**Ruby on Rails SDK Version 2 Implementation Steps**

**Setup**

1. Add social login interface code from your LoginRadius user account to your webpage.
2. Set-up the callback URL in your LoginRadius user account, this is the URL where user would be redirected after authentication. Example:- www.example.com/callback

Ruby wrapper for the LoginRadius API. Get social graph information and send messages using LoginRadius many social network clients!

Optional Asynchronous Event Machine friendly option!

## Installation

Add this line to your application's Gemfile:

gem "login\_radius", :path => "gemfilepath";

OR

get “login\_radius”, *:git => "git://github.com/bhgames/login\_radius.git"*

And then execute:

$ bundle

Or install it yourself as:

$ gem install login\_radius

## Usage

Take a peek:

@user\_profile = LoginRadius::UserProfile.new({

:token => "TOKEN",

:secret => "SECRET",

:async => false #Set to true if using EventMachine driven frameworks(must use Em Synchrony)

})

@user\_profile.access\_token

UserProfile = @user\_profile.userprofile

That's right, folks. It's that simple. In order to get access to LoginRadius API, you only need to create a UserProfile object with a user's access token and your site secret, and login.

### How to get a token and actually make a client

On the LoginRadius website, you are able to enter your own callback URL for your LoginRadius site. You will need to setup a route on whatever framework you use, maybe "/callback." Then, get the gem forward (www.example.com) and use it to set up a public web address for your localhost server. Then, you can enter the callback on LoginRadius as https://example.com/callback, for instance.

Paste their example login stuff onto your index page, and then enter Provider configuration on their LoginRadius site.

Now you're ready to go with testing!

When LoginRadius hits your callback after a user logs in, you'll see parameters like this:

Parameters: {"token"=>"yourtokenhere"}

This token is the token used in the above example. This is what you'll need to get your access token and user's profile. Just paste it into the code above, and you can immediately grab their user profile on the callback, login, and grab their contacts.

### Some examples

Below is just code exemplifying some of the more interesting methods and what they return.

@user\_profile.access\_token

=> false

Will return false, indicating you don't have access to that stream.

#### User Profile

@user\_profile.companies

=>[{:id=>"", :name=>""},

{:id=>"", :name=>""},

{:id=>"", :name=>""},

{:id=>"", :name=>""}]

@user\_profile.group

=>[{:id=>"", :name=>"", :email=>"", :description=>"", :type=>"", :country=>"", :postal\_code=>"", :logo=>"", :image=>"", :member\_count=>""},

{:id=>"", :name=>"", :email=>"", :description=>"", :type=>"", :country=>"", :postal\_code=>"", :logo=>"", :image=>"", :member\_count=>""},

{:id=>"", :name=>"", :email=>"", :description=>"", :type=>"", :country=>"", :postal\_code=>"", :logo=>"", :image=>"", :member\_count=>""}]

@user\_profile.post

=>[{:id=>"", :name=>"", :title=>"", :start\_time=>"", :update\_time=>"", :message=>"", :place=>"", :picture=>"", :likes=>"", :share=>"", :type=>""},

{:id=>"", :name=>"", :title=>"", :start\_time=>"", :update\_time=>"", :message=>"", :place=>"", :picture=>"", :likes=>"", :share=>"", :type=>""},

{:id=>"", :name=>"", :title=>"", :start\_time=>"", :update\_time=>"", :message=>"", :place=>"", :picture=>"", :likes=>"", :share=>"", :type=>""},

{:id=>"", :name=>"", :title=>"", :start\_time=>"", :update\_time=>"", :message=>"", :place=>"", :picture=>"", :likes=>"", :share=>"", :type=>""},

{:id=>"", :name=>"", :title=>"", :start\_time=>"", :update\_time=>"", :message=>"", :place=>"", :picture=>"", :likes=>"", :share=>"", :type=>""}]

@user\_profile.status

=>[{:id=>"", :text=>"", :date\_time=>"", :likes=>"", :place=>"", :source=>"", :image\_url=>"", :link\_url=>"", :name=>""},

{:id=>"", :text=>"", :date\_time=>"", :likes=>"", :place=>"", :source=>"", :image\_url=>"", :link\_url=>"", :name=>""},

{:id=>"", :text=>"", :date\_time=>"", :likes=>"", :place=>"", :source=>"", :image\_url=>"", :link\_url=>"", :name=>""},

{:id=>"", :text=>"", :date\_time=>"", :likes=>"", :place=>"", :source=>"", :image\_url=>"", :link\_url=>"", :name=>""},

