

- [About](#)

Making Twitter mobile client with Ruby using Rhodes & Rhosync (Part 1)

10Feb09

Once you play with [sample applications](#) provided from rhomobile, now you might want to make something on your own.

I've seen lots of desktop client app tutorial with "Creating Sample Twitter App" examples, so I also tried something similar.

I will try to explain my code examples step by step including error messages and mistakes I made, so that you can not only experience what it is like to develop mobile app using Rhodes/Rhosync, but also debug similar problems when you make your own application.

In this tutorial, I will start from minimum functionality to display Twitter's public timeline on iPhone simulator. I will then implement authentication and basic Read/Create actions, so that you can show your friends's tweet, as well as post your own tweet from the iPhone simulator.

All the codes will be tested against 0.3.1 tags for both [rhosync](#) and [rhodes](#).

Before I start, special thanks to Rhomobile support team for answering [so many questions at google group](#).

What is Twitter API?

[Twitter](#) is one of the most post popular Ruby On Rails based web app. It is a micro blogging service where you can post your status in small text(140 chars).

Following Ruby On Rails convention, they have very clean ([REST based APIs](#)). If you have never tried their API, click http://twitter.com/statuses/public_timeline.xml. You will see list of 20 public tweets in xml format like below.

```
<status>
  <created_at>Sat Feb 07 23:49:04 +0000 2009</created_at>
  <id>1187577382</id>
  <text>
    @ceetee dude! How's it like out there! Been there before?
  </text>
  <source>
    <a href="http://www.atebits.com/software/tweetie/">Tweetie</a>
  </source>
  <truncated>>false</truncated>
  <in_reply_to_status_id>1187570116</in_reply_to_status_id>
  <in_reply_to_user_id>8620122</in_reply_to_user_id>
  <favorited>>false</favorited>
  <user>
    <id>11686132</id>
    <name>bombayaddict</name>
    <screen_name>b50</screen_name>
    <description>Bombay's Addict</description>
    <location>Bombay. Always. </location>
    <profile_image_url>
      http://s3.amazonaws.com/twitter_production/profile_images/63468508/photo2_normal.jpg
    </profile_image_url>
    <url>http://www.bombayaddict.com</url>
    <protected>>false</protected>
    <followers_count>462</followers_count>
  </user>
</status>
```

If you change file extension from .xml to .json or .rss, they will show in each format. Unlike your private tweets which I cover later, accessing public tweets does not require authentication.

There are [a few Ruby Libraries](#) to handle their API, but the APIs are so easy that you can do most of what you want to do by just using basic Ruby network library, such as [net/http](#).

