# DEVELOPER CHALLENGE

Applicant: Klaus Wenger
Start Date: 30.11.2015

# Implement a RESTful API with NodeJS

## Skills

- demonstrate your coding skills and your ability to structure code in a reasonable way
- prove that you are able to complete a challenge within a given time frame
- show that you are able to adopt modern technologies in very short time

## Specification

Implement a RESTful API for managing wines.

The response format is JSON. Encoding is UTF-8.
Dates are represented in unix time.

Implement the following resources and resource methods:

### GET /wines/

Returns a list of wine object which are stored in the database.

Optional search parameters which filter the result:

- `year`
- `name`
- `type`
- `country`

Example response:

```
HTTP/1.1 200 OK
[{
   id: 1,
   name: 'Pinot noir',
   year: 2011,
   country: 'France',
   type: 'red',
   description: 'Sensual and understated'
}, {
   id: 2,
   name: 'Zinfandel',
   year: 1990,
   country: 'Croatia',
   type: 'red',
   description: 'Thick and jammy'
}]
```

## POST /wines/

Creates a new wine database entry.
The response is the created wine object.

Example response:

```
HTTP/1.1 200 OK
{
    id: 3,
    name: 'Cabernet sauvignon',
    year: 2013,
    country: 'France',
    type: 'red',
    description: 'The Sean Connery of red wines'
}
```

`name`, `year`, `country` and `type` are mandatory and are validated.

Valid values for `type` are *'red'*, *'white'* and *'rose'*.

If the request is invalid, an error response is sent.

Example Error response:

```
HTTP/1.1 400 Error
{
    error: 'VALIDATION_ERROR',
    validation: {
        country: 'MISSING',
        year: 'INVALID'
    }
}
```

## PUT /wines/:id

Update the wine with the given id.
Response is the updated wine.

Example response:

```
HTTP/1.1 200 OK
{
    id: 3,
    name: 'Cabernet sauvignon',
    year: 2013,
    country: 'France',
    type: 'red',
    description: 'Similar to merlot'
}
```

Error response:

```
HTTP/1.1 400 Invalid Object
{
   error: 'UNKNOWN_OBJECT'
}
```

## GET /wines/:id

Retrieves a wine by id

Example response:

```
HTTP/1.1 200 OK
{
   id: 3,
   name: 'Cabernet sauvignon',
   year: 2013,
   country: 'France',
   type: 'red',
   description: 'Similar to merlot'
}
```

Error response:

```
HTTP/1.1 400 Invalid Object
{
   error: 'UNKNOWN_OBJECT'
}
```

## DELETE /wines/:id

Delete a wine by id

Example response:

```
HTTP/1.1 200 OK
{
   success: true
}
```

Error response:

```
HTTP/1.1 400 Invalid Object
{
   error: 'UNKNOWN_OBJECT'
}
```

## Setup

To setup your project:

- create a public repository on Github for version control
- deploy your project to Heroku
- use MongoDB as data store

(please note that all services offer a free tier)

## Implementation

- use NodeJs to implement the API
- base your implementation on the Restify library
- verify your sourcecode with eslint to meet your coding conventions
- use environment variables to configure your application (see 12-factor App)
- implement tests for your API implementation to verify your implementation sends the correct responses and http codes
- avoid comments in your source code: good source code should be self documenting
- do not commit dependent 3rd party libraries into your code repository

## Delivery

- the maximum timeframe for this challenge is 10 days, **ending 10.12.2015 at midnight**
- you complete the challenge by sending the link to your GitHub Repository and your Heroku Deployment within this timeframe to info@tresmo.de

## Success

The challenge is completed successfully, if …

- the implemented API fulfills the specifications above and has tests to prove that it is working correctly
- you send us some lines, which areas of your implementation or the specification should be optimized in the future to improve the project

The challenge is to be solved on your own. No further questions should be necessary. Make reasonable assumptions if something is unspecified.

Good Luck!