REST & APIS

Code 301

AGENDA

- Standup’s
- Code Review
- Intro to REST
- Understanding RESTful APIs
- Demo

REST

REST

- **Representational State Transfer**
- The software architectural style of the web
- Leads to higher performing and more maintainable software
- Systems that conform to the constraints of REST are considered to be **RESTful**

REST

- Systems that conform to the constraints of REST are considered to be **RESTful**
- **RESTful** systems usually communicate over HTTP
- **RESTful** systems communicate with the same HTTP verbs that we are used to - with a few more to take note of:
  - GET
  - POST
  - PATCH
  - PUT
  - DELETE

REST VERBS

- REST verbs act on a noun resource (like your model)
- Let’s take a deeper look at a few commonly used REST verbs
  - **GET** - requests a representation of the specified resource
  - **POST** - requests that the server accept the entity enclosed in the request
  - **PUT** - requests that the enclosed entity be stored under the supplied URI (uniform resource identifier) - if this references a pre-existing resource, it is modified otherwise it will create the resource
  - **PATCH** - applies partial modifications to a specified resource
  - **DELETE** - deletes the specified resource

A DRAMATIC READING

Seriously, what you are about to read will change your life forever.

CONVERSATION TIME

- Alright everyone, here we go!
- Open up your browser and navigate to https://gist.github.com/brookr/5977550

RESTFUL ROUTES

XML-RPC/SOAP

- **Same method, varying endpoints:**
  - GET /getComment
  - GET /getComments
  - GET /addComment
  - GET /editComment
  - GET /deleteComment

RESTFUL ROUTES

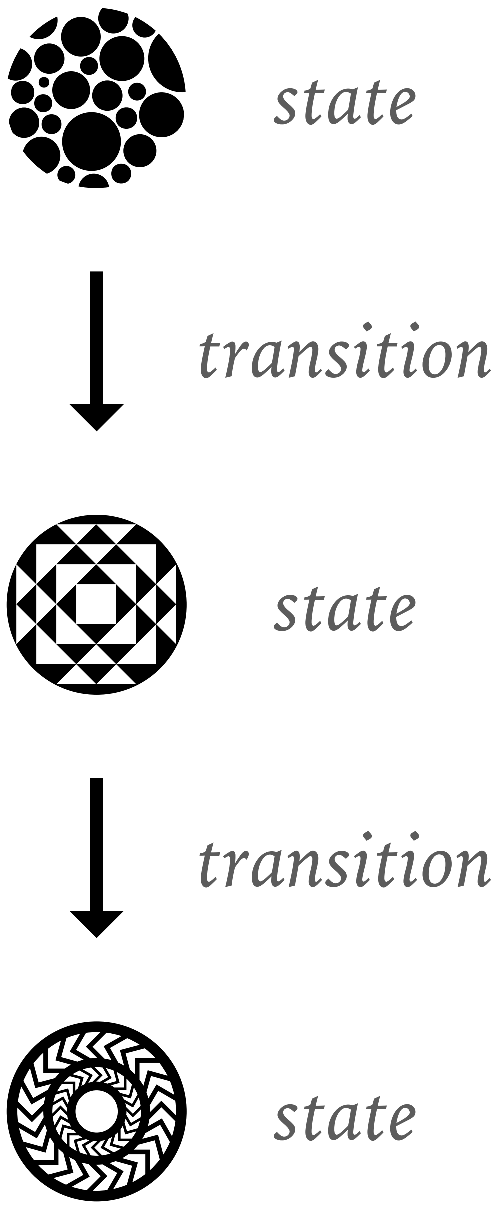
- **HTTP method + resource:**
- GET /comments/:id
- GET /comments
- POST /comments
- PUT /comments/:id
- DELETE /comments/:id
- *Resource Oriented Architecture*

RESTFUL ROUTES

- Each “endpoint” is a route, made of noun and verb.
    - the **verb** is the HTTP method (GET, POST, etc)
    - the **noun** is the resource (/comments/:id)
    - An “endpoint” is essentially just one end of a communication channel
- Always remember - HTTP is **stateless**
    - Requests do not have “memory”
    - **No individual request may be assumed to know anything about any other request.**
    - All the required information representing the possible actions to take should be present in every **response**.

