

MONGODB

with Rails & MongoMapper

Features

- Collection Oriented Storage of BSON documents
 - Binary JSON, key value pairs
- Full Indexing
- Inner Objects & Embedded Arrays
- Replication, Failover & Auto-sharding(1.1)
 - Master/Slave, Replica Pairs, Master/Master
- Schema-less

Data Types

| MySQL | MongoDB |
|--------------|----------|
| - | *Array |
| Blob | Binary |
| TinyInt | *Boolean |
| Date | Date |
| Float | Float |
| - | *Hash |
| Integer | Integer |
| - | *Object |
| Varchar/Text | String |
| Datetime | Time |

Terminology

| MySQL | MongoDB |
|----------|------------|
| database | database |
| table | collection |
| column | key |
| record | document |

Install MongoDB

<http://www.mongodb.org/display/DOCS/Downloads>

```
$ sudo mkdir -p /data/db/
```

```
$ sudo chown -R YOUR_USERNAME /data
```

```
$ curl -O http://downloads.mongodb.org/...
```

```
$ tar xzf mongodb-xxx.tgz
```

```
$ cd mongodb-xxx
```

```
$ ./mongod &
```

Mongo Binaries

```
| -- bin  
|   |-- mongo           (the database shell)  
|   |-- mongod         (the core database server)  
|   |-- mongos         (auto-sharding process)  
|   |-- mongodump      (dump/export utility)  
|   `-- mongorestore   (restore/import utility)
```

Master/Slave Setup

```
$ mongod --master
```

```
$ mongod --slave --source masterhostname:port
```

Install Mongo Ruby Driver

```
$ sudo gem install mongo
```

```
$ sudo gem install mongo_ext
```

Twitter Example

```
> require 'json'
> require 'open-uri'
> require 'mongo'
> include Mongo

> url = 'http://search.twitter.com/trends.json'
> buffer = open(url, "UserAgent" => "Ruby").read
> result = JSON.parse(buffer)

> c = Connection.new('localhost')
> db = c.db("twitter")
> trends = db.collection("trends")
> trends.insert result
> trends.find_one
> {"_id"=>4adcc2b17fe8b3038f000001, "trends"=>
  [{"name"=>"#MusicMonday", "url"=>"http://
search.twitter.com/search?q=%23MusicMonday"}]...}
```

Already have an account? [Sign In](#) or [Sign Up](#)








mongoHQ
yep, it's that easy.

Hosted Database Goodness.

MongoHQ lets you create, host and interact with MongoDB instances. We fell in love with MongoDB and wanted to provide people with a fast and easy way to discover its awesomeness. MongoHQ lets you take that first step into the world of database goodness.

 [Request a Beta Invitation](#)

-  Hosted MongoDB instances
-  Support for Remote Mongo databases
-  Secure, Private Databases
-  Blazin' Fast
-  Database Replication (Coming soon!)

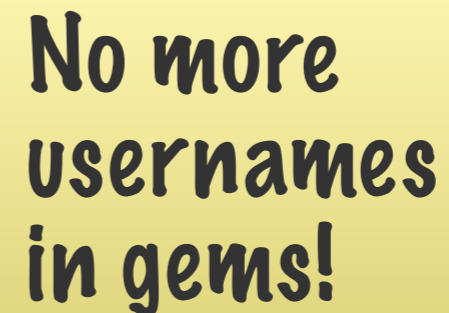
Install MongoMapper

```
$ gem install gemcutter
```

```
$ gem tumble
```

```
$ gem install mongo_mapper
```

```
$ gem install mongo_mapper \  
-s http://gemcutter.org
```



No more
usernames
in gems!

