

# ***Consuming REST services with ActiveSupport***

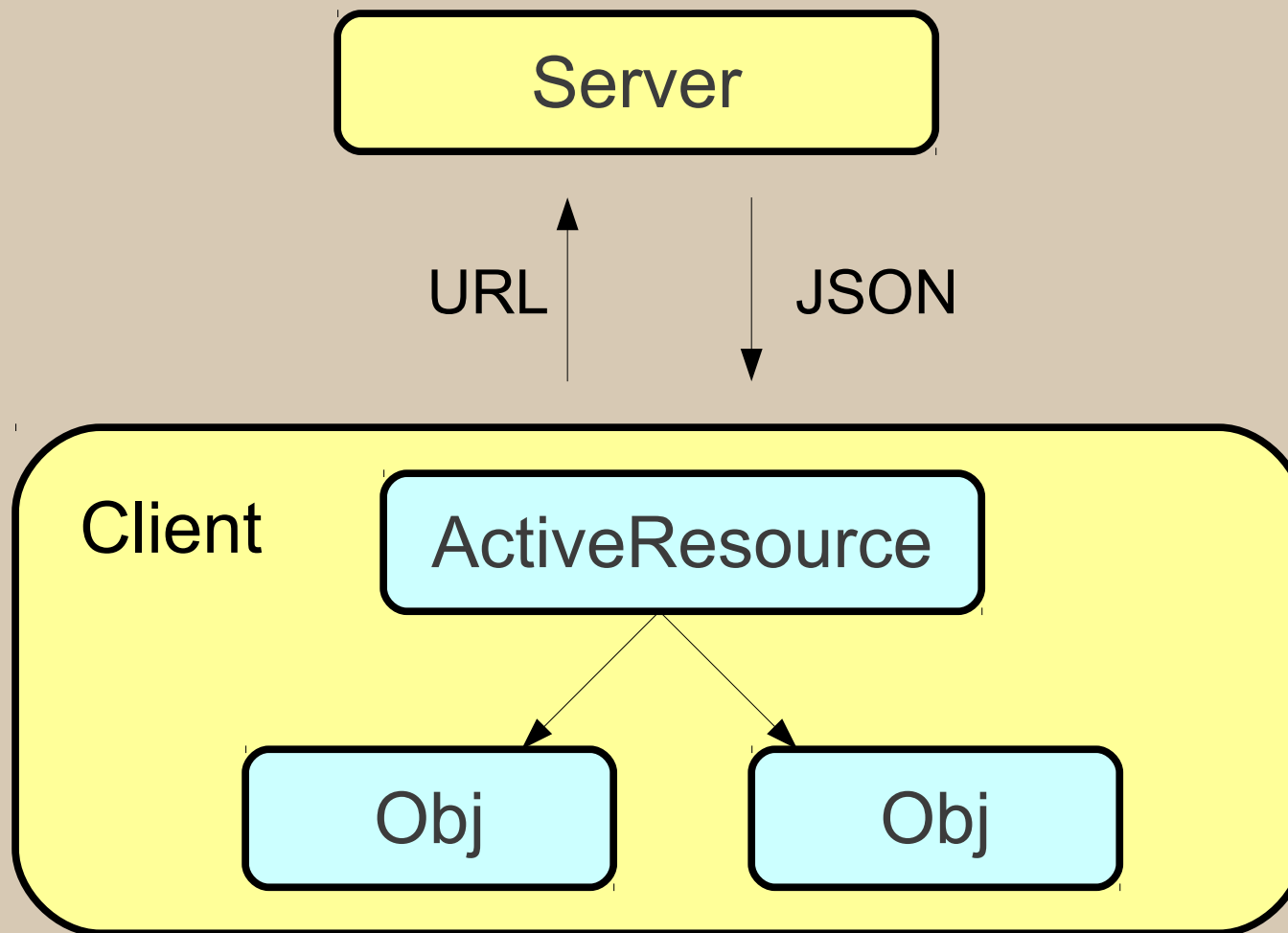
**Wolfram Arnold, @wolframarnold**  
**[www.rubyfocus.biz](http://www.rubyfocus.biz)**

# ActiveResource?

***Maps***

***Resources  
to  
Objects***

# ActiveResource



REST?

