

Solving the Riddle of Search

Using Sphinx with Rails

My name is Pat

<http://freelancing-gods.com>

<http://twitter.com/pat>

Questions?

**Within Current
Context?**

Ask as we go

Jumping topics?

Please Wait

**A Tute in
Two Parts**

Part One: Core Concepts

**(ie: the
boring bits)**

**Part Two:
Beyond the
Basics**

**(ie: the
fun bits)**

So...

**Part One:
Core Concepts**

What is Sphinx?

Open Source Search Service

<http://www.sphinxsearch.com/>

**Similar to Ferret,
Solr, Xapian, etc**

Talks to MySQL and PostgreSQL

**... and Microsoft
SQL Server and
Oracle in 0.9.9**

Two Key Tools

indexer

**Guess what
it does**

**Talks to your
database**

**Files away
your data**

searchd

**No prizes for
guessing this
either.**

**searchd = search
daemon**

Runs in the Background

Handles Search Requests

**Does not talk to
your database.**

**What gets
indexed?**

Documents

**No, not Microsoft
Word docs**

**A document is a
single database
record**

**(ie: a Rails
model instance)**

Documents have
fields and
attributes

**They are not
the same.**

**THEY ARE NOT
THE SAME.**

Fields are...

Textual Data

Strings

Words to
search for.

**Attributes, on
the other hand...**

Integers

Floats

Timestamps

Booleans

**... and Strings.
Kinda.**

Strings as Ordinals

Huh?

**Integer values
indicating
alphabetical
order of strings.**

**Useless for
Searching?**

Useful for
Sorting though.

Search queries
don't look at
attribute values.

Sorting

Grouping

Filtering

Exact value matching

**So, that's
Sphinx...**

**But what's
Thinking
Sphinx?**

Ruby Library/ Rails Plugin

Hooks into ActiveRecord

Configure Sphinx with Ruby and YAMIL

Search with Sphinx in Rails

**What's in it for
me though?**

**Why should I
use Sphinx?**

**SQL does it all,
right?**

Well, maybe

But it gets messy

- **Find all people**
- **Where any word matches**
- **Part of any of four fields**

```
# given a query of 'Pat Allan'  
# assuming MySQL
```

```
SELECT * FROM  
WHERE first_name LIKE '%Pat%'  
      OR first_name LIKE '%Allan%'  
      OR last_name LIKE '%Pat%'  
      OR last_name LIKE '%Allan%'  
      OR location LIKE '%Pat%'  
      OR location LIKE '%Allan%'  
      OR profile LIKE '%Pat%'  
      OR profile LIKE '%Allan%';
```

It's a bit long...

**And that's a
simple query!**

**If we use
Thinking Sphinx**

```
# given a query of 'Pat Allan'
```

```
Person.search "Pat Allan",  
  :match_mode => :any
```

Much cleaner

**Can we start
coding now?**

Not quite.

Indexing

Fields

**What textual
data do we want
indexed?**

**Enter the
indexes
method**

```
class Person < ActiveRecord::Base
  # ...

  define_index do
    indexes first_name, last_name,
            location, profile
  end

  # ...
end
```

```
# Combine columns into one field
indexes [first_name, last_name],
  :as => :name
```

```
# Avoid core Ruby class methods
indexes :name
```

```
# Use association columns
indexes photos.captions,
  :as => :photos
```

Attributes

**What do we want
to sort and filter
by?**

**Our friend the
has
method**

```
class Person < ActiveRecord::Base
  # ...

  define_index do
    # ...
    has created_at, admin, email, :id
  end

  # ...
end
```

```
# Multi-Value Attributes (MVAs)  
has tags.id, :as => :tag_ids
```

**What if I want to
sort by a field?**

```
indexes first_name, last_name,  
        :sortable => true
```

**What does all
this code do?**

**Defines a SQL
query for Sphinx**

**Complexity
leads to slower
indexing**

**So don't forget
to add database
indexes as well!**

**Warning for fans
of fixtures...**

```
# In your define_index block:  
set_property(  
  :sql_range_step => 10_000_000  
)
```

