from: https://docs.google.com/document/d/1AzQPHCdbJjsPlUQcNqrSD_7tiPLVirXZCmFw0D9XqSM/edit?usp=sharing

- https://github.com/rails-api/active_model_serializers/blob/master/docs/ARCHITECTURE.md

JSON API

https://github.com/rails-api/active_model_serializers/issues/1098
https://github.com/rails-api/active_model_serializers/pull/1301

JSON API Requests

- Query Parameters Spec

Headers:

- Request: Accept: application/vnd.api+json
- Response: Content-Type: application/vnd.api+json

Fetching Data

A server MUST support fetching resource data for every URL provided as:

- a `self` link as part of the top-level links object
- a `self` link as part of a resource-level links object
- a `related` link as part of a relationship-level links object

Example supported requests

- Individual resource or collection
    - GET /articles
    - GET /articles/1
    - GET /articles/1/author
- Relationships
    - GET /articles/1/relationships/comments
    - GET /articles/1/relationships/author
- Optional: Inclusion of related resources `ActiveModel::Serializer::IncludeTree`
    - GET /articles/1?`include`=comments
    - GET /articles/1?`include`=comments.author
    - GET /articles/1?`include`=author,comments.author
    - GET /articles/1/relationships/comments?`include`=comments.author
- Optional: Sparse Fieldsets `ActiveModel::Serializer::Fieldset`
    - GET /articles?`include`=author&`fields`[articles]=title,body&`fields`[people]=name
- Optional: Sorting
    - GET /people?`sort`=age
    - GET /people?`sort`=age,author.name
    - GET /articles?`sort`=-created,title
- Optional: Pagination
    - GET /articles?`page`[number]=3&`page`[size]=1

- Optional: Filtering
    - GET /comments?filter[post]=1
    - GET /comments?filter[post]=1,2
    - GET /comments?filter[post]=1,2

CRUD Actions

Asynchronous Processing

Bulk Operations Extension

JSON API Schema

JSON API Schema Documents

JSON API document	**JSON API properties**	**Required**	**notes**
schema	oneOf (success, failure)	yes	

success	☐ data: `"$ref": "#/definitions/data"` ☐ included: array of unique items of type `"$ref": "#/definitions/resource"` ☐ meta: `"$ref": "#/definitions/meta"` ☐ links: allOf ☐ link: `"$ref": "#/definitions/links"` ☐ pagination: `"$ref": "#/definitions/pagination"` ☑ jsonapi: `"$ref": "#/definitions/jsonapi"`	data	
failure	☐ errors: array of unique items of type `"$ref": "#/definitions/error"`	errors	

	☐meta: `"$ref": "#/definitions/meta"` ☐ jsonapi: `"$ref": "#/definitions/jsonapi"`		

JSON API Schema Definitions

JSON API definition name	**JSON API definition value**	**Required**	**notes**
`#/definitions/`meta	Object		Non-standard meta-information that can not be represented as an attribute or relationship.
`#/definitions/`included	UniqueArray(resource)		To reduce the number of HTTP requests, servers **MAY** allow responses that include related resources along with the requested primary resources. Such responses are called "compound documents".

`#/definitions/`data	oneOf(resource, UniqueArray(resource)		The document's "primary data" is a representation of the resource or collection of resources targeted by a request. Collection: one of an array of resource objects, an array of resource identifier objects, or an empty array ([]), for requests that target resource collections.
`#/definitions/`resource	Object properties: String(type), String(id), attributes, relationships, links, meta	type, id	"Resource objects" appear in a JSON API document to represent resources
`#/definitions/`links	Object properties: Uri(self) and/or		A resource object **MAY** contain references to other resource objects ("relationships"). Relationships may be to-one or to-many.

	Link(related), allows additional		Relationships can be specified by including a member in a resource's links object. A `self` member’s value is a URL for the relationship itself (a "relationship URL"). This URL allows the client to directly manipulate the relationship. For example, it would allow a client to remove an `author` from an `article` without deleting the people resource itself.
`#/definitions/`link	oneOf (linkString, linkObject)		A link **MUST** be represented as either: a string containing the link's URL or a link object."
link.linkString	String: Uri		A string containing the link's URL.
link.linkObject	Object: Uri(href), meta	href	

`#/definitions/`attributes	patternProperties: ^(?!relationships$\|links$)\\w[-\\w_]*$		Members of the attributes object ("attributes") represent information about the resource object in which it's defined. Attributes may contain any valid JSON value
`#/definitions/`relationships	patternProperties: ^\\w[-\\w_]*$" Links, relationships.data, meta		Members of the relationships object ("relationships") represent references from the resource object in which it's defined to other resource objects."
relationships.data	oneOf (relationshipToOne, relationshipToMany)		Member, whose value represents "resource linkage"
`#/definitions/`relationshipToOne	anyOf(empty, linkage)		References to other resource objects in a to-one ("relationship"). Relationships can be specified by including a member in a resource's links object.