Using Rhosync

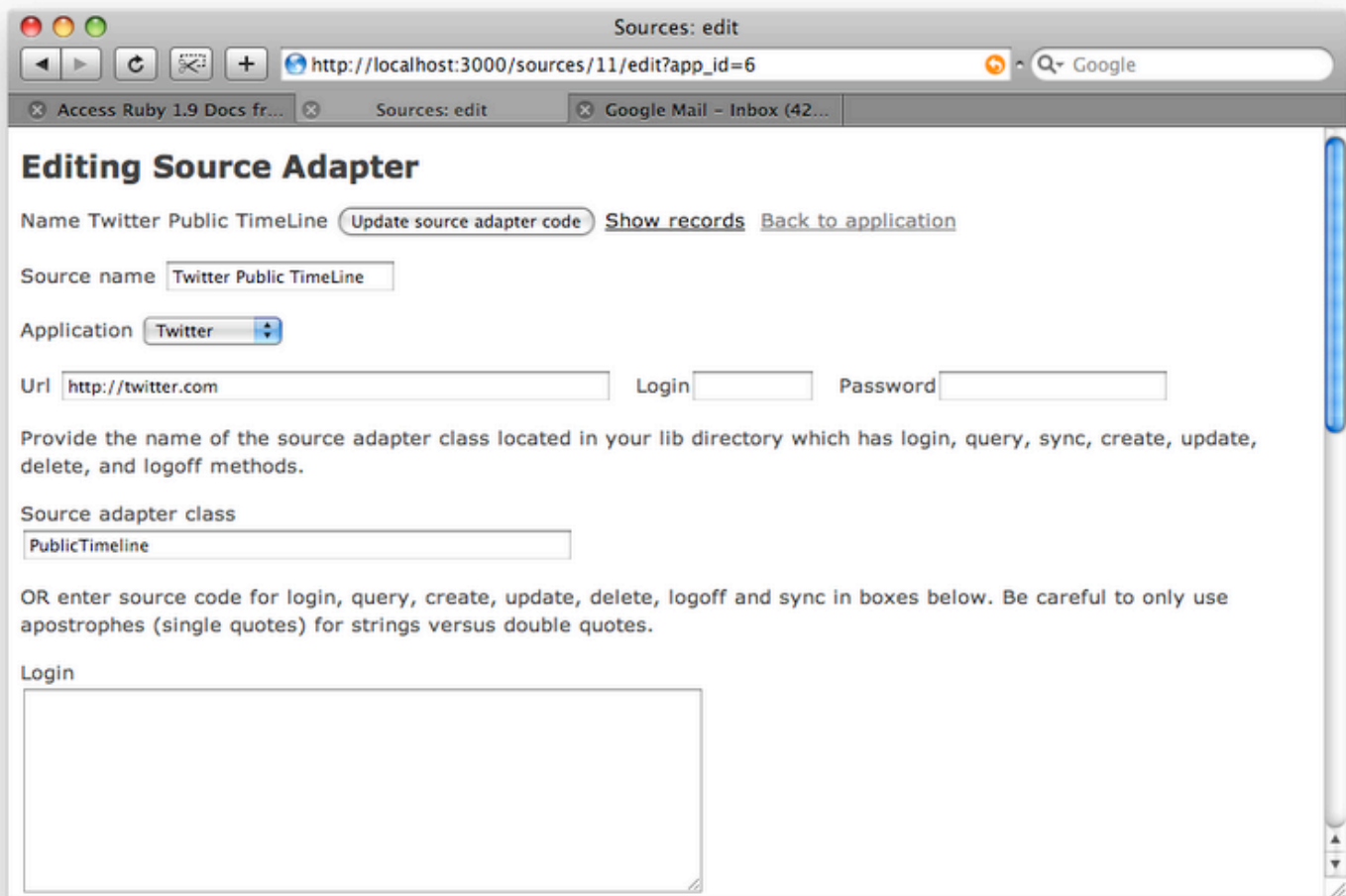
Rhosync will let you define common CRUD(Create, Read, Update, Delete) interfaces for mobile device via Rhodes.

Rhosync is built as a pure Ruby On Rails based application, so you should not have any problems if you know how to run and develop using Ruby On Rails.

Registering new source at Rhosync web form.

After you install Rhosync following the [instruction](#), let's startup the server, register your account, login to the server, and create an application called "Twitter". Once application is created, click "Add Source" and create an source called "Twitter Public Timeline"

Add "http://twitter.com" as url, but you can leave login and password for now, as Public Timeline does not require authentication.



The screenshot shows a web browser window titled "Sources: edit" with the URL `http://localhost:3000/sources/11/edit?app_id=6`. The browser has tabs for "Access Ruby 1.9 Docs fr...", "Sources: edit", and "Google Mail - Inbox (42...)". The page content is titled "Editing Source Adapter" and includes the following elements:

- Name: Twitter Public TimeLine
- Buttons: "Update source adapter code", "Show records", and "Back to application".
- Source name: Text input field containing "Twitter Public TimeLine".
- Application: Dropdown menu showing "Twitter".
- Url: Text input field containing "http://twitter.com".
- Login: Text input field (empty).
- Password: Text input field (empty).
- Instructions: "Provide the name of the source adapter class located in your lib directory which has login, query, sync, create, update, delete, and logoff methods."
- Source adapter class: Text input field containing "PublicTimeline".
- Instructions: "OR enter source code for login, query, create, update, delete, logoff and sync in boxes below. Be careful to only use apostrophes (single quotes) for strings versus double quotes."
- Login: A large text area for entering source code, currently empty.