**Because it's
SQL...**

**Can only use
model columns,
not model methods**

**Maybe that's not
good enough?**

**Write your
own SQL!**

```
has "YEAR(created_at)",  
    :as => :year, :type => :integer
```

**So, we've set up
our index.
Now what?**

```
rake thinking_sphinx:index
```

```
(in /Users/pat/Code/ruby/ts_test_2.2.0)
Generating Configuration to /Users/pat/Code/ruby/ts_test_2.2.0/
config/development.sphinx.conf
indexer --config /Users/pat/Code/ruby/ts_test_2.2.0/config/
development.sphinx.conf --all
Sphinx 0.9.8-release (r1371)
Copyright (c) 2001-2008, Andrew Aksyonoff

using config file '/Users/pat/Code/ruby/ts_test_2.2.0/config/
development.sphinx.conf'...
indexing index 'person_core'...
collected 1000 docs, 0.0 MB
collected 0 attr values
sorted 0.0 Mvalues, 100.0% done
sorted 0.0 Mhits, 100.0% done
total 1000 docs, 14436 bytes
total 0.191 sec, 75523.42 bytes/sec, 5231.60 docs/sec
indexing index 'person_delta'...
collected 0 docs, 0.0 MB
collected 0 attr values
sorted 0.0 Mvalues, nan% done
total 0 docs, 0 bytes
total 0.010 sec, 0.00 bytes/sec, 0.00 docs/sec
distributed index 'person' can not be directly indexed; skipping.
```

```
rake thinking_sphinx:start
```

```
(in /Users/pat/Code/ruby/ts_test_2.2.0)
searchd --pidfile --config /Users/pat/Code/ruby/ts_test_2.2.0/
config/development.sphinx.conf
Sphinx 0.9.8-release (r1371)
Copyright (c) 2001-2008, Andrew Aksyonoff

using config file '/Users/pat/Code/ruby/ts_test_2.2.0/config/
development.sphinx.conf'...
Started successfully (pid 6220).
```

```
rake thinking_sphinx:stop
```

```
(in /Users/pat/Code/ruby/ts_test_2.2.0)
```

```
Sphinx 0.9.8-release (r1371)
```

```
Copyright (c) 2001-2008, Andrew Aksyonoff
```

```
using config file '/Users/pat/Code/ruby/ts_test_2.2.0/config/  
development.sphinx.conf'...
```

```
stop: succesfully sent SIGTERM to pid 6220
```

```
Stopped search daemon (pid 6220).
```

```
rake ts:in
```

```
rake ts:start
```

```
rake ts:stop
```

**Can we
search yet?**

Person . search

Person.search "Pat Allan"

**Focus that
search on a
specific field**

```
Person.search("Ruby",  
  :conditions => { :name => "Pat Allan" }  
)
```

**Conditions are
not exactly like
ActiveRecord**

Strings only.

**And conditions
should always
be a Hash.**

Filtering on Attributes

```
Person.search("Ruby",  
  :with => { :admin => true }  
)
```

Filtering by Ranges

```
Person.search("Ruby",
  :with => {
    :created_at => 1.week.ago..Time.now
  }
)
```

Filtering by Arrays

```
Person.search("Ruby",  
  :with => { :tag_ids => [1, 2, 3] }  
)
```

**Matching all
Array values**

```
Person.search("Ruby",  
  :with_all => { :tag_ids => [1, 2, 3] }  
)
```

Excluding Filter Values

```
Person.search("Ruby",  
  :without => { :admin => true }  
)
```

Match Modes

: all

(The Default)

**All words must
exist in a
document**

: any

```
Person.search("Pat Allan",  
             :match_mode => :any  
            )
```

