Voyage

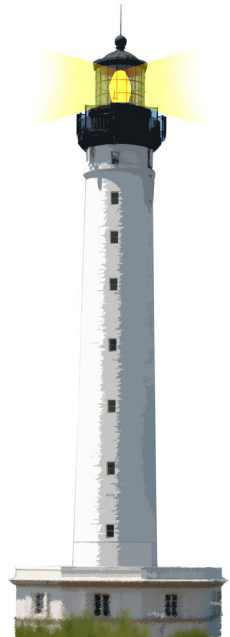
NoSQL Object Database

Damien Cassou, Stéphane Ducasse and Luc Fabresse

W4S11

Pharo

http://www.pharo.org



Goal

- To let you build a real little application
- Show you a nice way to store objects

MongoDB

- Document oriented, open-source NoSQL database
- Powerful query language
- Most popular document database so far

What is Voyage?

- Object-Document Mapper for MongoDB
- Think on Hibernate for MongoDB
- For Pharo, obviously ;)

Voyage Feature

- Simple
- Ensure object identity
- Provide error-handling
- Implement a connection pool

Setting a Connexion

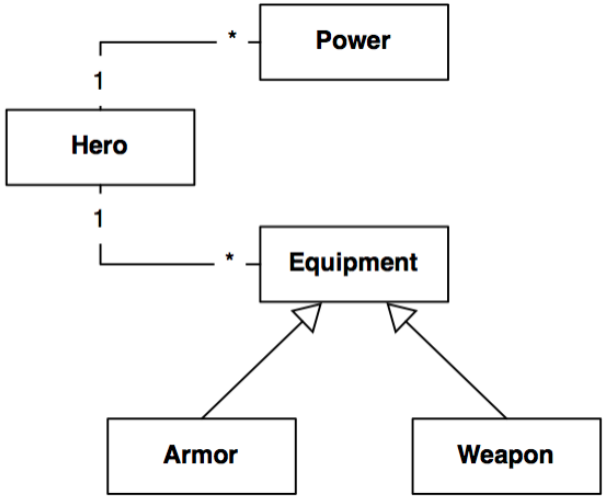
```
| repository |
repository := VOMongoRepository
    host: 'localhost'
    database: 'demo'.
repository enableSingleton.
```

Setting a In Memory Connexion

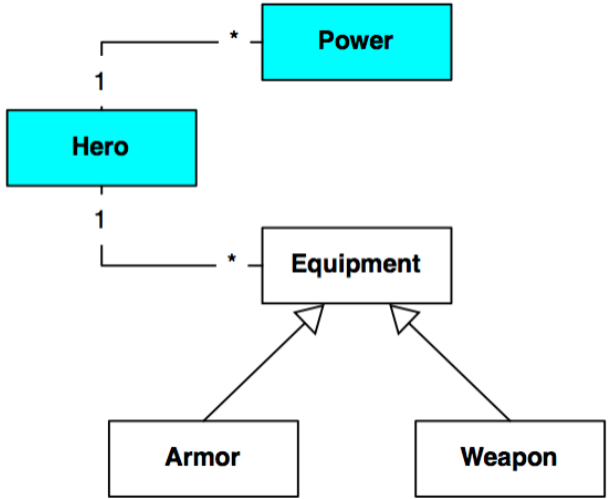
```
| repository |
repository := VOMemoryRepository new.
repository enableSingleton.
```

- Really nice to prototype
- Then can easily switch to a MongoDB

A Simple Model



A Simpler Model



A Simpler Model

```
Object subclass: #Hero
  instanceVariableNames: 'name level powers'
  classVariableNames: ''
  package: 'SuperHeroes'

Hero >> name
  ^ name
Hero >> name: aString
  name := aString
Hero >> level
  ^ level
Hero >> level: anObject
  level := anObject
Hero >> powers
  ^ powers ifNil: [ powers := Set new ]
Hero >> addPower: aPower
  self powers add: aPower
```

