Module 1, Assignment 1: MongoDB Ruby Driver Connection

The overall goal of the assignment is to:

- verify your MongoDB instance is running
- import sample data into a MongoDB collection
- verify you can communicate with the MongoDB server from a Ruby program
- have a working development environment with test data to follow along with the lectures

The functional goal of the assignment is to:

- populate the database with the `zips` test data from the MongoDB web site
- create a ruby program that can manipulate the `zips` collection in MongoDB.
- leave you at a point where you can locally implement many of the queries (and more) discussed in the upcoming lectures.

Documentation for the MongoDB Ruby Driver is located on the MongoDB site at https://api.mongodb.org/ruby/current/. You can locate API reference information using the tabs at the upper right side of the page http://api.mongodb.org/ruby/current/Mongo.html.

Getting Started

1. Download the `zips.json` document from http://media.mongodb.org/zips.json
2. Start your MongoDB server. This can be platform/installation-specific but essentially involves the start of the `mongod` process on your machine. See the installation lecture if you have not yet done so.
3. Download and extract the starter set of files. The root directory of this starter set will be referred to as the root directory of your solution.

```
---student-start
    |-- assignment.rb
    |-- .rspec (important hidden file)
    `-- spec
        |-- assignment_spec.rb
        `-- spec_helper.rb
```

   - assignment.rb - your solution must be placed within this file
   - spec - this directory contains tests to verify your solution. You should not modify anything in this directory

4. Install the following gems. You may already have them installed.

```
$ gem install rspec
$ gem install rspec-its
$ gem install mongo -v 2.1.2
```

5. Run the rspec command from the project root directory (i.e., `student-start` directory) to execute the unit tests within the spec directory. This should result in several failures until you complete your solution in `assignment.rb`.

```
$ rspec

(N) examples, (N) failures
...
```

6. Implement the Ruby technical requirements in `assignment.rb`

Technical Requirements

1. Use `mongoimport` to import the `zips.json` JSON file you downloaded from the MongoDB site in `getting started`. The command example, below, shows a `drop` of the existing collection if it already exists.

   - use the `test` database
   - use the `zips` collection

   ```
   $ mongoimport --drop --db test --collection zips zips.json
   ...    connected to: localhost
   ...    dropping: test.zips
   ...    imported 29353 documents
   ```

2. Implement a file `assignment.rb` that will:

   - require the `mongo` gem
   - optionally set the `Mongo::Logger.logger.level` to `::Logger::INFO` or `::Logger::DEBUG`
   - define a class called `Solution`. The grader will look for a class with this exact name.

3. Implement a class method in the `Solution` class called `mongo_client` that will:

   - create a `Mongo::Client` connection to the server using a URL (.e.g., 'mongodb://localhost:27017')
   - configure the client to use the `test` database
   - return the `Mongo::Database` client

   ```
   $ rspec -e rq03
   ```

4. Implement a class method in the `Solution` class called `collection` that will:

   - return the `zips` collection (`Mongo::Collection`)

   ```
   $ rspec -e rq04
   ```

5. Implement an instance method in the `Solution` class called `sample` that will:

   - return a single document from the `zips` collection from the database. This does not have to be random. It can be first, last, or any other document in the collection.

   ```
   $ rspec -e rq05
   ```

6. (ungraded) Use an instance of the `Solution` class to execute a pretty-print of the sample document. You will need to require `pp` for this to work.

   ```
   s=Solution.new
   pp s.sample
   ```

7. (ungraded) Use the Ruby `irb shell` to pretty-print a sample document from the collection without directly using your assignment.rb.

   ```
   $ irb
   > require 'mongo'
   > require 'pp'
   > db=Mongo::Client.new('mongodb://localhost:27017')
   > db=db.use('test')
   > pp db[:zips].find.first
   {"_id"=>"01001",
    "city"=>"AGAWAM",
    "loc"=>[-72.622739, 42.070206],
    "pop"=>15338,
    "state"=>"MA"}
   ```

8. (ungraded) Repeat the previous step, except this time load the code from your `assignment.rb` to automate getting the connection to MongoDB using the `Solution.mongo_client` class method. You can also use `Solution.collection` to get the collection but most of the examples in the lecture will use the syntax `db[:zips]` to refer to the collection.

```
$ irb
> require './assignment.rb'
> db=Solution.mongo_client
> pp db[:zips].find.first
{"_id"=>"01001",
 "city"=>"AGAWAM",
 "loc"=>[-72.622739, 42.070206],
 "pop"=>15338,
 "state"=>"MA"}
```

9. (ungraded) Use the Ruby `irb shell` to execute one of the commands shown in the lecture.

Self Grading/Feedback

Unit tests have been provided in the bootstrap files that can be used to evaluate your solution. They must be run from the same directory as your solution.

```
$ rspec
........

(N>0) examples, 0 failures
```

Submission

There is no submission required for this assignment but the skills learned will be part of a follow-on assignment so please complete this to the requirements of the unit test.

Last Updated: 2015-11-16