**One of the words
must exist in a
document**

: phrase

**The exact query
must exist in the
document**

:boolean

**Use boolean
logic in search
queries**

: extended

**Can match on
specific fields,
and apply
boolean logic**

Weighting

```
Person.search("Pat Allan",  
  :field_weights => {  
    "name"      => 100,  
    "profile"   => 50,  
    "location"  => 30  
  }  
)
```

```
define_index do
  # ...

  set_property :field_weights => {
    "name"      => 100,
    "profile"   => 50,
    "location"  => 30
  }
end
```

Sorting

: relevance

The Default

By Attributes

**Or Fields flagged
as :sortable**

: attr_asc

```
Person.search "Pat Allan",  
  :order => :created_at
```

: attr_desc

```
Person.search "Pat Allan",  
  :order      => :created_at,  
  :sort_mode => :desc
```

: extended

```
Person.search "Pat Allan",  
  :order => "last_name ASC,  
first_name ASC"
```

: expr

```
Person.search "Pat Allan",  
  :sort_mode => :expr,  
  :sort_by   => "@weight * ranking"
```

:time_segments

```
Person.search "Pat Allan",  
  :sort_mode => :time_segments,  
  :sort_by   => "created_at"
```

- **Last Hour**
- **Last Day**
- **Last Week**
- **Last Month**
- **Last 3 Months**
- **Everything Else**

**Sorted by
Segment, then
Relevance**

Multi-Model Searches

```
ThinkingSphinx::Search.search(  
  "Pat Allan"  
)
```

Pagination

Always On

```
Person.search "Pat Allan",  
  :page      => params[:page]
```

**You can request
really big pages
though.**

```
Person.search "Pat Allan",  
  :per_page => 1000,  
  :page      => params[:page]
```

If you want
massive pages...

```
Person.search "Pat Allan",  
  :page       => params[:page],  
  :per_page   => 1_000_000,  
  :max_matches => 1_000_000
```

```
# config/sphinx.yml
development:
  max_matches: 1000000
test:
  max_matches: 1000000
production:
  max_matches: 1000000
```

**Don't forget to
restart Sphinx**

Installing Thinking Sphinx

Using Git?

```
script/plugin install  
git://github.com/freelancing-god/  
thinking-sphinx.git
```

```
git clone git://github.com/  
freelancing-god/thinking-sphinx.git
```

**Want the gem
instead?**

```
sudo gem install  
  freelancing-god-thinking-sphinx  
  --source http://gems.github.com
```

```
# inside config/environment.rb
config.gem(
  "freelancing-god-thinking-sphinx",
  :version => "1.1.6",
  :lib      => "thinking_sphinx"
)
```

```
# at the end of your Rakefile:  
require 'thinking_sphinx/tasks'
```

Installing Sphinx

Windows?

Installer.
Piece of Cake.

***nix?**

**apt, yum, etc.
Should be
painless.**

OS X?

**Well... it can
be a little
complicated**

```
./configure
```

```
make
```

```
sudo make install
```

**With
PostgreSQL?**

```
./configure --with-pgsql=/usr/local/  
include/postgresql  
make  
sudo make install
```

```
pg_config --pkgincludedir
```

Or Use MacPorts

**... if you're using
it for MySQL or
PostgreSQL
as well**

THE #1 PROGRAMMER EXCUSE
FOR LEGITIMATELY SLACKING OFF:

"MY CODE'S COMPILING."

HEY! GET BACK
TO WORK!

COMPILING!

OH. CARRY ON.



**Anyone still
stuck?**

Examples in the Sample App

**git://github.com/freelancing-
god/sphinx-tute.git**

Set up Database

Hack as we go

Questions?

**Afternoon
Break?**

ARE YOU COMING TO BED?

) I CAN'T. THIS IS IMPORTANT.

WHAT?

|
SOMEONE IS WRONG ON THE INTERNET.



