

---

## Socialization

Socialization is a Ruby Gem that allows any ActiveRecord model to [Follow](#), [Like](#) and/or [Mention](#) any other model. ActiveRecord or Redis can be used as a data store.

The Follow feature is similar to Twitter's follow. For example, John follows Jane. Unlike Facebook's "friendship", Follow is a one-way concept. The fact that John follows Jane doesn't mean that Jane follows John.

The Like feature works just like a Facebook Like. For example, John likes Pulp Fiction.

The Mention feature was written with Facebook mentions in mind. For example, John mentions Jane in a comment. Typically, Jane would be highlighted in the comment user interface and possibly notified that John mentioned her. This Facebook feature is occasionally called Tagging, although tagging is generally something [entirely different]([http://en.wikipedia.org/wiki/Tag\\_\(metadata\)](http://en.wikipedia.org/wiki/Tag_(metadata))).



## Installation

Add the gem to the gemfile: `gem "socialization"`

Run the generator: `rails generate socialization -s`

Or if you want to use Redis as your data store: `rails generate socialization -s --store=redis`

This will generate three migration files (when using ActiveRecord) and three models named Follow, Like and Mention. You may delete any of the Follow, Like or Mention models and migrations if you don't need that functionality in your application.

## Legacy Rails Support

This gem requires Rails 6 or better. Sorry!

## Usage

### Setup

Allow a model to be followed:

---

```
1 class Celebrity < ActiveRecord::Base
2   ...
3   acts_as_followable
4   ...
5 end
```

Allow a model to be a follower:

```
1 class User < ActiveRecord::Base
2   ...
3   acts_as_follower
4   ...
5 end
```

Allow a model to be liked:

```
1 class Movie < ActiveRecord::Base
2   ...
3   acts_as_likeable
4   ...
5 end
```

Allow a model to like:

```
1 class User < ActiveRecord::Base
2   ...
3   acts_as_liker
4   ...
5 end
```

Allow a model to be mentioned:

```
1 class User < ActiveRecord::Base
2   ...
3   acts_as_mentionable
4   ...
5 end
```

Allow a model to mention:

```
1 class Comment < ActiveRecord::Base
2   ...
3   acts_as_mentioner
4   ...
5 end
```

Or a more complex case where users can like and follow each other:

```
1 class User < ActiveRecord::Base
2   ...
```

---

```
3   acts_as_follower
4   acts_as_followable
5   acts_as_liker
6   acts_as_likeable
7   acts_as_mentionable
8   ...
9 end
```

---

### **acts\_as\_follower Methods**

Follow something

```
1 user.follow!(celebrity)
```

Stop following

```
1 user.unfollow!(celebrity)
```

Toggle

```
1 user.toggle_follow!(celebrity)
```

Is following?

```
1 user.follows?(celebrity)
```

What items are you following (given that an Item model is followed)?

```
1 user.followees(Item)
```

Number of followees (Requires followees\_count column in db)

```
1 def change
2   add_column :#{Table_name}, :followees_count, :integer, :default => 0
3 end
4
5 user.followees_count
```

---

### **acts\_as\_followable Methods**

Find out if an objects follows

---

```
1 celebrity.followed_by?(user)
```

All followers

```
1 celebrity.followers(User)
```

Number of followers (Requires followers\_count column in db)

```
1 def change
2   add_column :#{Table_name}, :followers_count, :integer, :default => 0
3 end
4
5 celebrity.followers_count
```

---

### acts\_as\_liker Methods

Like something

```
1 user.like!(movie)
```

Stop liking

```
1 user.unlike!(movie)
```

Toggle

```
1 user.toggle_like!(celebrity)
```

Likes?

```
1 user.likes?(movie)
```

Number of likees (Requires likees\_count column in db)

```
1 def change
2   add_column :#{Table_name}, :likees_count, :integer, :default => 0
3 end
4
5 user.likees_count
```

---

### acts\_as\_likeable Methods

Find out if an objects likes

```
1 movie.liked_by?(user)
```

All likers

```
1 movie.likers(User)
```

Number of likers (Requires likers\_count column in db)

```
1 def change
2   add_column :#{Table_name}, :likers_count, :integer, :default => 0
3 end
4
5 movie.likers_count
```

---

### acts\_as\_mentioner Methods

**Note that a “mentioner” is the object containing the mention and not necessarily the actor. For example, John mentions Jane in a comment. The mentioner is the comment object, NOT John.**

Mention something

```
1 comment.mention!(user)
```

Remove mention

```
1 comment.unmention!(user)
```

Toggle

```
1 user.toggle_mention!(celebrity)
```

Mentions?

```
1 comment.mentions?(user)
```

All mentionees

```
1 comment.mentionees(User)
```

Number of mentionees (Requires mentionees column in db)

---

```
1 def change
2   add_column :#{Table_name}, : mentionees, :integer, :default => 0
3 end
4
5 user. mentionees_count
```

---

## acts\_as\_mentionable Methods

Find out if an objects mentions

```
1 user.mentioned_by?(comment)
```

All mentioners

```
1 user.mentioners(Comment)
```

Number of mentioners (Requires mentioners\_count column in db)

```
1 def change
2   add_column :#{Table_name}, :mentioners_count, :integer, :default => 0
3 end
4
5 movie.mentioners_count
```

---

## Documentation

You can find the compiled YARD documentation at <http://rubydoc.info/github/cmer/socialization/frames>. Documentation for methods inside `include` blocks is not currently generated although it exists in the code. A custom YARD filter needs to be written for YARD to pick those up.

---

## Note on Patches/Pull Requests

- Fork the project.
- Make your feature addition or bug fix.
- Add tests for it. This is important so I don't break it in a future version unintentionally.
- Send me a pull request. Bonus points for topic branches.

---

## **Similar Projects**

acts\_as\_follower is a similar project that I only discovered when I was 95% finished writing the first version of Socialization. I initially intended to name this project acts\_as\_follower only to find out the name was taken. You might want to check it out as well so see which one suits your needs better. Socialization is simpler, supports “Likes” and “Mentions” and easilly extendable; acts\_as\_follower has more “Follow” features, however.

## **Copyright**

Copyright (c) 2012-2022 Carl Mercier – Released under the MIT license.