

Amazon S3 persistent Ruby objects

Contributed by George Malamidis
Tuesday, 10 July 2007
Last Updated Tuesday, 10 July 2007

{mos_sb_discuss:50}

I have occasionally participated in conversations around the subject of the database as a product with an expiry date, destined to eventually be replaced by highly distributed data storage models. Given the current technological state, this sounds much a like science fiction scenario, but services like AWS S3 bring the idea closer to science and further from fiction.

{mosgoogle}

Although S3's data storage and retrieval model looks presently better suited for larger units of data (e.g. media content), it would be interesting to investigate how it could be applied as an Object persistence service.

In the following example, we will use Ruby's `AWS::S3` library to create a class resembling Ruby on Rails' `ActiveRecord::Base`, allowing Objects to be persisted to and retrieved from an S3 Bucket.

Objects need to be somehow serialized and de-serialized in order to be successfully stored and retrieved from S3. YAML is one of the standard means to object serialization in Ruby, so we will be making use of it.

```
require 'yaml'
```

```
require 'aws/s3'
```

```
class S3Record
```

```
  attr_accessor :id
```

```
  def initialize(attrs = {})
```

```
    attrs.each { |k, v| instance_eval "self.#{k} = v" }
```

```
  end
```

```
end
```

Requiring `YAML` provides `S3Record` with, among other functionality, a `to_yaml` instance method. Next, we add the ability to persist an instance of `S3Record` to S3.

```
def create
```

```
AWS::S3::S3Object.find(@id, self.class.name)

raise "Object with key [#{@id}] already exists"

rescue AWS::S3::NoSuchKey

  AWS::S3::S3Object.store(@id, self.to_yaml, self.class.name)

end
```

The first parameter to the `AWS::S3::S3Object#find` method is the unique identifier by which the Object will be keyed when stored and will be the one used to find the object.

The second parameter is the name of the Bucket in which the object will be stored. Here, we use the name of our class as the bucket name. This implies that a bucket with a matching name to this of our class must exist before we can start storing objects.

[Read more](#)