**Part Two:
Beyond the
Basics**

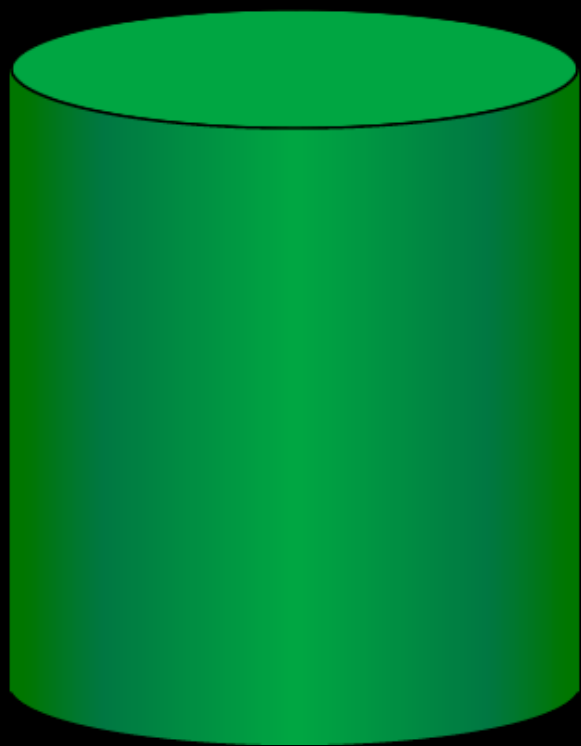
Delta Indexes

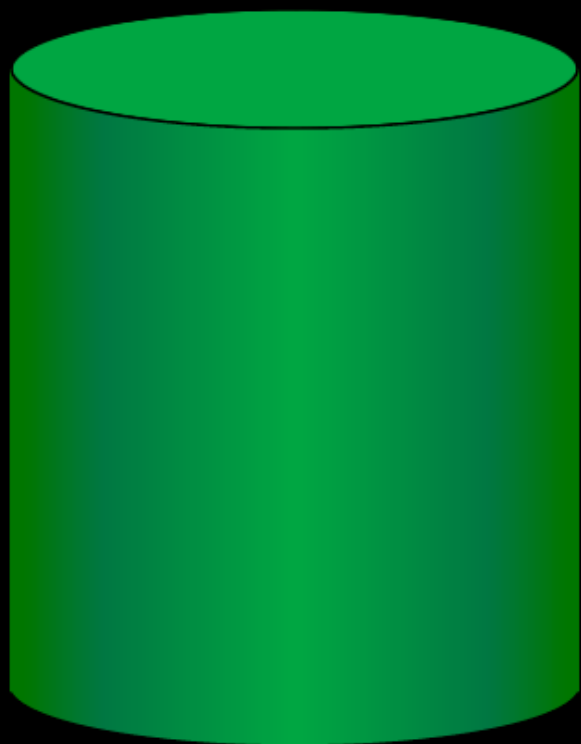
Why?

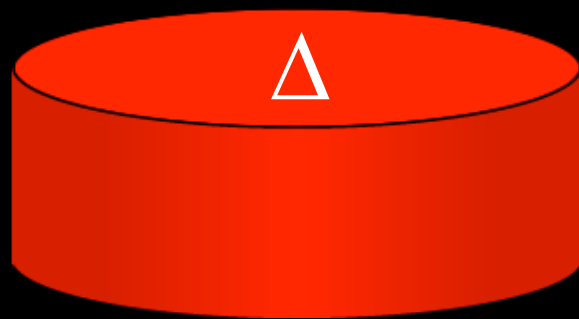
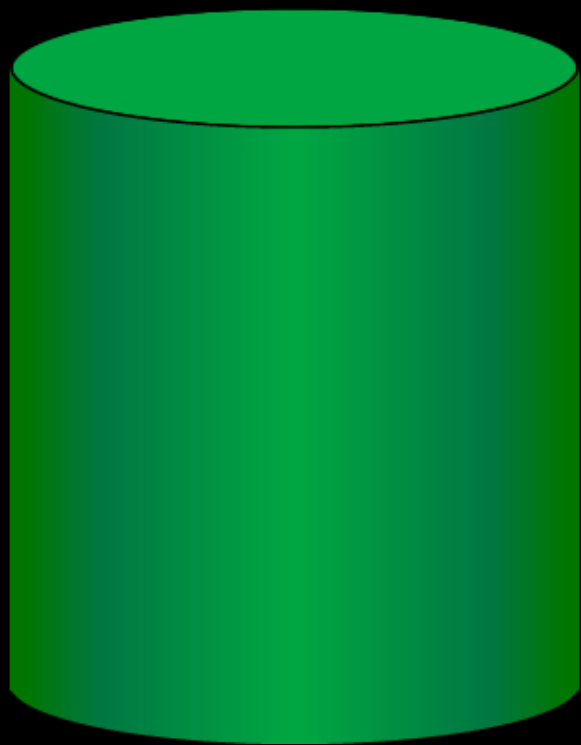
**No single-
document
updates**