To setup adapter, you can either generate the adapter using "rhogen" command, or type source code directly from the web form. I choose generating adapter myself for better debugging purpose. If your adapter code has some bugs, any exception will point the line number which contains the bug. If you type code in the

form, your code is kept at database and get evaluated dynamically when required, but exception only says your eval failed, which is harder to debug.

If you go for generating adapter, just type “PublicTimeline” at “Source Adapter Class”

rhogen

you need to create an adapter to talk to Twitter. Rhosync has some built in code generator, so creating the skeleton is very easy.

```
$ rhogen source PublicTimeline
Generating with source generator:
  [ADDED] lib/public_timeline.rb
```

This creates long list of code defining login, query, sync, create, update, delete, and logoff.

For this very first application, all you need to worry about are “query” and “sync” methods.

query method

Here is my minimalist code to fetch query from twitter.

```
def query
  uri = URI.parse(@source.url)
  res = Net::HTTP.start(uri.host, uri.port) {|http|
    http.get("/statuses/public_timeline.xml")
  }
  xml_data = XmlSimple.xml_in(res.body);
  @result = xml_data["status"]
end
```

I think it can't be as simple as this. It take url from @source instance variable. @source.url is the url you added when you first created the source from web page. Then it fetches tweets and put the body of the response into [XMLSimple library](#). Lastly I put each entry under into @result instance variable as Array.

sync method

Next is “sync” method. The code repopulated by rhogen command iterates through each entry of @result and put value into ObjectValue object. ObjectValue is a table where you can store the result of API response as key => value pair. This way, you can

The below are the difference between normal ORM(Object Relational Mapping) object table and ObjectValue table.

Normal table

ID	Industry	Name
44e804f2-4933-4e20-271c-48fced9450d	Technology	Mobio India
69ae8551-c0bd-0cb3-5c25-48ff9bae965a	Finance	vSpring

ObjectValue table

Object	Attribute	Value
44e804f2-4933-4e20-271c-48fced9450d	industry	Technology
44e804f2-4933-4e20-271c-48fced9450d	Name	Mobio India
69ae8551-c0bd-0cb3-5c25-48ff9bae965a	industry	Finance
69ae8551-c0bd-0cb3-5c25-48ff9bae965a	Name	vSpring

What I did not understand much was where “entrylist” and “namevalue_list” attributes are coming.

At the [rhomobile tutorial](#)

, @result is set like below at query method.

```
result = client.get_entry_list(session_id,'Account','',0,['name','industry'],10000,0)
```

And it also explains that “client” variable is available and not for REST API.

The Rhomobile tutorial page shows example using SugarCRM which uses SOAP API. Since Twitter uses REST API, I referred to [Lighthouse](#) and [BaseCamp](#) examples which are also Rails based app and uses REST API.

By examining these codes, you notice that there is a helper called RestAPIHelpers . Let’s use that. I included the helper at the top of the class like this

```
class PublicTimeline < SourceAdapter

  include RestAPIHelpers
```

They use “add_triple” method which also adds your API result into ObjectValue, so I decided to use the method to inject my Tweets.

```
def sync
  @result.each do |item|
    item_id = item["id"].first.to_i
    item.keys.each do |key|
      value = item[key] ? item[key][0] : ""
      add_triple(@source.id, id, key.gsub('-', '_'), value)
      # convert "-" to "_" because "-" is not valid in ruby variable names
    end
  end
end
```

