URLS & ROUTING

Code 301

ROUTING & CONTROLLERS

> The controller in a web app is a bit more complicated, **because it has two parts**. The first part is the web server (such as a servlet container) that **maps incoming HTTP URL requests to a particular handler** for that request. The second part is those handlers themselves, which are in fact often called “controllers" [or actions].
>
> So the C in a web app MVC includes both the web server "overlord" that routes requests to handlers and the logic of those handlers themselves, which **pull the data from the database and push it into the template**.
>
> *-Terence Parr*

ROUTES: YOUR PUBLIC API

- What resources does your app offer?
- Abstract away the details of html files
  - https://www.codefellows.org/blog.html
  - https://www.codefellows.org/blog/
  - Same page!

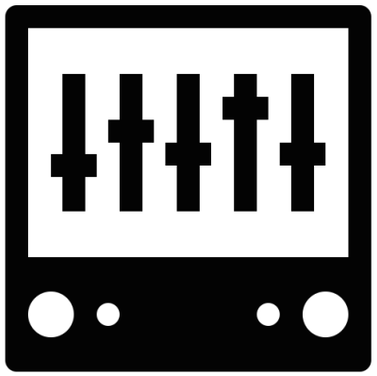
ROUTES: YOUR PUBLIC API

- Making files is slow
- We want programmatic control of what our app can do
- GET: /
- GET: /about
- GET: /articles/42
- GET: /articles/42/edit
- PUT: /articles/42

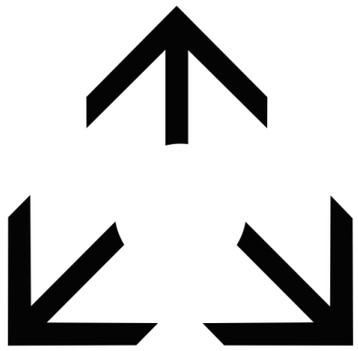
CLIENT-SIDE ROUTING

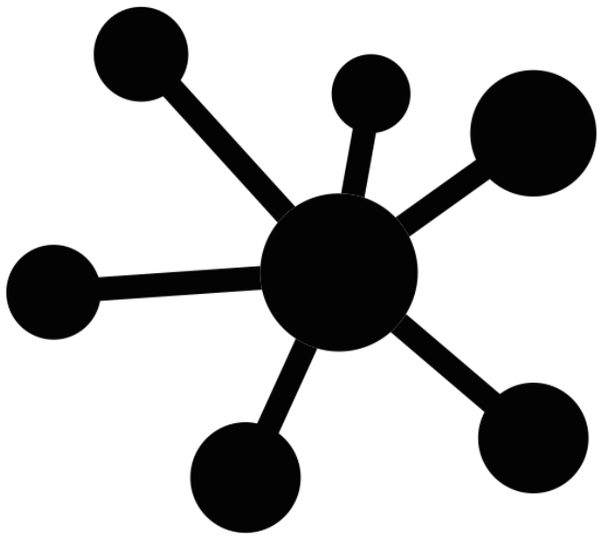
- Client-side, the route can tell us a few things:
  - What resource the user wants:
    - hipmunk.com/flights
  - Additional info:
    - hipmunk.com/flights#f=SEA;t=HNL;d=2015-12-22
- JavaScript can interpret this route, and…
  - take apart the pieces
  - call the proper function to handle it all
  - …all from a single index file: NO RELOAD!

MVC FLOW

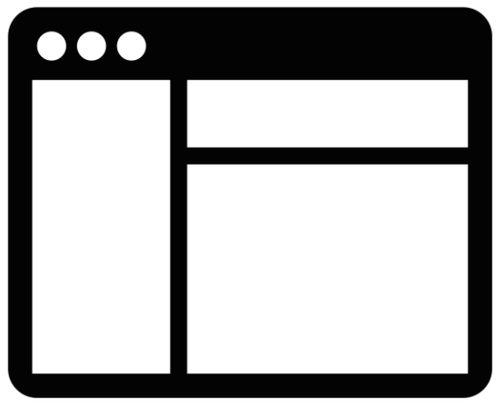


Controller





Model



View

CLIENT-SIDE CONTROLLER

- Our controller will handle the user request
- The controller converts a route into displayed content…
- with the proper data load.
- Simply a list of functions (aka: “actions”), waiting to be called
- One controller per resource:
  - ArticlesController
  - FlightsController
  - UsersController

CLIENT-SIDE ROUTING: HELPFUL LIBRARIES

- page.js
  - Connect routes with handling function:
    - `page('/', user.list)`
    - `page('/', index);`
    - `page('/about', about);`
    - `page('/contact', contact);`
- Many many more examples:
  - https://github.com/visionmedia/page.js
  - Install: copy/paste `page.js` file into your project

CLIENT-SIDE ROUTING: HELPFUL LIBRARIES

- `pushstate-server`
  - Sends all requests to: `index.html`
  - Passes through static files:
    - `/scripts/blogArticles.json`
    - `/templates/article.html`
- Requires full-path URLS, with starting slash
- More details: https://github.com/scottcorgan/pushstate-server
- Install: `npm install -g pushstate-server`

ROUTING DEMO

RECAP

RECAP

- URLs power the browser
- JavaScript can leverage the URL to control the app, without going back and forth with a server
- Assignment:
  - Work from assignment directory code
  - Add a router
  - Add a controller for articles