**Can only process
a full index.**

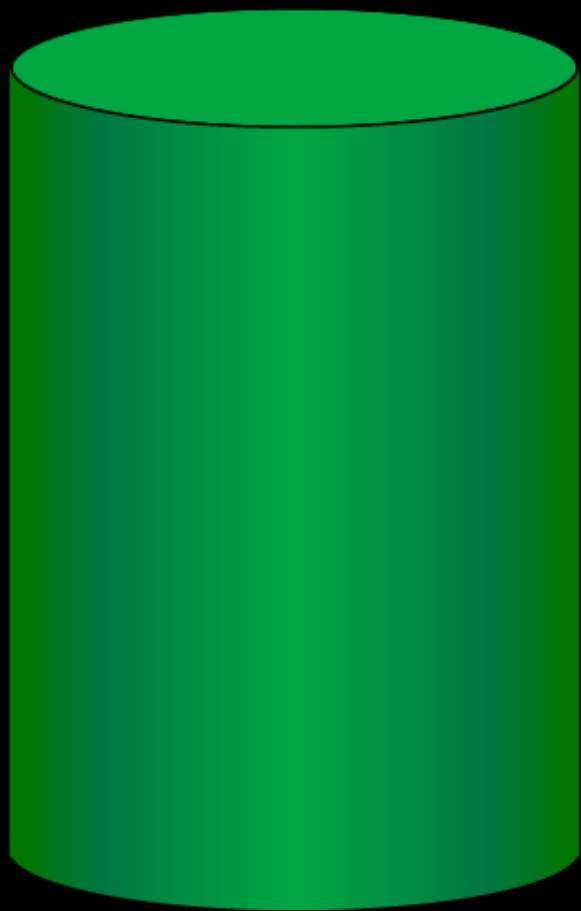
**Delta Index
holds the
changes**







```
rake thinking_sphinx:index
```



**What does this
mean?**

**We can track
changes as
they happen**

**... at the cost of a
little extra
overhead.**

How?

1. Add a boolean column 'delta' to your model.

**2. Tell your index
to use a delta.**

```
define_index do
```

```
  # ...
```

```
  set_property :delta => true
```

```
end
```

**3. Stop,
Re-index,
Restart.**

```
rake thinking_sphinx:stop  
rake thinking_sphinx:index  
rake thinking_sphinx:start
```

**... that's the
default
approach**

Datetime Deltas

```
define_index do
  # ...

  set_property :delta => :datetime,
               :threshold => 1.day
end
```

**Use existing
updated_at
column**

```
rake thinking_sphinx:index:delta
```

**Please Note: Not
entirely reliable**

Delayed Deltas

```
create_table :delayed_jobs do |t|  
  # ...  
end
```

```
define_index do
```

```
  # ...
```

```
  set_property :delta => :delayed
```

```
end
```

```
rake thinking_sphinx:delayed_delta
```

Uses delayed_job

**Removes
overheard from
your web stack**

Give it a Shot

Facets

Search Summaries

Ruby

Gemstones

Any Material Type

[Ruby](#) (6,869)

Metal Type

Any Metal Type

[Gold](#) (5,059)

[Silver](#) (980)

[Platinum](#) (86)

[Titanium](#) (46)

[Stainless Steel](#) (181)

[Other Metals](#) (214)

Brand

Any Brand

[Amazon.com Collection](#) (161)

