RESTful API Design

6.170 Recitation 7
Recall: big ideas of REST

Data abstraction more powerful than functional abstraction
Recall: big ideas of REST

The *type* of resource dictates the actions that can be done on it
Recall: big ideas of REST

HTTP methods (GET, POST, PUT, DELETE) plus resource identifies actions to take
The need for documentation

An API allows others to build applications on top of your services
The need for documentation

How do I access the API function?
What inputs are expected?
What kind of outputs should I expect?
What authentication is needed?
How are errors handled and reported?

...all without reading the code!
The need for documentation

How do I **access** the API function?

GET /users/23/photos/?after=101614
The need for documentation

What **inputs** are expected?

GET /users/23/photos/?after=101614

**URL parameters:**

- **after:** date, encoded MMDDYY. Results returned will all be from after this date.
The need for documentation

What inputs are expected?

PUT /photos/23/comments/2

Request parameters:
- content (string): the contents of the comment
The need for documentation

What kind of outputs should I expect?

GET /photos/23/comments/

Response:
- comments: a list of Comments belonging to the photo
  - author
  - content
  - lastModified
The need for documentation

What authentication is needed?

GET /photos/23/comments/

Client must be authenticated as a registered user of the system; may not be blocked by the uploader of the photo.

What if authentication fails?
The need for documentation

How are errors **handled** and reported?

GET /photos/23/comments/

Returns error code 304 Forbidden if authentication fails.

The error object:

- **err**: A description of the error (e.g. “Unauthorized”)
- **message**: A longer, human-readable description of the error
Examples

https://api.imgur.com/endpoints/album

https://developers.google.com/google-apps/calendar/v3/reference/events/insert

http://www.reddit.com/dev/api