

Standardized API Development

Nicolas De Smyter

FDS

March 17, 2017

- 1 Intro
- 2 Installation and configuration
 - First setup
 - Configure database
 - Configure model
 - Start API server
- 3 Query data

Loopback

LoopBack is a highly-extensible, open-source Node.js framework that enables you to create dynamic end-to-end REST APIs with little or no coding.

First setup

```
$ npm install -g loopback-cli  
$ lb
```

Configure database

```
$ lb datasource
? Enter the data-source name: postgres-db
? Select the connector for oracledb: PostgreSQL
Connector specific configuration:
? Connection String url to override other settings:
postgres://postgres:postgres@localhost/mantis
? host: localhost
? port: 5432
? user: postgres
? password: *****
? database: mantis
? install loopback-connector-postgresql@^2.4 Yes
```

Configure model

```
$ lb model
? Enter the model name: dive
? Select the data-source to attach dive to:
postgres-db
? Select model's base class PersistedModel
? Expose dive via the REST API? Yes
? Custom plural form (used to build REST URL):
dives
? Common model or server only? common
Let's add some dive properties now.
```

Configure model properties

Enter an empty property name when done.

? Property name: **id**

invoke loopback:property

? Property type: **number**

? Required? **Yes**

? Default value [leave empty for none]:

Link models

```
$ lb relation ? Name of the model to create the
relationship from: dive
? Relation type: has many
? Name of the model to create a relationship with:
participant
? Name for the relation: participants
? Custom foreign key: dive_id
? Whether a "through" model is required? No
```


Start API server

```
$ node .
```

In debugging mode:

```
$ DEBUG=loopback:connector:postgresql node .
```

Query data

Use the built-in explorer: <http://localhost:3000/explorer>

dive

[Show/Hide](#) | [List Operations](#) | [Expand Operations](#)

| | | |
|--------|--------------------|--|
| PATCH | /dives | Patch an existing model instance or insert a new one into the data source. |
| GET | /dives | Find all instances of the model matched by filter from the data source. |
| PUT | /dives | Replace an existing model instance or insert a new one into the data source. |
| POST | /dives | Create a new instance of the model and persist it into the data source. |
| PATCH | /dives/{id} | Patch attributes for a model instance and persist it into the data source. |
| GET | /dives/{id} | Find a model instance by {{id}} from the data source. |
| HEAD | /dives/{id} | Check whether a model instance exists in the data source. |
| PUT | /dives/{id} | Replace attributes for a model instance and persist it into the data source. |
| DELETE | /dives/{id} | Delete a model instance by {{id}} from the data source. |
| GET | /dives/{id}/exists | Check whether a model instance exists in the data source. |

The End

Questions ?

All code and presentation:

<https://github.com/ndsmyter/loopback-brownbag>

Nicolas.DeSmyter@esfds.com