`#/definitions/`relationshipToMany	UniqueArray(linkage)		An array of objects each containing "type" and "id" members for to-many relationships
`#/definitions/`empty	null		Describes an empty to-one relationship.
`#/definitions/`linkage	String(type), String(id), meta	type, id	The "type" and "id" to non-empty members.
`#/definitions/`pagination	pageObject(first), pageObject(last), pageObject(prev), pageObject(next)		
pagination.pageObject	oneOf(Uri, null)		The <x> page of data
`#/definitions/`jsonapi	String(version), meta		An object describing the server's implementation
`#/definitions/`error	String(id), links, String(status),		id: A unique identifier for this particular occurrence of the problem.

	String(code), String(title), String(detail), error.source, meta		status: The HTTP status code applicable to this problem, expressed as a string value code: An application-specific error code, expressed as a string value. title: A short, human-readable summary of the problem. It **SHOULD NOT** change from occurrence to occurrence of the problem, except for purposes of localization. detail: A human-readable explanation specific to this occurrence of the problem."
error.source	String(pointer), String(parameter)		parameter: A string indicating which query parameter caused the error
pointer	JSON Pointer RFC6901		A JSON Pointer [RFC6901] to the associated entity in the request document [e.g. "/data" for a primary data object, or "/data/attributes/title" for a specific attribute]

The http://jsonapi.org/schema makes a nice roadmap.

1. TODO: Use schema/schema.json to validate and document our implementation
    - #1270
    - #1162
    - maybe use prmd? http://www.sitepoint.com/document-your-json-api-schema-with-prmd/
2. TODO: use uri_template in link generation? use json pointer gem?
    - #1282 (comment)
    - > I believe what you want here is a uri_template. See https://github.com/hannesg/uri_templateSee e.g. https://developer.github.com/v3/#schema http://tools.ietf.org/html/rfc6570 see e.g.https://api.github.com/ (though not a JSON API, is an example of usage)

JSON API to definition to class/module/method mapping

For brevity, let

- `AMS` refer to `ActiveModelSerializers`
- `AM::S` refer to `ActiveModel::Serializer`
- `AM::S::A` refer to `ActiveModel::Serializer::Adapter`

Adding serializers https://github.com/rails/rails/pull/21496

Current	Desired	Notes

require 'active_model_serializers'	require 'active_model_serializers' vs. require ‘active_model/serializers’	
ActiveModelSerializers	ActiveModelSerializers vs. ActiveModel::Serializers	
ActiveModel::SerializableResource	ActiveModelSerializers::SerializableResource vs. ActiveModel::SerializableResource vs. ActiveModel::Serializers::Serializa	

	bleResource	
namespace ActiveModel::Serializer	AMS::Current, ActiveModelSerializers::V08, etc vs. ActiveModel::Serializers	versioned
class AM::S	AMS::Current::Resource vs. AMS::Resource vs. AM::Serializers::Serializer vs. AM::Serializers::Resource	is a class that defines the properties and behavior of the data that you present to the user. Consider distinguishing ActiveModel vs. ActiveRecord resource
AMS::Model	AMS::Model vs.	e.g. ActiveRecord object but a PORO. Maybe should be AMS::Record A record is an instance of a model that contains data loaded from a server. Your application

	ActiveModel::Serializers::Model	can also create new records and save them back to the server.
AM::S::Lint	AMS::Model::Lint	
ActiveModel::Serialization	AMS::ActionController::Serialization	
AM::S::CollectionSerializer	AMS::Current::ResourceCollection	
namespace AM::S::Adapter	AMS::Adapter	adapter: knows how to talk to your server. knows how to translate requests from AMS into requests on your server. object that translates requests from AMS/Rails/Grape etc (such as "find the user with an ID of 123") into a requests to a server. let you completely change how your API is implemented without impacting your application code.
class AM::S::A::JsonApi	AMS::Adapter::JsonApi, AMS::Adapter::V08::Json	not versioned

class ActiveModel::Serializer, AM::S::A::Attributes	AMS::Formatter	maps keys and values to desired format, seehttps://github.com/orbitjs/orbit.js/blob/master/lib/orbit-common/jsonapi-serializer.js
N/A	AMS::Current::Store	store is the central repository of records in your application, use the store to retrieve records, as well to create new ones. The store will automatically cache records for you.
N/A	AMS::Adapter::JsonApi::Exceptions	
AM::S::A::JsonApi::ApiObject::JsonApi	AMS::Adapter::JsonApi::ApiObject:: {JsonApi,Links,Link,Resource,Error, etc}? vs. JsonApiFormat or even schemavalidation	
AMS.logger	AMS.logger vs.	

