Team Red API Documentation

Connecting

Calls to the API are to be directed to the base API address. The base address is
`http://rin.cs.ndsu.nodak.edu:4567`

Requesting Resources

A resource is an object, or a collection of data, tracked by the server. You can request resources by sending HTTP requests to the API. Responses to resource requests will be in JSON format. You will need to parse the JSON responses yourself. Requests are formatted like this:

`http://rin.cs.ndsu.nodak.edu:4567/resource_path`

Here are the supported resource paths and their descriptions:

`/venues`

Returns a collection of venues tracked by Rin with their ID and Name.
Return values:

- count: The number of returned venues
- venues: A collection of venues by 'id' and 'name'.

`/venues/<ID>/meals`

Returns a collection of meals belonging to the venue with the specified ID.
Return values:

- count: The number of returned meals
- meals: A collection of meals with the following fields:
  - id: Unique integer ID of the meal
  - name: Text name as specified by venue
  - servingsizeoz: Portion of the meal of which the following field (nutritionvalues) applies to
  - nutritionvalues: An associative collection of nutrient names and their values. Values are made up of 'unit' (eg g for gram) and 'count'.

`/meals/<ID>`

Returns one meal matching the specified ID.
Return values:

- count: 0 if nothing is found, 1 if a meal is found
- meals: A container array for the returned meal
  - id: Unique integer ID of the meal
  - name: Name of the meal
  - servingsizeoz: Portion of the meal of which the following field (nutritionvalues) applies to
  - nutritionvalues: An associative collection of nutrient names and their values. Values are made up of 'unit' (eg g for gram) and 'count'.

`/recommendations/<nutrient>`

Returns a collection of recommendations concerning a specified nutrient.
Return values:

- count: number of recommendations found
- recommendations: a collection of recommendations with the following fields:
  - key: A unique identifier for the recommendation
  - nutrient: The nutrient that the recommendation addresses
  - recommendation: The message of the recommendation intended for the user
  - datasource: The origin of the data used to make this recommendation

Adding Resources

To add a resource, send a POST request to the resource URL with the specified parameters in the BODY of the request (instead of the HEADER).

/venues

Adds a new venue.
Parameters:
    name: Name of the new venue
Return values:
    Either 'error' or 'message' with a response about the operation

/venues/<ID>/meals

Adds a new meal associated with the venue with <ID>.
Parameters:
    name: Name of the meal
    servingsizeoz: Amount of meal in ounces that the following nutritionvaluesjson corresponds to
    nutritionvaluesjson: An associative collection of nutrient names and their values. Values are made up of 'unit' (eg g for gram) and 'count'.
Return values:
    Either 'error' or 'message' with a response about the operation

/recommendation

Adds a new recommendation.
Parameters:
    key: A unique identifier for the recommendation
    nutrient: The nutrient that the recommendation addresses
    recommendation: The message of the recommendation intended for the user
    datasource: The origin of the data used to make this recommendation
Return values:
    Either 'error' or 'message' with a response about the operation

Modifying Resources

There are no methods for modifying resources yet.

Troubleshooting

If your application is having trouble communicating with Rin, it could be a variety of reasons:

IPv6 Incompatibility

Rin is known to not communicate over IPv6 but still has an externally facing IPv6 address referenced by its DNS name. This problem is being looked into but there is a workaround in the meantime. If your app is resolving rin.cs.ndsu.nodak.edu to an IPv6 address, connect to 134.129.125.7 instead. (The advanced solution is to force your app to resolve this domain name to an A record instead of AAAA record.)

If none of the previous situations apply to you, the API service may be down.

Implementation Details

The Team Red API utilizes the following components:

- Ruby
- MySQL
- Sinatra
- JSON

Sinatra (http://www.sinatrarb.com/) is a web-framework for creating HTTP applications. The API itself is written in Ruby while Sinatra acts as a web server and does work to glue Team Red code to a data-interacting application. The API also utilizes the Ruby gem mysql2 for interacting with a MySQL database.

Packages needed for this system:

- bundler
- gem daemons
- gem json
- gem mysql2
- gem rubygems
- gem sinatra
- gem sinatra-contrib
- libmysqlclient-dev
- mysql-client
- mysql-server
- ruby
- [gem rack]
- [gem tilt]

Useful Links

Postman – Make API calls from your browser: https://www.getpostman.com/