Setup Rails

config/initializers/mongo.rb

```
db_config = YAML::load(File.read(RAILS_ROOT + "/config/database.yml"))

if db_config[Rails.env]
  && db_config[Rails.env]['adapter'] == 'mongodb'
  mongo = db_config[Rails.env]
  MongoMapper.connection = Mongo::Connection.new(mongo['hostname'])
  MongoMapper.database = mongo['database']
end
```

config/database.yml

```
development:  
  adapter: mongodb  
  host: localhost  
  database: foo_development
```

config/environment.rb

```
Rails::Initializer.run do |config|  
  config.gem "mongo_mapper"  
  config.frameworks -= [ :active_record ]  
end
```

Associations

```
class Article
  include Mongomapper::Document

  key :title, String
  key :body, String
  key :tags, Array
  key :published, Boolean
  key :published_at, Time
  timestamps!

  has_many :comments,
    :dependent => :destroy
end
```

```
class Comment
  include Mongomapper::Document

  key :email, String
  key :body, String
  key :article_id, String
  timestamps!

  belongs_to :articles
end
```

Document ID's
are
alphanumeric
strings

```
a = Article.create({
  :title      => "Lorem Ipsum",
  :body       => "Lorem ipsum dolor...",
  :tags       => ["old", "latin"],
  :published  => true,
  :published_at => mktime(1981, 11, 25, 2, 30)
})
```

```
a << Comment.new({
  :email => "foo@bar.com",
  :body  => "First post!"
})
```

```
a.save
a.comments
```

```
Comment.first
```



No date
select
helpers

```
Article.all(  
  :conditions => {  
    :title => "Lorem Ipsum",  
    :tags => "latin"},  
  :order => "published_at desc")
```

```
Article.find(:all)
```

```
Article.first
```

```
Article.find(:first)
```

```
Article.last
```

```
Article.find
```

```
Article.paginate(:page => 1, :per_page => 5)
```

Embedded Documents

```
class Article
  include Mongomapper::Document

  key :title, String
  key :body, String
  key :tags, Array
  key :published, Boolean
  key :published_at, Time
  timestamps!

  has_many :comments
end
```

```
class Comment
  include
  Mongomapper::EmbeddedDocument

  key :email, String
  key :body, String
timestamps!
belongs_to :articles
end
```

```
a = Article.create({
  :title      => "Lorem Ipsum",
  :body       => "Lorem ipsum dolor...",
  :tags       => ["old", "latin"],
  :published  => true,
  :published_at => mktime(1981, 11, 25, 2, 30)
})
```

```
a << Comment.new({
  :email => "foo@bar.com",
  :body  => "First post!"
})
```

a.save

a.comments

~~Comment.first~~

Polymorphic Associations

```
class Article
  include Mongomapper::Document

  key :title, String
  key :body, String
  key :tags, Array
  key :published, Boolean
  key :published_at, Time
  timestamps!

  has_many :comments,
    :as => :commentable,
    :class_name => 'Comment'
end
```

```
class Comment
  include Mongomapper::Document

  key :email, String
  key :body, String
  key :commentable_id, String
  key :commentable_type, String
  timestamps!

  belongs_to :articles,
    :polymorphic => true
end
```

Backup Strategy

- No true point-in-time snapshotting yet.
- Procedure
 - Run master/slave
 - Shutdown Slave
 - Run mongodump
 - Restart Slave
- Slave catches up with Master

Ideas

- Liquid + Mongo = User defined documents
- Heroku + MongoHQ for hosting

Issues

- ActionView helpers tightly coupled to ActiveRecord
 - `date_select` & `datetime_select`
- Roll your own pagination helpers
- No generators
- No GUI's
- Need to account for missing keys

Source Highlights

- Examples
 - `test/models.rb`
- Types
 - `lib/mongo_mapper/support.rb`
- Validations
 - `lib/mongo_mapper/validations.rb`

References

- <http://www.mongodb.org/display/DOCS/Quickstart>
- <http://wiki.github.com/jnunemaker/mongomapper/quickstart>
- <http://www.viget.com/extend/getting-started-with-mongodb-mongomapper/>