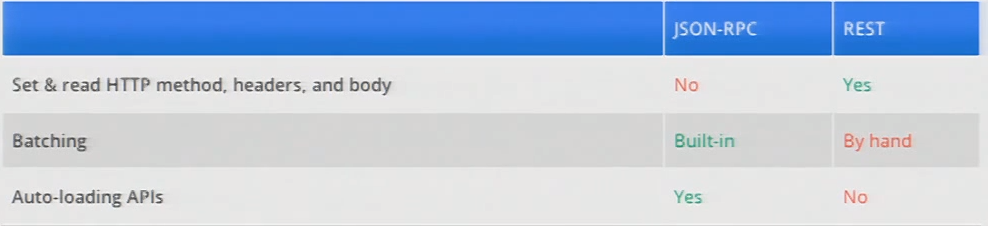
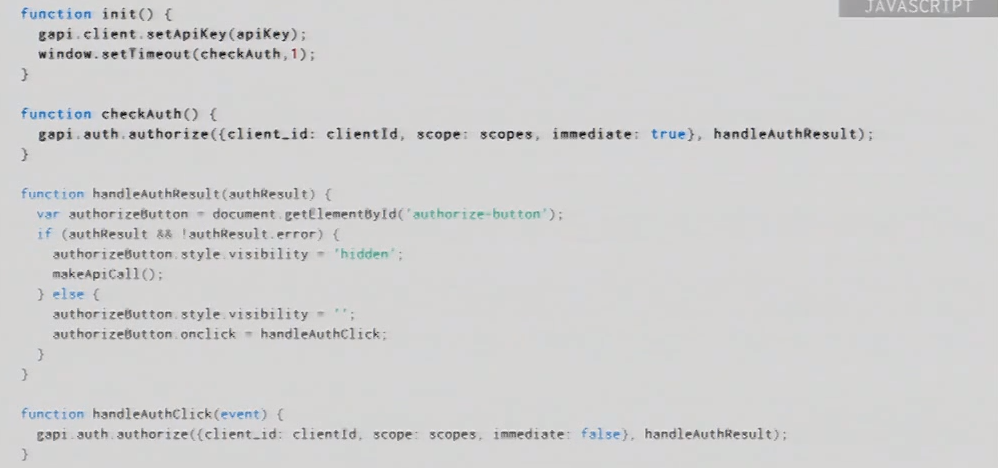
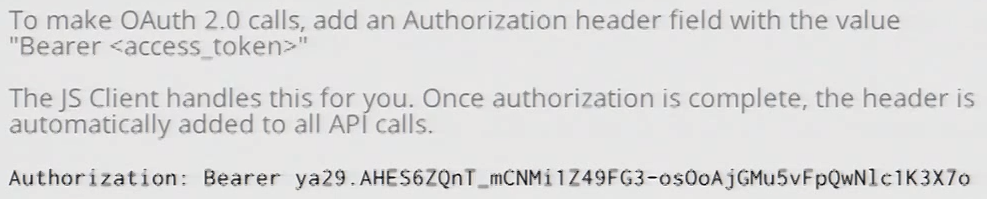
* Google APIs
  + RESTful APIs
    - Including Calendar
  + Requests to googleapis.com
  + Make JavaScript calls directly from browser
* JavaScript Client
  + Chrome/Firefox/IE/Safari
  + OAuth2.0
  + RESTful or JSON-RPC API calls
  + OAuth2.0
    - Client ID
    - Scopes for access
  + Use:
    - APIs Console
      * Create API key and client ID
      * Active APIs to access
      * Manage project – quota, team management
    - APIs Explorer
      * Documentation
      * Try API calls directly from explorer
    - URL for JS Client
      * <https://apis.google.com/js/client.js>
      * Use ?onload parameter to use the client once finished loading
      * <script src=“<https://apis.google.com/js/client.js?onload=init>”>
    - REST request
      * path
        + identifies resource to get or manipulate
      * params
        + optional
        + url parameters
        + control how method is executed or response is returned
      * method
        + optional – default to get, supports post
        + http method
      * headers
        + optional
        + headers to send with request. http headers
        + can be set to control or manipulate request
      * body
        + optional
        + serialize a json object into body to send to api
      * callback
        + optional
        + callback function to be executed when object is returned
    - 
      * me – currently logged in user
    - gapi.client.load registers all the methods to save some time and space
  + CORS – Cross Origin Resource Sharing
    - make direct xhrs to google apis
    - similar to REST requests
    - 
    - <http://caniuse.com/#search=cors>
  + Request Format Summary
    - 
  + Transport Summary
    - 
* Authorization
  + OAuth 2.0
  + <http://oauth.net/2/>
  + <https://developers.google.com/accounts/docs/OAuth2>
  + Intro:
    - <http://hueniverse.com/2010/05/introducing-oauth-2-0>
  + Authorization vs Authentication
    - Authorize: Sign in to accounts.google.com
    - User authorization for access permissions
  + Access unprotected resources
    - Only need API key
  + Protected resources
    - Client ID exchanged for access token, access to a user’s protected resources
    - How To:
      * APIs console
      * Services page
      * Turn on calendar
      * Agree to TOS
      * API Access
        + API key – identify application for unprotected resources
        + Create an OAuth2.0 ID
      * OAuth2.0 ID Setup
        + Product Name
        + Google Account
        + Product Logo (optional)
        + App type
        + Site or hostname
        + Authorized Redirect URLs

Unknown if needed

* + - * + Authorized JavaScript Origins

Definitely need these!

* + - * + Origins need to match EXACTLY!
        + If using a port, include it there
      * Then Client ID is generated
      * Passed to exchange for access token
      * Recommended to always send API key
    - 
      * immediate: true
        + Check if user is already logged in
    - Access tokens expire!
      * Can be changed!
      * Probably better to make it refresh 5 minutes sooner
    - 
* 