HTTP API Design for iOS Applications
@vanstee
Big Nerd Ranch
• Modeling Resources
• Tools: Server and Client
• Real world problems
• Future
Modeling Resources
The key abstraction of information in REST is a resource.
A resource is a conceptual mapping to a set of entities, not the entity that corresponds to the mapping at any particular point in time.
Resources are not just database records
Resources are the **nouns**.
HTTP methods are the **verbs**.
URIs are the **identifiers**.
Media types are the **representations**.
But what about transactions? searches? complex actions?
Don’t do this:
POST /accounts/1/transfer/500.00/to/2

Try this instead:
POST /transactions

{ “from”: 1, “to”: 2, “amount”: 500.00 }
Tools
Server-side

- Rails
- Active Model Serializers
- Custom Responders
- rack-test and json_spec
Client-side

- AFNetworking
- RestKit (if you really need it)
- VCRURLConnection and mitmproxy
Real World Problems
Versioning

Don’t do this:

POST /v1/users/1

Try this instead:

POST /users/1

Accept: application/json; version=1.0
Authentication

- OAuth2 with API routes for token generation
- NSURLConnection supports cookies
- Basic Authentication over HTTPS*
Caching

- NSURLCache has support for **Cache-Control** and **ETags**
- AFNetworking supports this by default
- Rails gives you these for free
Smarter Requests

- Side loading associated resources
- HTTP Pipelining for GET, HEAD, PUT, and DELETE requests
- HTTP compression
Future
HTTP 2.0

- Based on SPDY
- Multiplexing
- Server Push
- Better compression
JSON API

- Standard hypermedia type
- Always namespaced
- Always returns collections for easy parsing
- Support for batch operations
JSON Patch

- Standard hypermedia type for updating records
- Easily handle associations
- Send minimal amount of information
Thanks

blog.steveklabnik.com
designinghypermediaapis.com
afnetworking.com
jsonapi.org