	AM::Serializers.logger	
AM::Serializer.config	AM::Serializers.config vs. AMS.config	

How versioned releases might look

```
# lib/active_model_serializers/v08/serializer.rb
# lib/active_model_serializers/v08/collection_serializer.rb
# lib/active_model_serializers/v08/adapter.rb
module ActiveModelSerializers::V08
  # Serializer might not need anything, but good to have defined
  Serializer = ActiveModel::Serializer
  CollectionSerializer = ActiveModel::Serializer::CollectionSerializer # well, it's Array now, but hopefully we'll get that renaming pr soon
  class Adapter < ActiveModel::Serializer::Adapter::Base
    # whatever methods you need
  end
end
```

And test like the other serializers and adapters

```
adapter = ActiveModelSerializers::V08::Adapter
serializer = ActiveModelSerializers::V08::CollectionSerializer
each_serializer = ActiveModelSerializers::V08::Serializer
collection_resource = Post.all
expected_response = [{ something }]
assert ActiveModel::SerializableResource.new(collection_resource, serializer: serializer, each_serializer: each_serializer, adapter:
adapter).as_json, expected_response

# and

adapter = ActiveModelSerializers::V08::Adapter
serializer = ActiveModelSerializers::V08::Serializer
resource = Post.first
expected_response = { something }
assert ActiveModel::SerializableResource.new(colle
```

Class Naming and Nesting

ref: https://github.com/rails-api/active_model_serializers/pull/1310#issuecomment-154956802
related:

- https://github.com/rails-api/active_model_serializers/pull/1310#discussion_r43813265
    - https://github.com/rails-api/active_model_serializers/pull/1310#issuecomment-153867830
    - https://github.com/rails-api/active_model_serializers/pull/1310#issuecomment-153892764
- https://github.com/rails-api/active_model_serializers/issues/1298
- https://github.com/rails-api/active_model_serializers/pull/1301/files#r42963185

see

- http://guides.emberjs.com/v2.1.0/models/
- https://github.com/cerebris/jsonapi-resources/blob/master/lib/jsonapi/formatter.rb
- https://andycrum.github.io/ember-data-model-maker/
- https://github.com/cerebris/jsonapi-resources#jsonapiresource
- origins of naming in AMS:
    - rails/rails@d2b78b3#diff-80d5beeced9bdc24ca2b04a201543bdd
    - rails/rails@783db25
    - rails/rails@c6bc8e6
    - rails/rails@fbdf706#diff-531b52ff1432c7c9f9472188b39d6a43
    - rails/rails@4bfbdc1
    - rails/rails@8896b4f
    - https://github.com/rails-api/active_model_serializers/commits/master?page=19
    - 729a823#diff-fe7aa2941c19a41ccea6e52940d84016
    - 4a2d985
    - f00fe55

The initial name of the class in Rails was ActiveModel::Serializers -> ActiveModel::Serializer and the gem was originally named by **@steveklabnik** as ActiveModelSerializers, then renamed toActiveModel::Serializer, then referred to as ActiveModel::Serializers, though the gem name didn't change, and recently we started thinking about what the name of the gem should be

Now, Rails has ActiveModel::Serializers::JSON and ActiveModel::Serialization all of which makes Benjamin think we're better off taking advantage of the gem already having a different name and prefer to use it. For example, he asks, what would you call ActiveModelSerializers::Model? ActiveModel::Serializer::Model? (Not to mention that using a class as a namespace causes load-order issues).

Misc links:

https://github.com/rails-api/active_model_serializers/labels/Mandatory
https://github.com/rails-api/active_model_serializers/issues/1235

https://medium.com/@joaomdmoura/the-future-of-ams-e5f9047ca7e9
> https://github.com/rails/rails/pull/19832/files#diff-0e72f2d0ba32bae788816a4f7e59bc34R24
> https://github.com/rails/rails/pull/21496
> https://github.com/rails-api/active_model_serializers/pull/1310

ref:
- https://amserializers.slack.com/archives/core-maintainers/p1447255805000002
- https://amserializers.slack.com/archives/core-maintainers/p1446693705000002
- https://twitter.com/dgeb/status/662091767446159360
- https://github.com/rails-api/active_model_serializers/issues/1236
- https://github.com/rails-api/active_model_serializers/pull/1322
- https://github.com/rails/rails/pull/19832/files#diff-70cc8da393475ab7f80a5625887b0dc9R1
- https://github.com/rails/rails/commit/2daac47d585c5b8f37e4749d6a9a3aea4b989bd0
- https://github.com/rails/rails/pull/21496
- https://github.com/rails-api/active_model_serializers/pull/1301#issuecomment-156538885

```
ction_resource, serializer: serializer, adapter: adapter).as_json, expected_response
```

refs:

