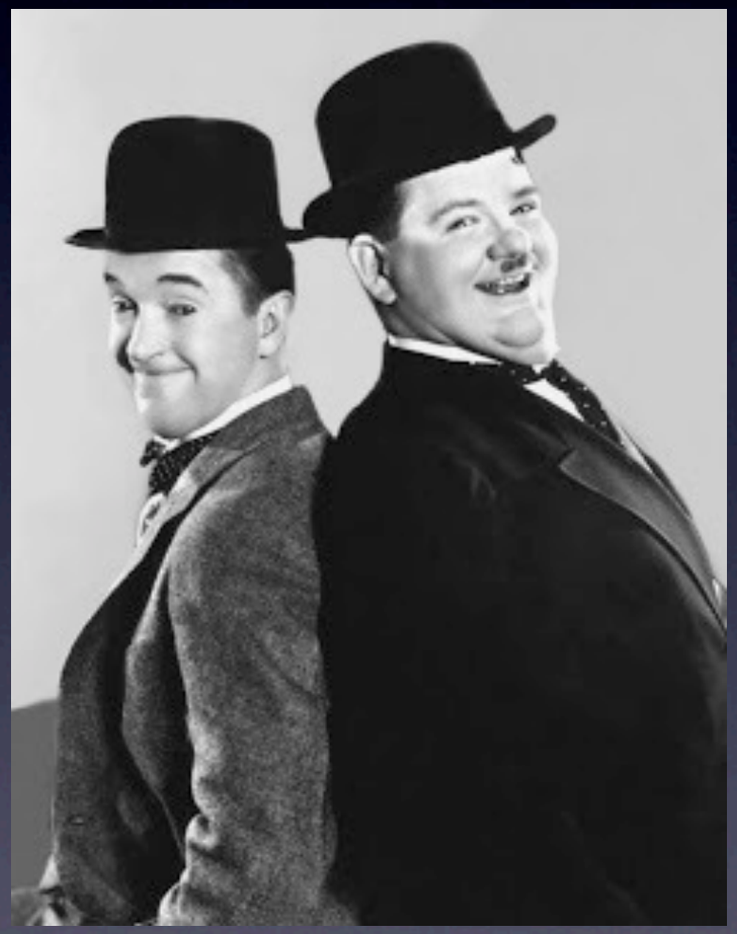
MongoTalk/Voyage

by Nico and Esteban



Pre-requisites

- MongoDB installed and running
- Pharo 2.0

Getting MongoTalk

Gofer it
  smalltalkhubUser: 'MongoTalkTeam' project:
  'mongotalk';
  configurationOf: 'MongoTalk';
  load.

ConfigurationOfMongoTalk load.

What is Mongo?

- NoSQL database
- Opensource
- Document oriented
- Powerful query language

MongoTalk basics

- MongoTalk is a driver for Mongo
- JSON/BSON (dictionaries)
- Mongo databases, collections and documents

Let's play with it!

- Creating a database
- Creating a collection
- Manipulating some documents

Databases and collections

```
| mongo db users |

mongo := Mongo default open.
```

“db is created on the fly”

```
db := mongo databaseNamed: ‘esug’.
```

“Same goes for collections”

```
users := db addCollection: ‘users’.
```

Insert/Update/Delete

```
users add: {
   ‘name’ -> ‘nico’.
   ‘age’ -> 27} asDictionary.
users
   update: { ‘name’ -> ‘nico’ } asDictionary
   with: {
      ‘name’ -> ‘nico’.
      ‘age’ -> 28 } asDictionary.

users delete: {‘name’ -> ‘nico’ } asDictionary.
```

Simple queries

"Use the traditional collection enumerating methods, with dictionaries"

```
users select: {'name' -> 'nico'} asDictionary.
users detect: {'age' -> 28} asDictionary.
```

Limit, offset, order

```
users
   select: {‘name’ -> ‘nico’} asDictionary
   limit: 5
   offset: 10
   order: {‘age’ -> 1} asDictionary
```

MongoQueries

```
users select: [ :each | each name = ‘nico’ ].
```

```
users detect: [ :each |
  (each name = ‘nico’) & (each age > 25) ]
```

...Voyage

Install

Gofer it
  smalltalkhubUser: 'estebanlm' project: 'Voyage';
  configurationOf: 'VoyageMongo';
  load.

ConfigurationOfVoyageMongo load.

Singleton vs. Instanced

| repository |

```
repository := VOMongoRepository
  host: 'localhost'
  database: 'esug'.

repository enableSingleton.
```

Basic operation

```
| user |

user := User new
   name: 'You';
   save.

user age: 'shhh'.
user save.

user remove.
```

Querying (1)

```
User selectAll.
User selectOne: [ :each | each name = 'you' ].
User selectMany: [ :each | each name = 'you' ].
```

Querying (2)

```
User selectOne: {
  #name -> ‘you’ } asDictionary.
User selectMany: {
  #name -> ‘you’ } asDictionary.
```

Querying (3)

```
User
selectMany: {  #name -> 'you' } asDictionary
sortBy: { #name -> VOOrder ascending }
limit: 10
offset: 10.
```

Querying (4)

```
User
  selectMany: {
    #name -> {
    ‘$regex’ -> ‘^y.\*’.
    ‘$options’ -> ‘i’ } asDictionary } asDictionary
sortBy: { #name -> VOOrder ascending }
limit: 10
offset: 10.
```

References

- Embedded objects
- Regular references
- Cyclic

Customize (1)

```
User class>>#mongoContainer
  <mongoContainer>
^VOMongoContainer new
  collectionName: ‘users’;
  kind: User;
  enableMissingContent;
  yourself
```

Customize (2)

```
User class>>#mongoName
  <mongoDescription>

^VOMongoToOneDescription new
  attributeName: 'name';
  beLazy;
  yourself.
```

Customize (3)

```
User class>>#mongoGroups
  <mongoDescription>

  ^VOMongoToManyDescription new
    attributeName: 'groups';
    beEager;
    kind: Group;
    kindCollection: Set;
    convertNullTo: [ MissingGroup new ];
    yourself.
```

Customize (4)

```
User class>>#mongoName
  <mongoDescription>
^VOMongoToOneDescription new
  accessor: (MAPluggableAccessor
    read: [ :user | user nameForPersist ]
    write: [ :user :v | user name: v ]);
  yourself.
```