[AsiaEXP](#) (17)

[Charles Winston](#) (1)

[Cheline](#) (7)

[Finejewelers](#) (411)

[Peora](#) (111)

[SilverSpeck.com](#) (35)

[Symbols in Silver](#) (1)

▼ Categories

[Jewelry & Watches](#) (31,410)

[Loose Diamonds &
Gemstones](#) (8,720)

[Fashion Jewelry](#) (11,253)

[Fine Jewelry](#) (5,593)

[Vintage & Antique Jewelry](#) (2,019)

[Loose Beads](#) (1,103)

[More](#) ▼

[Pottery & Glass](#) (2,628)

[Glass](#) (2,538)

[Pottery & China](#) (62)

[See all categories](#)

```
define_index do
  # ...
  indexes location, country,
    :facet => true

  has admin, :facet => true
end
```

```
Person.facets("Ruby")
```

```
{  
  :country => {  
    "Australia" => 15,  
    "USA"       => 11,  
    "Germany"  => 8,  
    "Canada"   => 12  
  },  
  :location => {  
    "Melbourne" => 3,  
    # ...  
  }  
}
```

```
@facets.for(:country => "Australia")
```

Give it a Shot

Extended Configuration

config/sphinx.yml

Just like
database.yml

Global Sphinx settings

**Common
Example:
Partial Matching**

Rai* == **Rails**

development:

enable_star: 1

min_infix_len: 1

test:

enable_star: 1

min_infix_len: 1

production:

enable_star: 1

min_infix_len: 1

**Stop,
Re-index,
Restart.**

Deployment

**Using
Capistrano?**

```
# config/deploy.rb
```

```
load 'vendor/plugins/thinking-sphinx/lib/  
thinking_sphinx/deploy/capistrano'
```

```
cap thinking_sphinx:configure
cap thinking_sphinx:index
cap thinking_sphinx:install:sphinx
cap thinking_sphinx:install:ts
cap thinking_sphinx:rebuild
cap thinking_sphinx:restart
cap thinking_sphinx:shared_sphinx_folder
cap thinking_sphinx:start
cap thinking_sphinx:stop
```

Installing Sphinx

**Set Sphinx index
location**

**Sphinx does not
like symlinks**

```
# config/sphinx.yml
production:
  searchd_file_path: "/var/www/
sphinx-tute/shared/db/sphinx/
production/"
```

```
# config/deploy.rb
after "deploy:setup",
    "thinking_sphinx:shared_sphinx_folder"
```

**Regular
indexing via
Cron**

Geo-Searching

**Need latitude
and longitude
attributes**

As floats

In Radians

```
define_index do
```

```
  # ...
```

```
  has latitude, longitude
```

```
end
```

**Stop,
Re-index,
Restart.**

```
Bar.search(  
  "Las Vegas",  
  :geo => [@lat, @lng],  
  :sort => "@geodist ASC"  
)
```

```
@bars.each_with_geodist do |bar,  
distance|  
  # distance is in metres  
end
```

**Using custom
column names?**

```
define_index do
  # ...

  set_property(
    :latitude_attr => :updown,
    :longitude_attr => :leftright
  )
end
```

**Geo-searching
across multiple
models?**

```
ThinkingSphinx::Search.search(  
  "pancakes",  
  :geo          => [@lat, @lng]  
  :latitude_attr => :lat,  
  :longitude_attr => :lng  
)
```