- Rename ArraySerializer to CollectionSerializer #1251
- Describe AMS architecture in the big picture #1253

- RFC: Name-spacing of objects under (module) ActiveModelSerializers vs. (core class) ActiveModel::Serializer including where the logger should be defined #1298
- Document ActiveModelSerializers::Model #1283 and #1296 #1296

Keeping a well-defined interface, e.g. grape support: #1258 and serialization_context #1289

'm thinking of modeling naming after more complete (and well-informed) JSON API libraries such as

- generally http://jsonapi.org/implementations/
- https://github.com/cerebris/jsonapi-resources/blob/master/lib/jsonapi/resource.rbhttps://github.com/cerebris/jsonapi-resources#jsonapiresource
- ember-data (https://github.com/emberjs/data/blob/3930e457f36a0bbdde0155c1b98ebe976e607358/packages/ember-data/lib/adapters/json-api-adapter.js vs.https://github.com/emberjs/data/blob/3930e457f36a0bbdde0155c1b98ebe976e607358/packages/ember-data/lib/serializers/json-serializer.js)
- https://github.com/endpoints/endpoints/blob/master/es5/format-jsonapi/index.jshttps://github.com/endpoints/endpoints/blob/master/es5/validate-json-schema/index.js
- https://github.com/orbitjs/orbit.js/blob/master/lib/orbit-common/schema.jshttps://github.com/orbitjs/orbit.js/blob/master/lib/orbit-common/jsonapi-serializer.js
-

Other

`ActiveModelSerializers:: SerializationContext`

BF Asks: why is ActionController::Renderers::RENDERERS changed from a hash to a set https://github.com/rails/rails/commit/a4c04a43cc96d9c94b7a5ecd5c7458fdc2f8617f (original works was in the AMS pr)

Here is a list of stuff we should probably discuss internally before getting feedback from the community.

- Clarify the situation with options (which belong to the adapter, the serializer, or the serializable_hash method)
  - Details:
    - Given: render json: object, options
    - SerializableResource defines `ADAPTER_OPTION_KEYS = Set.new([:include, :fields, :adapter])`
    - And then partitions the options: @adapter_opts, @serializer_opts =
    - options.partition { |k, _| `ADAPTER_OPTION_KEYS.include? k }.map { |h| Hash[h] }`
    - So, that: `@serializer_instance ||= serializer.new(resource, serializer_opts)`
    - And: `@adapter ||= ActiveModel::Serializer::Adapter.create(serializer_instance, adapter_opts)`
    - i.e. **all** options go to the serializer unless reserved for the Adapter.
      - I (Benjamin) wouldn't mind making the Serializer initializer options explicit, as well.
    - And finally: `delegate :serializable_hash, :as_json, :to_json, to: :adapter`
    - Where `serializable_hash` is part of the (Rails) ActiveModel::Serialization interface used by (Rails) ActiveModel::Serializers::Json
      - And then called in Rails by the Json Renderer during `render_to_body(options)`
      - So, the options passed to `` `serializable_hash` `` are the options passed to render and called on e.g. `json = json.to_json(options) unless json.kind_of?(String)`
      - ref: https://github.com/rails/rails/pull/21496
  - Concerns / Usability:
    - Serializers don't serialize, Adapters do. SerializableResource.new at least behaves like Serializers used to in 0.8
    -
- Reforge the test infrastructure
  - ActiveModel::Serializer::Lint::Tests
  - bf4 https://github.com/rails-api/active_model_serializers/pull/1099#discussion_r39120220
  - bf4 https://github.com/rails-api/active_model_serializers/pull/967
  - JSON API schema validation https://jsonapi.org/schema
  - Validate against a given schema, e.g. prmd http://www.sitepoint.com/document-your-json-api-schema-with-prmd/?

- Clarify the deserialization situation
    - Given a hash of serialized resource, (JSON/JSON API?), for each root: resource in the hash, infer the model from the root and instantiate model(s) using the associated resource(s)
    - Q: no model can be inferred?
    - Q: invalid attributes?
    - Q: model is not active record?
    -

Should we discuss anything at:

- https://groups.google.com/forum/#!forum/rails-api-core ? It's more topically threaded than slack