MY API Reference

This API reference describes resources created in a **Spring Boot** Web services application running locally. It is a REST API with simple resources to use as the basis to create sample API Reference Documentation.

Note: The environment and code, and the actual reference documentation are works in progress.

About the API

- uses resource URIs and standard HTTP requests
- the resources are simple operations using ideas found on the web
- is a Java Spring Boot project on the backend (I created the server and wrote the Java code)
- uses working Java methods
- Postman and cURL used to test the requests and confirm the expected responses

About the Documentation

The documentation design is based on various public API References. While reviewing the public API references, the primary aspects considered were content and format.

Contents

Authentication

This service uses **Spring Web Security** API keys and secrets.

Base URL

http://localhost:8080/api/

Resources

X-men

The **xmen** resource allows you to create, retrieve, update, and delete information about Marvel, X-men characters.

Properties

Property	Type	Description	Example
id	Integer	The unique identifier for the character	1
name	String	The character's real name.	"Scott Summers"
heroName	String	The character's hero name	"Cyclops"
power	String	The character's power(s).	"Optic blasts"
debut	String	The comic book title and year of debut.	"X-Men #1 (1963)"
creator	String	The character's creator(s).	"Stan Lee, Jack Kirby"

POST /xmen

Create a new X-men character. All parameters are required in the request.

Parameter(s)	id, name, heroName, power, debut, creator *required*
Message Body	{ "id": 1, "name": "Hisako Ichiki", "heroName": "Dust", "power": "Sand manipulation", "debut": "New X-Men #133 (2002)", "creator": "Grant Morrison, Ethan Van Sciver" }

```
# REQUEST
curl -L 'http://localhost:8080/api/xmen' \
-H 'Content-Type: application/json' \
-X-API-KEY: '{key}'
-d '{
    "id": 1,
    "name": "Hisako Ichiki",
    "heroName": "Dust",
    "power": "Sand manipulation",
    "debut": "New X-Men #133 (2002)",
    "creator": "Grant Morrison, Ethan Van Sciver"
}
```

```
# RESPONSE
200 OK

{
    "id": 1,
    "name": "Hisako Ichiki",
    "heroName": "Dust",
    "power": "Sand manipulation",
    "debut": "New X-Men #133 (2002)",
    "creator": "Grant Morrison, Ethan Van Sciver"
}
```

Needles

The **needles** resource matches specified words in at block of text. Send up to five different words (needles) and a single text block (haystack). Find the needles are in the haystack.

Properties

Property	Type	Description	Example
haystack	Integer	The text block to search for word matches.	"haystack":"Like finding a needle in a haystack"
needles	String Array	The word(s) to match	"needles":["finding","needle","a", "haystack"]

POST /needles

If the request contains more than five needles, the response will return the following:

"The maximum number of needles allowed is 5."

Parameter(s)	haystack *required*
	needles *required*

Message Body	{
	"haystack":"Like finding a needle in a haystack",
	"needles":["finding","needle","a","haystack","needles"]
	}

```
# REQUEST

curl -L 'http://localhost:8080/api/needles' \
-H 'Content-Type: application/json' \
-X-API-KEY: '{key}'
-d '{"haystack":"Like finding a needle in a
haystack","needles":["finding","needle","a","haystack","need
les"]}'
```

```
# RESPONSE

200 OK

{"needles":[
   {"count":"1","needle":"finding"},
   {"count":"1","needle":"needle"},
   {"count":"2","needle":"a"},
   {"count":"1","needle":"haystack"},
   {"count":"0","needle":"needles"}
]}

200 OK

"The maximum number of needles allowed is 5."
```

Numbers

The **numbers** resource calculates a set of random lottery numbers for the specified game. Send a lottery game type and receive a set of random numbers.

Properties

Property	Type	Description	Example
game	String	The specific game for which to get the random numbers.	pick4

GET /numbers

Note:

- The *game* parameter can only have one of three values: *pick4, pick5, lotto*
- Only one game type can be sent in the request.

If the request contains an invalid game, the response will return the following:

<<game>> is not a valid game type. Value must be pick4, pick5, or lotto.

Parameter(s)	game *required*
Message Body	Not applicable

```
# REQUEST

curl -L 'http://localhost:8080/api/numbers?api_key={}&game={}'
```

```
# RESPONSE

200 OK

{"pick4": [3,21,5,15]}

{"pick5": [17,5,22,40,46]}

{"lotto": [17,3,19,20,4,43]}

200 OK

<<game>> is not a valid game type. Value must be pick4, pick5,
or lotto.
```

Responses

200

Standard response for a successful HTTP request. The actual response will depend on the actual request method used. Refer to the applicable resource section for specific 200 response message.

400

The request cannot be filled due to bad syntax.

Field	Example
timestamp	YYYY-MM-DDT00:00:00.000+00:00
status	400
error	Bad Request
path	/api/example

```
# RESPONSE


400 Bad Request

{
    "timestamp": "2024-08-15T00:03:04.554+00:00",
    "status": 400,
    "error": "Bad Request",
    "path": "/api/numbers"
}
```

404

The requested resource could not be found but may be available again in the future. Subsequent requests by the client are permissible.

Field	Example
timestamp	YYYY-MM-DDT00:00:00.000+00:00
status	404
error	Not Found
path	/

```
# RESPONSE


404 Not Found

{
    "timestamp": "2024-08-16T12:24:11.041+00:00",
    "status": 404,
    "error": "Not Found",
    "message": "No static resource .",
    "path": "/"
}
```

500

A generic error message given when no more specific message is suitable.

For authentication error, the response will return the following:

“Missing API Key or Secret”

Field	Example
timestamp	YYYY-MM-DDT00:00:00.000+00:00
status	500
error	Internal Server Error
path	/api/example

```
# RESPONSE


500 Internal Server Error

{
    "timestamp": "2024-08-16T13:18:45.511+00:00",
    "status": 500,
    "error": "Internal Server Error",
    "message": "Missing API Key or Secret",
    "path": "/api/data"
}
```