Computer Science E-75: Building Dynamic Websites
Harvard Extension School
Fall 2010

Project 3: The BART

due Monday, 29 November 2010, noon ET

Goals.

- Introduce you to Ajax.
- Introduce you to third-party APIs.

Recommended Reading.

- http://www.w3schools.com/dom/
- http://www.w3schools.com/js/
- http://www.w3schools.com/ajax/

- http://www.bart.gov/schedules/developers/etas.aspx
- http://api.bart.gov/docs/overview/
- http://code.google.com/apis/maps/documentation/javascript/

Academic Honesty

All work that you do toward fulfillment of this course's expectations must be your own unless collaboration is explicitly allowed by some project. Viewing or copying another individual's work (even if left by a printer, stored in an executable directory, or accidentally shared in the course's virtual terminal room) or lifting material from a book, magazine, website, or other source—even in part—and presenting it as your own constitutes academic dishonesty, as does showing or giving your work, even in part, to another student.

Similarly is dual submission academic dishonesty: you may not submit the same or similar work to this course that you have submitted or will submit to another. Nor may you provide or make available your or other students' solutions to Project 0, Project 1, Project 2, or Project 3 to individuals who take or may take this course (or CSCI S-75) in the future.

You are welcome to discuss the course's material with others in order to better understand it. You may even discuss problem sets with classmates, but you may not share code. You may also turn to the Web for instruction beyond the course's lectures and sections, for references, and for solutions to technical difficulties, but not for outright solutions to problems on projects. However, failure to cite (as with comments) the origin of any code or technique that you do discover outside of the course's lectures and sections (even while respecting these constraints) and then integrate into your own work may be considered academic dishonesty.

If in doubt as to the appropriateness of some discussion or action, contact the staff.

All forms of academic dishonesty are dealt with harshly.

Grades.

Your code (CSS, JavaScript, PHP, SQL, XHTML, *etc*.) will be evaluated along the following axes.

Correctness. To what extent is your code consistent with our specifications and free of bugs?
Design. To what extent is your code written well (*i.e.*, clearly, efficiently, elegantly, and/or logically)?
Style. To what extent is your code readable (*i.e.*, commented and indented with variables aptly named)?

Implementation.

☐ Your mission for this project is to implement The BART, a mashup that allows users to visualize BART routes on a Google Map and also click stations to see when the next trains depart (or arrive). The overall design and aesthetics of this site are ultimately up to you, but we require that your site meet some requirements. All other details are left to your own creativity and interpretation.

Feature Requirements.

☐ Your site's homepage must display an embedded Google Map, centered and zoomed in on San Francisco.

☐ Your site's homepage must provide the user with a way of selecting one BART route at a time. Once selected, a route should be drawn as polylines on the map in the route's official color, with markers representing each of that route's stations. Each station, when clicked, should trigger an info window that summarizes the next trains departing from (or arriving at) that station.

Technical Requirements.

☐ Your site must live at `http://projects.domain.tld/3/`, where `projects.domain.tld` is your own domain.

☐ All PHP files must be chmod'd 600.

☐ Your site must use version 3 of the Google Maps JavaScript API.

☐ It suffices to use only the Real BART API (`http://api.bart.gov/docs/overview/`), but you are welcome to use the Simple ETA Feed (`http://www.bart.gov/schedules/developers/etas.aspx`) and/or the GTFS feed (`http://www.bart.gov/dev/schedules/google_transit.zip`).

☐ You should cache locally (on disk or in a MySQL database) data that does not change every minute (*e.g.*, routes and their stations). Your mashup should only query the BART API or (Simple ETA Feed) for real-time departure (or arrival) times.

☐ You are welcome, but not required, to use any of the JavaScript libraries recommended in Lecture 6's slides.

☐ Your XHTML must be valid (or "tentatively" valid), unless some feature of your site requires otherwise (for the sake of some browser); explain in XHTML comments any intentional invalidities. Your XHTML should also be as pretty-printed as possible. Your CSS need not be valid.

☐ Your PHP must be extensively commented and be as pretty-printed as possible.

☐ You may use a WYSIWYG editor to generate XHTML and/or CSS that you would like to use in your site.

☐ If you incorporate or adapt snippets of code from the Web into your project (*e.g.*, examples from `php.net`), cite the code's origins with PHP comments.

- □ If you incorporate images from the Web into your project, cite the images' with XHTML comments.
- □ Your website must appear and behave the same on at least two major browsers, namely:
  - □ Chrome 5.x
  - □ Firefox 3.x
  - □ Internet Explorer 8.x
  - □ Opera 10.x
  - □ Safari 5.x

Exit Interview.

□ Once done with your site, put together a readme at:

```
http://projects.domain.tld/3/readme/index.php
```

Treat this readme as your opportunity not only to explain but to justify your design decisions. Tell us why you modeled your database tables as you did. Tell us why you chose, say, select menus over radio buttons for some feature. Tell us with which two (or more) browsers we should evaluate your site. And give us an overall sense of how your site works (*e.g.*, tell us which files do what). But still be succinct; keep this readme to just a few paragraphs in length.

How to Submit.

□ A few days prior to this project's deadline, instructions for submitting your work will be posted to the course's Google Group. Be sure to look for those directions and then submit your work prior to this project's deadline.