Give it a Shot

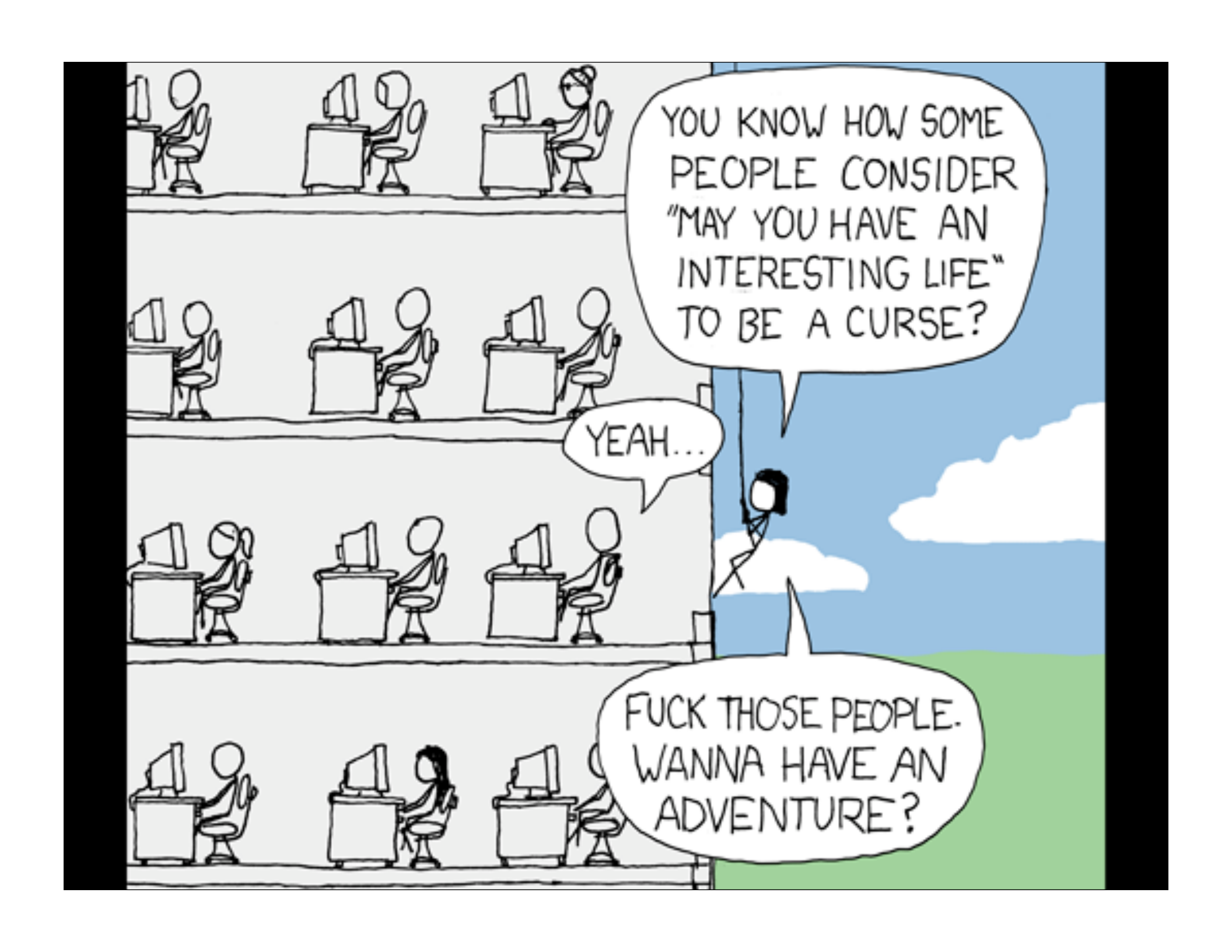
Are we done yet?

<http://ts.freelancing-gods.com>

**[http://groups.google.com/
group/thinking-sphinx](http://groups.google.com/group/thinking-sphinx)**

**[http://peepcode.com/products/
thinking-sphinx-pdf](http://peepcode.com/products/thinking-sphinx-pdf)**

Questions?

A cartoon illustration of a multi-story office building. On the right side, a person is climbing the exterior of the building. On the left side, four floors of office workers are visible, each sitting at a desk with a computer monitor. The background shows a blue sky with white clouds and a green field at the bottom.

YOU KNOW HOW SOME
PEOPLE CONSIDER
"MAY YOU HAVE AN
INTERESTING LIFE"
TO BE A CURSE?

YEAH...

FUCK THOSE PEOPLE.
WANNA HAVE AN
ADVENTURE?

Thank you.