An Introduction to

{ json:api }

Zhuo-Fei Hui
@zfhui
Blinkist

Problem

RESTful APIs using JSON

Problem

RESTful APIs using JSON

We don't have a shared understanding about the structure.

Problem

Bikeshedding

Futile investment of time and energy in discussion of marginal technical issues.

– Wiktionary

Problem

Different clients prefer different structures.

Solution

{ json:api }

History

2013-05-03 Yehuda Katz released initial the draft

2013-07-21 media type application/vnd.api+json registration with IANA completed

2014-07-05 `v1.0rc` released

2015-05-29 `v1.0stable` released

Today `v1.1` still in draft

jsonapi.org

A Specification for Building APIs in JSON

jsonapi.org

A Specification for Building APIs in JSON

How a client should request for resources to be fetched or modified.

jsonapi.org

A Specification for Building APIs in JSON

How a client should request for resources to be fetched or modified.

How a server should respond to those requests.

A Simple Resource Object

A Simple Resource Object

```
User(id: integer, name: string)
```

Fetching a User

```
User(id: integer, name: string)
```

```
GET /users/1
```

Fetching a User

```
User(id: integer, name: string)
```

```
GET /users/1
```

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" }
  }
}
```

Fetching a User

```
User(id: integer, name: string)
```

```
GET /users/1
```

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" }
  }
}
```

Fetching a User

```
User(id: integer, name: string)
```

```
GET /users/1
```

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" }
  }
}
```

Fetching a User

```
User(id: integer, name: string)
```

```
GET /users/1
```

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" }
  }
}
```

Creating a User

```
POST /users
```

Creating a User

```
POST /users

{
  "data": {
    "type": "users",
    "attributes": { "name": "Yehuda Katz" }
  }
}
```

Creating a User

```
POST /users

{
  "data": {
    "type": "users",
    "attributes": { "name": "Yehuda Katz" }
  }
}
```

Creating a User

```
POST /users

{
  "data": {
    "type": "users",
    "attributes": { "name": "Yehuda Katz" }
  }
}
```

Creating a User

```
POST /users

{
  "data": {
    "type": "users",
    "attributes": { "name": "Yehuda Katz" }
  }
}
```

Updating a User

```
PATCH /users/2
```

Updating a User

```
PATCH /users/2

{
  "data": {
    "id": "2",
    "type": "users",
    "attributes": { "name": "Dan Gebhardt" }
  }
}
```

Updating a User

```
PATCH /users/2

{
  "data": {
    "id": "2",
    "type": "users",
    "attributes": { "name": "Dan Gebhardt" }
  }
}
```

Getting a List of Users

```
GET /users
```

GET /users


```
{
  "data": [
    {
      "id": "1",
      "type": "users",
      "attributes": { "name": "Steve Klabnik" }
    },
    {
      "id": "2",
      "type": "users",
      "attributes": { "name": "Yehuda Katz" }
    }
  ]
}
```

GET /users

```
{
  "data": [
    {
      "id": "1",
      "type": "users",
      "attributes": { "name": "Steve Klabnik" }
    },
    {
      "id": "2",
      "type": "users",
      "attributes": { "name": "Yehuda Katz" }
    }
  ]
}
```

GET /users

```
{
  "data": [
    {
      "id": "1",
      "type": "users",
      "attributes": { "name": "Steve Klabnik" }
    },
    {
      "id": "2",
      "type": "users",
      "attributes": { "name": "Yehuda Katz" }
    }
  ]
}
```

GET /users

```
{
  "data": [
    {
      "id": "1",
      "type": "users",
      "attributes": { "name": "Steve Klabnik" }
    },
    {
      "id": "2",
      "type": "users",
      "attributes": { "name": "Yehuda Katz" }
    }
  ]
}
```

Deleting a User

```
DELETE /users/2
```

1:n Relationship

1:n Relationship

```
User(id: integer, name: string)

Article(
  id: integer,
  title: string,
  content: text,
  user_id: integer
)
```

Fetching a User

```
GET /users/1
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /users/1


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

GET /articles/2