REpresentational  
State  
Transfer

Requests and responses are built around the transfer of **representations** of **resources**.

Individual resources are identified in requests, [...] using **URIs** in web-based REST systems.

Source: Wikipedia

# RESTful URL's Rails-style

map.resources :trips (in config/routes.rb)

GET	/trips	→ “index” action
POST	/trips	→ “create” action
GET	/trips/new	→ “new” action
GET	/trips/:id/edit	→ “edit” action with ID
GET	/trips/:id	→ “show” action with ID
PUT	/trips/:id	→ “update” action with ID
DELETE	/trips/:id	→ “destroy” action with ID

# Why it's hard?

- Opinionated
- Poorly documented
  - Read the sources
  - Use an editor that makes this easy
- Less mature than ActiveRecord
- Trial by fire

# ***DEMO***

***demo project:***

***git://github.com/wolframarnold/SFMeetup-Talk-on-ActiveResource-2011-12-08.git***

***server:***

***git://github.com/wolframarnold/where-have-you-been.git***  
***branch: sfmeetup\_2011\_12\_08***



# CRUD

<code>Trip.find(:all)</code>	→ GET /trips.json
<code>Trip.find(1)</code>	→ GET /trips/1.json
<code>trip = Trip.create(name: "...")</code>	→ POST /trips.json
<code>trip.name = "California"</code>	
<code>trip.save</code>	→ PUT /trips/1.json
<code>trip.delete</code>	→ DELETE /trips/1.json

# Declaration

```
class Trip < ActiveResource::Base
```

```
  self.site = "http://localhost:3000"
```

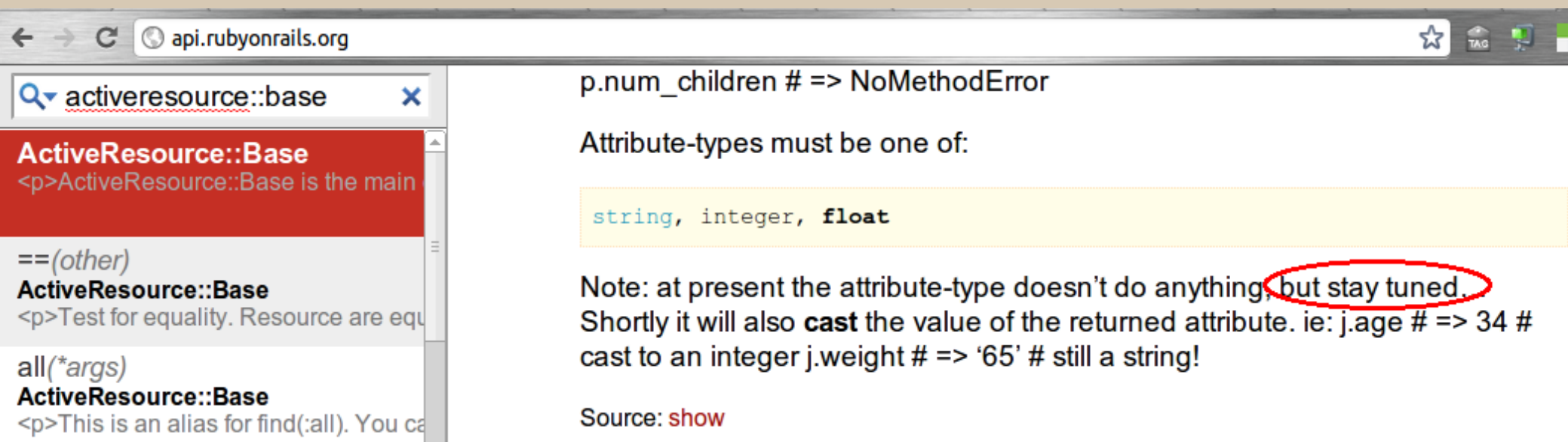
```
end
```

# Declaration

```
class Trip < ActiveResource::Base
  self.site = "http://localhost:3000"

  schema do
    string :name
  end
end
```

# Schema



The screenshot shows a web browser window at `api.rubyonrails.org` displaying the documentation for `ActiveResource::Base`. The left sidebar contains a search bar with `activeresource::base` and a list of methods including `==(other)`, `ActiveResource::Base`, and `all(*args)`. The main content area shows the error `p.num_children # => NoMethodError` and states that attribute types must be one of: `string, integer, float`. A note explains that the attribute type currently does nothing but will cast values in the future, with the phrase "but stay tuned" circled in red. Examples show `j.age` casting to an integer and `j.weight` remaining a string.

