Crypting Api Documentation

Toci

August 19, 2015

Contents

1 Indtroduction

Resource: /api/models/

Resource destination: This resource allows to Load/Save/Update and Delete encrypted records in the databae, using a password. The entities are stored encrypted in the database, so only it can be only read using the right password.

Error Handling: The error handling is unified in the entire API, and the error list is created bitwise. We use following error codes:

- 1 - IdMissing
- 2 - WrongId
- 4 - DataMissing
- 8 - PasswordMissing
- 16 - NameMissing

Error message example:

- Password missing, please provide a/an Password in Your request.

2 Retrieve the data for resource operation.

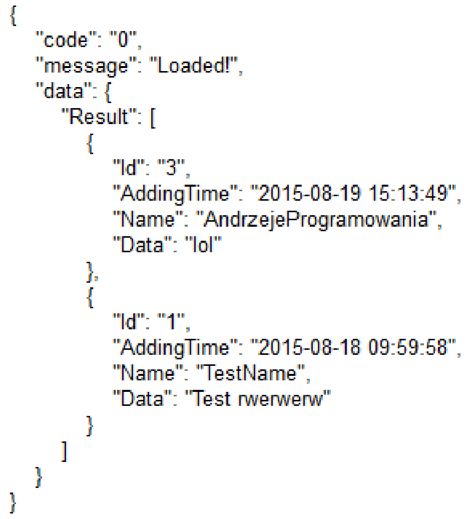
Example usage:
POST /api/models/load

Request body:

- password - string type, required, it is the password necessary to decode the entities, only the records encrypted with the given password will be returned.

Response formats: JSON

Example JSON Response structure:



```
{
   "code": "0",
   "message": "Loaded!",
   "data": {
      "Result": [
         {
            "Id": "3",
            "AddingTime": "2015-08-19 15:13:49",
            "Name": "AndrzejeProgramowania",
            "Data": "lol"
         },
         {
            "Id": "1",
            "AddingTime": "2015-08-18 09:59:58",
            "Name": "TestName",
            "Data": "Test rwerwerw"
         }
      ]
   }
}
```

Response structure description:

- code - the error code specified in the ”Error Handling” section
- message - response message from the api
- data/result - the requested content
  - id - entity id in the database
  - addingTime - it is the time, when the record was added
  - name - the name/nick off the adding person
  - data - the text data that is encrypted

3 Save the data for resource operation.

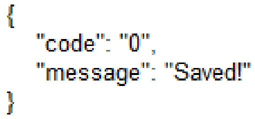
Example usage:
POST /api/models/save

Request body:

- password - string type, required, it is the password necessary to encode/decode the entities.
- data - the data that should be encrypted
- name - the name of the adding user

Response formats: JSON

Example JSON Response structure:



```
{
   "code": "0",
   "message": "Saved!"
}
```

Response message description:

- code - error code
- message - the operation result message

4 Update the data for resource operation.

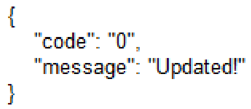
Example usage:
POST /api/models/update

Request body:

- password - string type, required, it is the password necessary to decode the entities.
- data - the data that should be encrypted
- name - the name of the adding user
- addingTime - the adding time
- id - the id of the entity

Response formats: JSON

Example JSON Response structure:



```
{
    "code": "0",
    "message": "Updated!"
}
```

Response message description:

- code - error code
- message - the operation result message

5 Delete the data for resource operation.

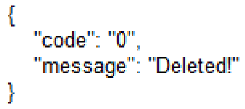
Example usage:
POST /api/models/delete

Request body:

- id - the id of the entity

Response formats: JSON

Example JSON Response structure:



```
{
    "code": "0",
    "message": "Deleted!"
}
```

Response message description:

- code - error code
- message - the operation result message