{:id=>"", :text=>"", :date\_time=>"", :likes=>"", :place=>"", :source=>"", :image\_url=>"", :link\_url=>"", :name=>""}]

@user\_profile.album

=>[{:id=>"", :owner\_id=>"", :owner\_name=>"", :title=>"", :description=>"", :location=>"", :type=>"", :created\_date=>"", :updated\_date=>"", :cover\_image\_url=>"", :image\_count=>"", :directory\_url=>},

{:id=>"", :owner\_id=>"", :owner\_name=>"", :title=>"", :description=>"", :location=>"", :type=>"", :created\_date=>"", :updated\_date=>"", :cover\_image\_url=>"", :image\_count=>"", :directory\_url=>},

{:id=>"", :owner\_id=>"", :owner\_name=>"", :title=>"", :description=>"", :location=>"", :type=>"", :created\_date=>"", :updated\_date=>"", :cover\_image\_url=>"", :image\_count=>"", :directory\_url=>},

{:id=>"", :owner\_id=>"", :owner\_name=>"", :title=>"", :description=>"", :location=>"", :type=>"", :created\_date=>"", :updated\_date=>"", :cover\_image\_url=>"", :image\_count=>"", :directory\_url=>}]

@user\_profile.like

=>[{:id=>"", :name=>"", :category=>"", :created\_date=>"", :website=>"", :description=>""},

{:id=>"", :name=>"", :category=>"", :created\_date=>"", :website=>"", :description=>""},

{:id=>"", :name=>"", :category=>"", :created\_date=>"", :website=>"", :description=>""},

{:id=>"", :name=>"", :category=>"", :created\_date=>"", :website=>"", :description=>""}]

@user\_profile.video

=>[{:id=>"", :description=>"", :name=>"", :image=>"", :source=>"", :created\_date=>"", :owner\_id=>"", :owner\_name=>"", :embed\_html=>"", :updated\_date=>"", :duration=>"", :direct\_link=>""},

{:id=>"", :description=>"", :name=>"", :image=>"", :source=>"", :created\_date=>"", :owner\_id=>"", :owner\_name=>"", :embed\_html=>"", :updated\_date=>"", :duration=>"", :direct\_link=>""},

{:id=>"", :description=>"", :name=>"", :image=>"", :source=>"", :created\_date=>"", :owner\_id=>"", :owner\_name=>"", :embed\_html=>"", :updated\_date=>"", :duration=>"", :direct\_link=>""},

{:id=>"", :description=>"", :name=>"", :image=>"", :source=>"", :created\_date=>"", :owner\_id=>"", :owner\_name=>"", :embed\_html=>"", :updated\_date=>"", :duration=>"", :direct\_link=>""}]

And, finally, you can also make Post Status and Direct message:

params = {

:access\_token => @userprofile.access\_token,

:title => "LoginRadius SDK Title",

:url => "https://loginradius.com",

:status => "LoginRadius SDK Status",

:caption => "LoginRadius SDK caption",

:description => "LoginRadius SDK description”

}

@user\_profile.post\_status(params)

=> true

params = {

:access\_token => @userprofile.access\_token,

:to => @userprofile.contact.first[:id],

:subject => "LoginRadius SDK Subject",

:message => "LoginRadius SDK message"

}

@user\_profile.direct\_message (params)

=> true

Also, you can grab a user's companies:

@user\_profile.company

=> [{:id=>11111, :name=>"Name"}]

#### Social Network Contacts

Note this is all the contacts from every social network the user is connected to.

@user\_profile.contacts

=> [{:name=>"Name",

:email\_id=>"",

:phone\_number=>"",

:id=>"11111",

:profile\_url=>nil,

:image\_url=>nil,

:status=>nil,

:industry=>nil,

:country=>nil,

:gender=>nil}]

#### User Profile Stuff

When you login, the API returns a hash of user profile information, which is then turned into methods on the LoginRadius::UserProfile object for your accessing pleasure. A quick call to methods will show this.

@user\_profile.userprofile

=>[:id,

:provider,

:prefix,

:first\_name,

:middle\_name,

:last\_name,

:suffix,

:full\_name,

:nick\_name,

:profile\_name,

:birth\_date,

:gender,

:website,

:email,

:country,

...]

Calling .methods on a user profile right after a successful login call will reveal these. For more information, please read the section in caveats on the matter, as it's a bit esoteric.

## Caveats

This section is for the nitpickers. It's got notes about the finer pickings of this API for those who are wondering, but is not strictly necessary reading.