```
{
  "data": {
    "id": "2",
    "type": "articles",
    "attributes": {
      "title": "Intro to JSON API",
      "content": "Lorem opossum ..."
    },
    "relationships": {
      "user": {
        "data": { "id": "1", "type": "users" }
      }
    }
  }
}
```

GET /articles/2

```
{
  "data": {
    "id": "2",
    "type": "articles",
    "attributes": {
      "title": "Intro to JSON API",
      "content": "Lorem opossum ..."
    },
    "relationships": {
      "user": {
        "data": { "id": "1", "type": "users" }
      }
    }
  }
}
```

GET /articles/2

```
{
  "data": {
    "id": "2",
    "type": "articles",
    "attributes": {
      "title": "Intro to JSON API",
      "content": "Lorem opossum ..."
    },
    "relationships": {
      "user": {
        "data": { "id": "1", "type": "users" }
      }
    }
  }
}
```

Compound Documents

Instead of 3 requests:

`GET /users/1`

`GET /articles/2` and `GET /articles/5`

We can do 1 request:

`GET /users/1?include=articles`

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles.title


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API", "content": "Lorem opossum ..." }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding", "content": "Marsupial fur trees ..." }
      }
    ]
  }
}
```

GET /users/1?include=articles.title


```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    },
    "included": [
      {
        "id": "2",
        "type": "articles",
        "attributes": { "title": "Intro to JSON API" }
      },
      {
        "id": "5",
        "type": "articles",
        "attributes": { "title": "Anti-Bikeshedding" }
      }
    ]
  }
}
```

Creating a User with Relationships

```
POST /users
```

POST /users

```
{
  "data": {
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

POST /users

```
{
  "data": {
    "type": "users",
    "attributes": { "name": "Steve Klabnik" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "2", "type": "articles" },
          { "id": "5", "type": "articles" }
        ]
      }
    }
  }
}
```

☝ Updating a User with Relationships

```
PUT /users/1
```

☝ PUT /users/1

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Dan Gebhardt" },
    "relationships": {
      "articles": {
        "data": [
          { "id": "3", "type": "articles" },
          { "id": "6", "type": "articles" }
        ]
      }
    }
  }
}
```

☝ PUT /users/1

```
{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Dan Gebhardt" },
    "relationships": {
      "articles": {
        "data": []
      }
    }
  }
}
```

```
PUT /users/1

{
  "data": {
    "id": "1",
    "type": "users",
    "attributes": { "name": "Dan Gebhardt" }
  }
}
```

Manipulating relationships

Use POST and DELETE!

Adding Relationships to a User

Adding Relationships to a User

```
POST /users/1/relationships/articles
```

Adding Relationships to a User

```
POST /users/1/relationships/articles

{
  "data": [
    { "id": "3", "type": "articles" },
    { "id": "7", "type": "articles" }
  ]
}
```

Deleting a User's Relationships

```
DELETE /users/1/relationships/articles
```

```
{
  "data": [
    { "id": "3", "type": "articles" },
    { "id": "7", "type": "articles"}
  ]
}
```

Things we didn't discuss

- error objects, meta objects, links objects
- n:m relationships
- pagination, sorting, filtering, sparse fieldset
- does not support creating nested resources

To summarise ...

- ultimate anti-bikeshedding tool
- one endpoint to serve different client needs
- tight coupling between the data and underlying data structure
- leverages HTTP content negotiation mechanism

Implementation with JSONAPI::Resources

- most compliant with { json:api }
- maintained by Larry Gebhardt
- relies heavily on Active Record Models
- currently at `v0.9` and `v0.10beta`

Create your new Rails API

```
rails new my-new-json-api --api

rails g scaffold user name:string

rails g scaffold article title:string content:text

rails db:setup
```

Add `gem "jsonapi-resource"` to your Gemfile

```
bundle install
```

Generate JSONAPI::Resources

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
```

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
  create  app/resources/user_resource.rb
```

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
  create  app/resources/user_resource.rb

class UserResource < JSONAPI::Resource; end
```

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
  create  app/resources/user_resource.rb

class UserResource < JSONAPI::Resource



end
```

