id": "coll",
    "_returned": 2,
    "desc": "my first collection created with restheart"
}
```

Note that the two documents are within the `_embedded` array, while the rest is metadata (the meaning of metadata, such as `_etag` and `_returned`, should appear self-explanatory).

Also beware that RESTHeart applies a pagination algorithm to all requests, but by default it works only if the collection contains more than 100 documents, which is not this case.

GET Document by URL (by id)

If you look into the `_embedded` array and copy the `_id` element, then it's possible to get that single document directly, exactly as we did before by looking at the `Location` header

For example, let's say the `_id` of a document in the `_embedded` array is "5ae1de72586f80fc867131f4" then we can GET it immediately:

```
$ http -a 'admin:changeit' GET
localhost:8080/db/coll/5ae1de72586f80fc867131f4

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:35:42.549Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 119
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:20:42 GMT
ETag: 5ae1de72a7b11b0005a3c420
X-Powered-By: restheart.org
```

```
{
    "_etag": {
        "$oid": "5ae1de72a7b11b0005a3c420"
    },
    "_id": {
        "$oid": "5ae1de72586f80fc867131f4"
    },
    "name": "RESTHeart",
    "rating": "cool"
}
```

Query documents by properties

Using directly the document `_id` is of course not the only available option. We can actually leverage MongoDB queries with the `filter` query parameter in HTTP calls:

```
$ http -a 'admin:changeit' GET localhost:8080/db/coll?filter="
{'name':'MongoDB'}"

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Location, ETag, Auth-Token, Auth-Token-
Valid-Until, Auth-Token-Location, X-Powered-By
Auth-Token: 65iz1g89ao5r2e47whohfx6ffw6vfl6nf6d44nyxez0ri7yzzh
Auth-Token-Location: /_authtokens/admin
Auth-Token-Valid-Until: 2018-04-26T14:44:29.115Z
Connection: keep-alive
Content-Encoding: gzip
Content-Length: 192
Content-Type: application/json
Date: Thu, 26 Apr 2018 14:29:29 GMT
ETag: 5ae1dddfa7b11b0005a3c41f
X-Powered-By: restheart.org
```

```
{
    "_embedded": [
        {
            "_etag": {
                "$oid": "5ae1dec0a7b11b0005a3c421"
            },
            "_id": {
                "$oid": "5ae1dec0586f80fc86713200"
            },
            "name": "MongoDB",
            "rating": "super cool"
        }
    ],
    "_etag": {
        "$oid": "5ae1dddfa7b11b0005a3c41f"
    },
    "_id": "coll",
    "_returned": 1,
    "desc": "my first collection created with restheart"
}
```

Now you could jump to Queries for more complex examples on how to search documents. Remember that you will have all MongoDB’s queries power at your disposal.

Clean-up

To stop all containers and clean-up everything, just run the `docker-compose down -v` command:

```
$ docker-compose down -v

Stopping restheart       ... done
Stopping restheart-mongo ... done
Removing restheart       ... done
Removing restheart-mongo ... done
Removing network restheart_backend
Removing volume restheart_restheart-datadb
```

Additional resources

- Setup
- Resource URI
- Resource Representation Format
- Queries