A Simpler Model

```
Object subclass: #Power
  instanceVariableNames: 'name'
  classVariableNames: ''
  package: 'SuperHeroes'.

Power>>name
   ^ name
Power>>name: aString
   name := aString
```

Root Classes

- Entry-point for our database
- Can be any class in the system
- Marked as root by `isVoyageRoot` class method

Root Classes

```
Hero class >> isVoyageRoot
  ^ true
```

Some Heroes

```
Hero new
    name: 'Spiderman';
    level: #epic;
    addPower: (Power new name: 'Super-strength');
    addPower: (Power new name: 'Wall-climbing');
    addPower: (Power new name: 'Spider instinct');
    save.

Hero new
    name: 'Wolverine';
    level: #epic;
    addPower: (Power new name: 'Regeneration');
    addPower: (Power new name: 'Adamantium claws');
    save.
```

In DB

```
> db.Hero.find()[0]
{
  "_id" : ObjectId("d847065c56d0ad09b4000001"),
  "#version" : 688076276,
  "#instanceOf" : "Hero",
  "level" : "epic",
  "name" : "Spiderman",
  "powers" : [
    {
      "#instanceOf" : "Power",
      "name" : "Spider instinct"
    },
    {
      "#instanceOf" : "Power",
      "name" : "Super-strength"
    },
    {
      "#instanceOf" : "Power",
      "name" : "Wall-climbing"
```

Querying

```
Hero selectAll.
> an OrderedCollection(a Hero, a Hero)
```

```
Hero selectOne: [ :each | each name = 'Spiderman' ].
> a Hero
```

```
Hero selectMany: [ :each | each level = #epic ].
> an OrderedCollection(a Hero, a Hero)
```

Querying 2

```
Hero selectOne: { #name –> 'Spiderman' } asDictionary.
> a Hero
```

```
Hero selectMany: { #level –> #epic } asDictionary.
> an OrderedCollection(a Hero, a Hero)
```

Querying 3

```
Hero
    selectMany: { #level –> #epic } asDictionary
    sortBy: { #name –> VOOrder ascending }
    limit: 10
    offset: 0.
> an OrderedCollection(a Hero, a Hero)
```

Some Operations

```
Hero count.
> 2
```

```
Hero count: [ :each | each name = 'Spiderman' ]
> 1.
```

```
Hero removeAll. "Beware of this!"
> Hero class
```

```
hero := Hero selectAll anyOne.
hero remove.
> a Hero
```

Defining Document Roots

- If you want to query them
- If you want to share objects between roots

Defining Document Roots

- Yes, Hero is a root on the example
- Power can be a root too (you can “share” powers between heroes)
- Any class can be declared as root, Voyage will manage it automatically

Defining Document Roots

```
Power class >> isVoyageRoot
 ^ true
```

```
Power new name: 'Fly'; save.
Power new name: 'Super−strength'; save.
```

```
fly := Power selectOne: [ :each | each name = 'Fly'].
superStrength := Power selectOne: [ :each | each name = 'Super
      −strength'].
Hero new
   name: 'Superman';
   level: #epic;
   addPower: fly;
   addPower: superStrength;
   save.
```

Important

Reset repository when changing the schema

```
VORepository current reset.
```

In DB

```
> db.Hero.find()[0]
{
  "_id" : ObjectId("d8474983421aa909b4000008"),
  "#version" : NumberLong("3874503784"),
  "#instanceOf" : "Hero",
  "level" : "epic",
  "name" : "Superman",
  "powers" : [
    {
      "#collection" : "Power",
      "#instanceOf" : "Power",
      "__id" : ObjectId("d84745dd421aa909b4000005")
    },
    {
      "#collection" : "Power",
      "#instanceOf" : "Power",
      "__id" : ObjectId("d84745dd421aa909b4000006")
    }
```

Relationships

- Root references (kind of “foreign keys”)
- Voyage handles cyclic references of root objects
- Beware, Voyage does not support cyclic references of embedded objects (yet)

Conclusion

- We can easily save objects in MongoDB
- More information in **Entreprise Pharo: a Web Perspective**

A course by

and

in collaboration with