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
  create  app/resources/user_resource.rb

class UserResource < JSONAPI::Resource
  attribute :name


end
```

Generate JSONAPI::Resources

```
rails g jsonapi:resource user
  create  app/resources/user_resource.rb

class UserResource < JSONAPI::Resource
  attribute :name
  has_many :articles,
           always_include_linkage_data: true
end
```

```ruby
# app/resources/article_resource.rb

class ArticleResource < JSONAPI::Resource
  attributes :title, :content
  has_one :author,
          always_include_linkage_data: true,
          foreign_key: :user_id
end
```

```ruby
# app/resources/article_resource.rb

class ArticleResource < JSONAPI::Resource
  attributes :title, :content
  has_one :author,
          always_include_linkage_data: true,
          foreign_key: :user_id
end
```

```ruby
# config/routes.rb

Rails.application.routes.draw do
  resources :articles
  resources :users
end
```

```ruby
# config/routes.rb

Rails.application.routes.draw do
  resources :articles
  resources :users
end
```

```ruby
# config/routes.rb

Rails.application.routes.draw do
  jsonapi_resources :articles
  jsonapi_resources :users
end
```

```ruby
# app/controllers/application_controller.rb

class ApplicationController < ActionController::API

end
```

```ruby
# app/controllers/application_controller.rb

class ApplicationController < ActionController::API
  include JSONAPI::ActsAsResourceController
end
```

```ruby
# app/controllers/users_controller.rb

class UsersController < ApplicationController
  def index
    ...
  def create
    ...
  def update
    ...
    ...
end
```

```ruby
# app/controllers/users_controller.rb

class UsersController < ApplicationController; end
```

```ruby
# app/controllers/users_controller.rb

class UsersController < ApplicationController; end


# app/controllers/articles_controller.rb

class ArticlesController < ApplicationController; end
```


You've just created a JSON API!

Where to go from here?

Only Readable API: `fast_jsonapi` / `active_model_serializers`

Tests: `jsonapi-rspec`

Authorization: `jsonapi-authorization`

Client: `json-api-client` / `jsonapi-consumer`

jsonapi-suite.org

A collection of ruby libraries that facilitate the jsonapi.org specification.

- serialiser and deserialiser for { json:api }
- spec helpers
- helpers for swagger documentation

References

Talk JSON API: convention driven API design by Steve Klabnik

Talk Past, Present and Future of JSON API by Steve Klabnik

Website Media Type Specs

Talk "The JSON API Spec" by Marco Otto-Witte

Talk "Pragmatic JSON API Design" by Jeremiah Lee

Podcast "Dan Gebhard - json-api, jsonapi-resources, orbit.js & Ember Data" by Byle Daigle

Podcast "Data Loading Patterns with the JSON API with Balint Erdi" by The Frontside Podcast

Thanks!

```ruby
# app/resources/user_resource.rb

class UserResource < JSONAPI::Resource
  attributes :name, :first_name, :last_name

  has_many :articles, always_include_linkage_data: true

  def name
    "#{@model.first_name} #{@model.last_name}"
  end

  def fetchable_fields
    %i(name articles)
  end

  def self.creatable_fields(_context)
    %i(first_name last_name articles)
  end

  def self.updatable_fields(_context)
    %i(first_name last_name articles)
  end
end
```

Yehuda Katz

{ json:api } is a wire protocol for incrementally fetching and updating a graph over HTTP.

https://twitter.com/wycats/status/925847356532125696

json:api vs GraphQL

GraphQL is protocol agnostic

JSON API leverages HTTP functionalities, such as:

json:api vs GraphQL

```
GET /users/1` HTTP/1.1
Accept: application/vnd.api+json

HTTP/1.1 200 OK
Content-Type: application/vnd.api+json
ETag: "bf3291afe28105e12b9ff5941a3cf6d7"
```

json:api vs GraphQL

```
GET /users/1` HTTP/1.1
Accept: application/vnd.api+json
If-None-Match: "bf3291afe28105e12b9ff5941a3cf6d7"

HTTP/1.1 305 Not Modified
```