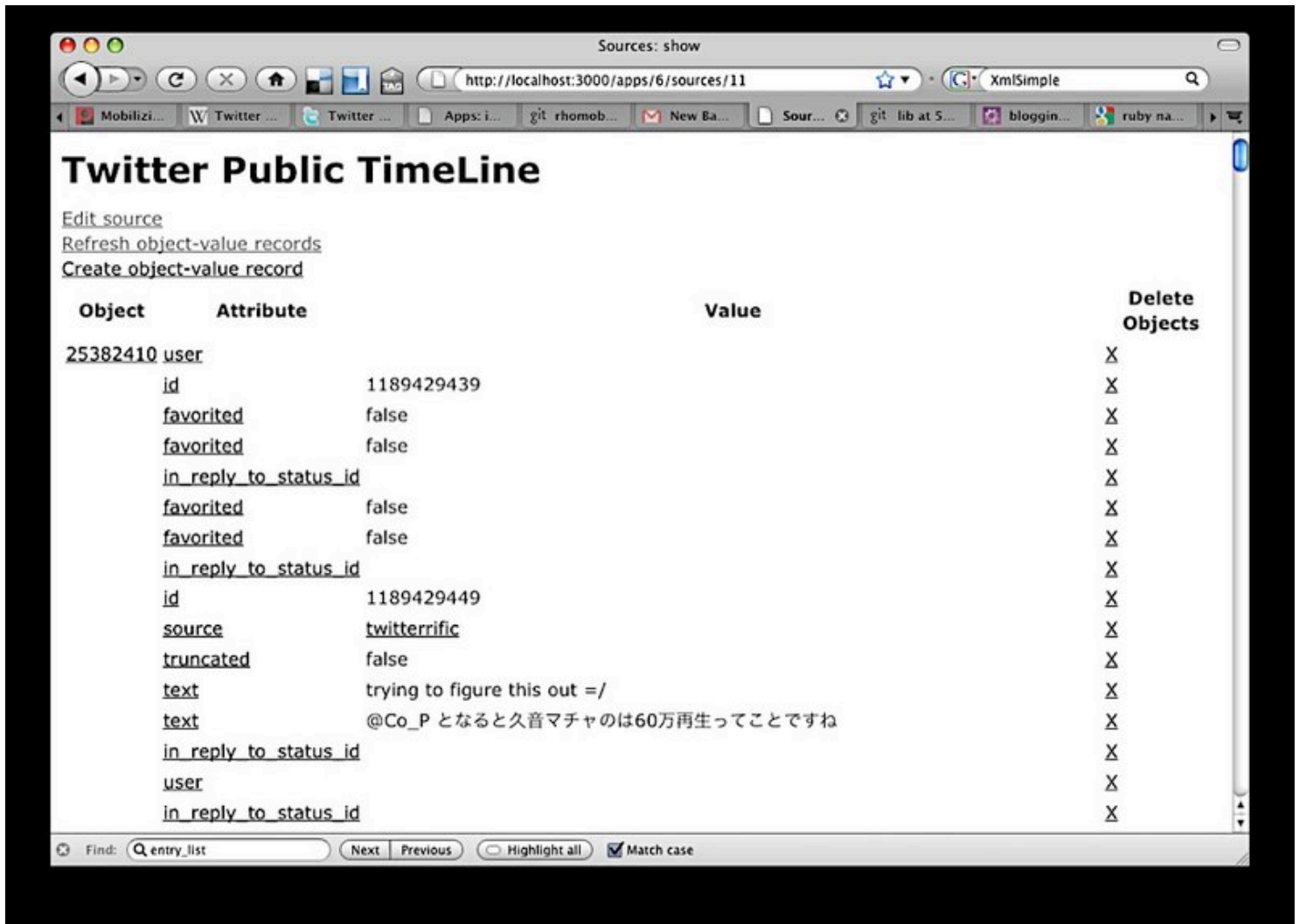
This is also simple one. @result contains array of Tweet where each tweet contains data in hash format like below.

```
{
  "text"=>
  ["@ceetee dude! How's it like out there! Been there before?"],
  "id"=>["1187577382"],
  "created_at"=>["Sat Feb 07 23:49:04 +0000 2009"]
}
```

I iterated through each hash key and send its key and value into “add_triple” methods

```
@result.first.keys
=> ["text", "id", "created_at"]
```

Now it’s done. Go back to “Editing Source Adapter” page, click “Show records”, then “Refresh object-value records”. This should will show all recent tweets.