### The Bang Methods

When authenticated, a method like contacts will return

@user\_profile.access\_token

=> true

@user\_profile.contacts

=> [{:name=>"Name",

:email\_id=>"",

:phone\_number=>"",

:id=>"11111",

:profile\_url=>nil,

:image\_url=>nil,

:status=>nil,

:industry=>nil,

:country=>nil,

:gender=>nil},

...

When unauthenticated, a method like contacts will return false.

@user\_profile.access\_token

=> false

@user\_profile.contacts

=> false

However, if you want to be a bit more flammable, try using contacts!, the bang version. It still returns the same results when authed, but when unauthed, will actually raise a LoginRadius::Exception.

### The LoginRadius::UserProfile object before and after login

Once you've got a valid user\_profile, you have many methods available to you. Let's quickly take a look at the methods in a LoginRadius::UserProfile object that hasn't yet been authenticated:

@user\_profile.access\_token

=> false

@user\_profile.methods - Object.new.methods

=> [:secret,

:secret=,

:token,

:token=,

:async,

:async=,

:guid\_valid?,

:access\_token?,

:call\_api,

:post\_status,

:post\_status!,

:direct\_message,

:direct\_message!,

:contactes,

:event,

:mention,

:company,

:group,

:post,

:status,

:following,

:album,

:checkin,

:like,

:photo,

:audio,

:video,

:page]

Notice we've got a lot of stuff we can access. But we're not yet authenticated. Running login and then checking the new methods created, we see we have much more than before!

@user\_profile.access\_token

=> true

@user\_profile.userprofile

=> true

@user\_profile.methods - Object.new.methods

=> [:id,

:provider,

:prefix,

:first\_name,

:middle\_name,

:last\_name,

:suffix,

:full\_name,

:nick\_name,

:profile\_name,

:birth\_date,

:gender,

:website,

:email,

:country,

:thumbnail\_image\_url,

:image\_url,

:favicon,

:profile\_url,

:home\_town,

:state,

:city,

:industry,

:about,

:time\_zone,

:local\_language,

:language,

:verified,

:updated\_time,

:positions,

:educations,

:phone\_numbers,

:im\_accounts,

:addresses,

:main\_address,

:created,

:local\_city,

:profile\_city,

:local\_country,

:profile\_country,

:relationship\_status,

:quota,

:interested\_in,

:interests,

:religion,

:political,

:sports,

:inspirational\_people,

:https\_image\_url,

:followers\_count,

:friends\_count,

:is\_geo\_enabled,

:total\_statuses\_count,

:associations,

:num\_recommenders,

:honors,

:skills,

:current\_status,

:certifications,

:courses,

:volunteer,

:recommendations\_received,

:languages,

:public\_repository,

:hireable,

:repository\_url,

:age,

:patents,

:favorite\_things,

:professional\_headline,

:provider\_access\_credential,

:secret,

:secret=,

:token,

:token=,

:async,

:async=,

:guid\_valid?,

:access\_token?,

:call\_api,

:post\_status,

:post\_status!,

:direct\_message,

:direct\_message!,

:contactes,

:event,

:mention,

:company,

:group,

:post,

:status,

:following,

:album,

:checkin,

:like,

:photo,

:audio,

:video,

:page]

Holy moly we got a whole bunch of new methods! That's because on login, all the user profile information returned by the API is turned into dynamically generated GETTERs on the object that you can then use later.

### Running the tests in test/unit

The base\_test.rb file isn't meant to be run, it is inherited by user\_profile\_test.rb. In that file you'll notice the lines at the top of the file:

TOKEN = "yourtokenhere"

SECRET = "yoursecrethere"

These are meant to be changed to your API secret and some user's token. What I do is I run a dummy rails app locally and use the forward gem so that it can receive requests from the internet. Then I setup a callback route and login page on the app and use it to have LoginRadius ping me with tokens that I can paste into this test file.

Beware, LoginRadius invalidates any token after 15 minutes, so during testing, you'll have to repeat the token creation A LOT

To run the tests, do:

bundle exec ruby -Itest test/unit/user\_profile\_test.rb

If you login with a facebook user, for instance, you'll notice all the twitter/linked in tests failing. This is to be expected, you don't have access to those streams!

### Evented Frameworks

I've built this gem to work with frameworks like Cramp. I've done a little bit of testing by creating a basic event loop script in basic\_async\_test.rb in the test folder, which you can run to see that things work there.

I've also done some AB testing using a Cramp framework elsewhere, but you should take caution in using this gem in evented frameworks and make sure to use apache bench to test it for non-blocking yourself.