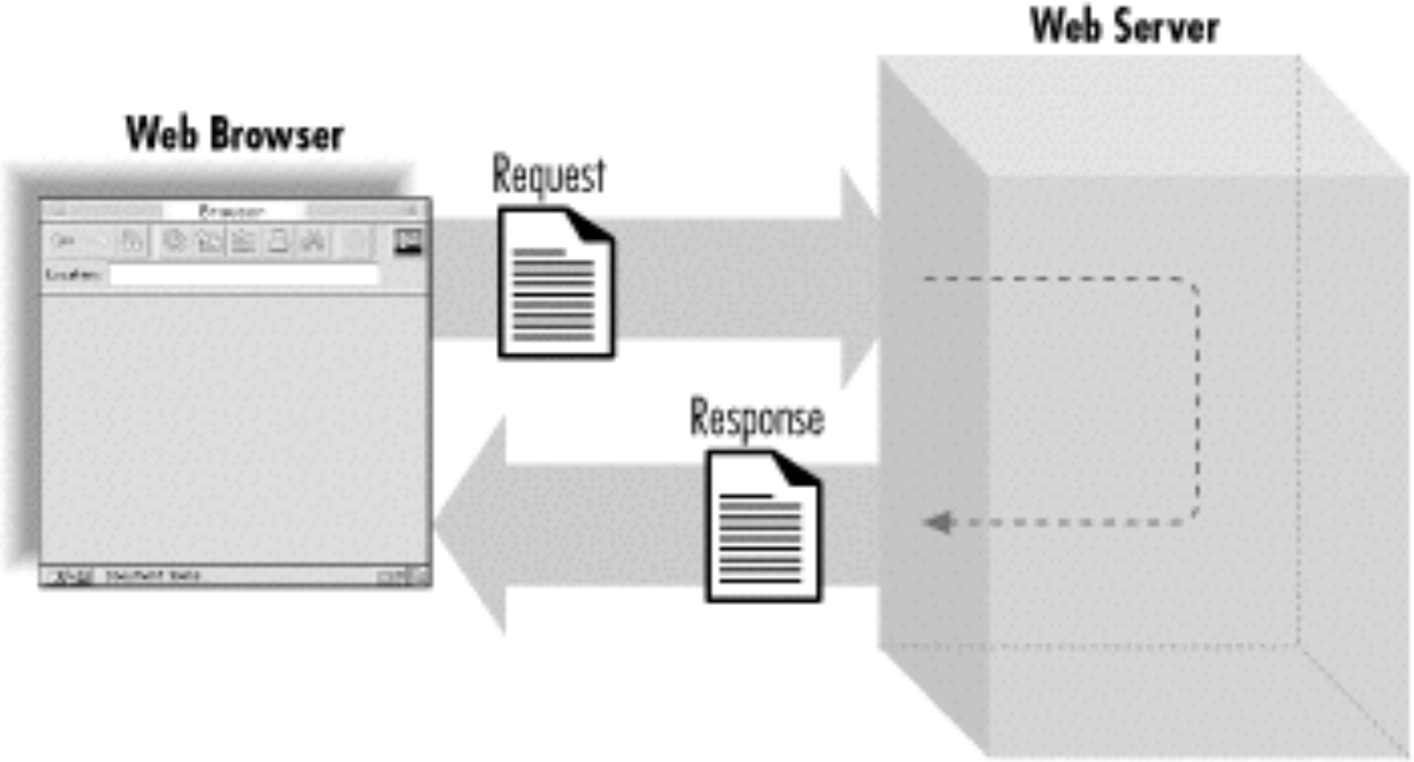
APP STATE AND TRANSITION

- GET:
  - asks for the current state
- POST/PATCH/PUT/DELETE:
  - changes the state



QUICK POP QUIZ!!!!

- What are the 3 parts of every request?
  - Which of these parts is the **noun**?
  - Which of these parts is the **verb**?
- What are the 3 parts of every response?



APIs

API - WHAT DOES THAT MEAN?

- Application Programming Interface
- Set of routines, protocols, and tools
- Can be restricted or public
- What are some commonly used (and awesome) public APIs?
  - The Google APIs (Maps, Calendar, YouTube, etc)
  - Twitter
  - Facebook
  - Soundcloud

RESTFUL APIS

- **RESTful** APIs adhere to the **REST** architectural constraints
- They are defined with a few of the following aspects:
  - a base URI (**http://mysite.com/resources/**)
  - a web media type for passing around data - such as JSON
  - the use of standard HTTP methods - **GET, POST, etc.**

REST API DEMO

Let’s start out by creating a new personal access token!

CREATE A NEW ACCESS TOKEN

- Github personal access tokens allow us to authenticate our API calls… Let’s create a new one!
- **Step 1** - Login to your Github Account
- **Step 2** - Click on your avatar/profile (top right of page)
- **Step 3** - Select “Settings”
- **Step 4** - Select “Personal Access Tokens”
- **Step 5** - Select “Generate new token”
- **Step 6** - Name your token and select any relevant “scopes”
- **Step 7** - Generate the token!

LET’S TALK TODO’S!