Ummm, certain values are missing, such as “user”. By looking at original xml, you should see that certain attributes are nested, so just going through each key of hash won’t work. You need to iterate recursively every time value contains hash.

Final code

Here is the complete code including the change to recursively iterate through each tweets. I refactored the code by separating iteration part into “iteratekeys” and calls the method recursively if the hash value contains another hash. When you call iteratekeys, it also adds prefix, so “user” => {”screenname”} will be stored into “userscreen_name” key name at ObjectValue

```
class PublicTimeline < SourceAdapter

  include RestAPIHelpers

  def initialize(source)
    super
  end

  def query
    uri = URI.parse(@source.url)
    res = Net::HTTP.start(uri.host, uri.port) {|http|
      http.get("/statuses/public_timeline.xml")
    }
    xml_data = XmlSimple.xml_in(res.body);
    @result = xml_data["status"]
  end

  def sync
```

```

@result.each do |item|
  item_id = item["id"].first.to_i
  iterate_keys(:item => item, :item_id => item_id)
end
end

private
def iterate_keys(option)
  item = option[:item]
  item_id = option[:item_id]
  prefix = option[:prefix] || ""

  # item.keys => ["user", "favorited", "truncated"...]
  item.keys.each do |key|
    value = item[key] ? item[key][0] : ""
    if value.kind_of?(Hash) && value != {}
      # eg. :user => {:url => 'foo'} becomes user_url
      iterate_keys(:prefix => (prefix + underline(key) + "_"), :item => value, :item_id => item_id)
    else
      # This method is from rest_api_helper
      add_triple(@source.id, item_id, prefix + underline(key), value)
    end
  end
end

def underline(key)
  key.gsub('-', '_')
end
end

```

This is the complete “Show Record”, including nested attributes, such as user_id/name/url



Next

Since we now have a source adapter which loads public tweet, I will explain the code I wrote to display these tweets on iPhone simulator using Rhodes.

Filed under: [Rhodes](#) |

Tags: [tutorial](#)

[2 Responses to “Making Twitter mobile client with Ruby using Rhodes & Rhosync \(Part 1\)”](#)

[Feed for this Entry](#) [Trackback Address](#)



1. [1](#) **madhan** on [February 10, 2009](#) said:

Hi Makoto,

Its really good tutorial. I follow each and every step of this tutorial and i got the latest tweets in my local rhosync server. But how can i see those latest tweets in my emulators. From my emulator i posted some public timeline tweets but i cant see them in http://twitter.com/statuses/public_timeline.xml. but i can see those tweets in my emulator. Please guide me...

Thanks in Advance.



2. [2](#) **inouemak** on [February 10, 2009](#) said:

Hi, madhan.

Glad to know you were able to follow the tutorial without much problems. I am in the middle of writing tutorial on how to show public tweets on your iphone emulator.

Regarding posting part, it is less likely that you can see your own tweets at public tweets, because Twitter only shows 20 tweets out of million other tweets and it caches for 60 sec. Please wait until I cover the topic about posting/displaying your private tweets.

Thanks.

Makoto

Leave a Reply

Name (required)

Mail (will not be published) (required)

Website

Submit

☐ Notify me of follow-up comments via email.



[Subscribe in a reader](#)

search...

go

• Blogroll

- [My Github repository](#)
- [rhomobile | Google Groups](#)
- [日本語ブログ](#)

• Recent Posts

- [Making Twitter mobile client with Ruby using Rhodes Rhosync \(Part 3\)](#)
- [Aplix: Converting NTT Docomo i-appli to Windows Mobile Android, iPhone, S60 and even to portable game machine](#)
- [Making Twitter mobile client with Ruby using Rhodes & Rhosync \(Part 2\)](#)
- [Making Twitter mobile client with Ruby using Rhodes & Rhosync \(Part 1\)](#)
- [Hello world!](#)



• Categories

- [Rhodes](#)
- [Uncategorized](#)

• Meta

- [Log in](#)
- [Entries RSS](#)
- [Comments RSS](#)
- [WordPress.com](#)