← → ↻ api.rubyonrails.org ☆ TAG

Q activeresource::base ✕

**ActiveResource::Base**  
 <p>ActiveResource::Base is the main

==(other)  
**ActiveResource::Base**  
 <p>Test for equality. Resource are equ

all(\*args)  
**ActiveResource::Base**  
 <p>This is an alias for find(:all). You ca

p.num\_children # => NoMethodError

Attribute-types must be one of:

`string, integer, float`

Note: at present the attribute-type doesn't do anything, but stay tuned.  
 Shortly it will also **cast** the value of the returned attribute. ie: j.age # => 34 #  
 cast to an integer j.weight # => '65' # still a string!

Source: [show](#)

# Schema

<a href="#">←</a> <a href="#">→</a> <a href="#">↻</a> <a href="#">GitHub, Inc. [US]</a> <a href="https://github.com/rails/rails/blame/master/activeresource/lib/active_resource/base.rb">https://github.com/rails/rails/blame/master/activeresource/lib/active_resource/base.rb</a>		
	311	<i># ActiveRecord types made be</i>
	312	<i># string, integer, float</i>
	313	<i>#</i>
	314	<i># Note: at present the at</i>
<a href="#">669c5eec</a> » <a href="#">josh</a>	315	<i># tuned...</i>
<a href="#">2009-12-20</a> <a href="#">Rename SchemaDefinition =&gt; Schema</a>		

# DateTime objects via JSON

config/initializer/active\_resource.rb

```
ActiveSupport::JSON::Encoding.  
  use_standard_json_time_format = true
```

```
ActiveSupport.parse_json_times = true
```

# Validations

- Local

```
class Trip < ActiveRecord::Base
  validates :name, :presence => true
end
```

- Remote

- Server issues 422 (“Unprocessable Entity”)
- `trip.errors[:name]` (theoretically)
- (Default Rails Scaffold omits “errors” key)
- Attribute-specific errors happen only if declared in schema

# Nested Resources

```
class Comment < ActiveResource::Base  
  self.prefix = "posts/:post_id/comments"  
end
```

```
Comment.find(100, post_id: 200)
```

→ /posts/200/comments/100.json



# Nested Resources != Associations

- You don't get:
  - `post.comments`
  - `comment.post`
- No Association methods
- No validations across parent/children

# RESTful Relationships???

- Individual controllers, flat data structure
  - /posts
  - /posts/111/comments
  - tough to enforce constraints across requests
- Single controller, nested data structure
  - /posts
  - post: {comments: [ {body: “blah”} ]}
  - lose 1:1 mapping object to resources

***ActiveResource***

***!=***

***ActiveRecord***

# ActiveRecord vs. ActiveResource

## “AR” (ActiveRecord)

- Tightly coupled to DB
- Maps relations well
- SQL strictly defined
- SQL drives consistency

## “ARes” (ActiveResource)

- API's vary wide and far
- No underlying standards
- “REST” loosely defined

# Limitations

Additional query parameters on create, update

```
Trip.create(name: "Canada")
```

```
→ "trip"=>{"name"=>"canada"}
```

# Limitations

Additional query parameters on create, update

```
Trip.create(name: "Canada",  
            auth_token: "mpohuZVe8dKD")
```

```
→ "trip"=>{"name"=>"canada",  
            "auth_token"=>"mpohuZVe8dKD"}
```

# Recommendations

- Good starting point for Rails REST servers
- Don't pretend it's ActiveRecord
- Informs design of API object adapters for non-Rails services
- Many gems define custom API-object mapping
- The Command Pattern