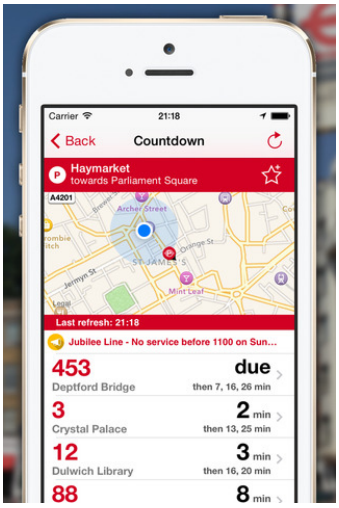
Bus Stop Test – Notes by Arturo Batanero (31/03/2015)

Requirements

- Based on the suggested requirements, I've tried to develop the most realistic app possible in a time limit of 1,5 working days (this past weekend)
- The use case scenario is a London bus stop without a departures panel, so the app has been designed to run on your smartphone's web browser.
- Looking for references in UI design (in a real project I would expect that UI to be delivered to me) I ended up choosing this mobile app: "London Bus Live Countdown".
- The app tries to follow this UI and most of its suggested functionality:



Development notes

- The app has been built using the Backbone.Marionette framework.
- It is modularized using Require.js. The only variable in the global context is "app", the Marionette application object.
- **Important**: I didn't entirely build the HTML/CSS layout. I can modify a layout and even build one from scratch, but not in this time frame. My transition from ActionScript development to Web standards development is still a work in progress. I expect to be able to create this kind of layouts soon, but that bus hasn't arrived yet…

Almost done, but…

- My first intention was to let the app to be in control of the passing of time, instead of relaying entirely on API calls. It would have updated the departures data on every API call, just like now, but if no further calls were made (internet failure, for instance) it would have kept updating departure times as time passed, until no more buses were left. In fact, it already parses arrival time strings into integer timestamps, but there was no time left.
- It doesn't follow the MVC pattern as much as I would have liked. Time constraints again. Models should broadcast state changes using events and be totally unaware of views existence. It's not always the case, I'm afraid.
- On app start my intention was to automatically select the closest bus stop to the user's location. No time left, again.
- A specific marker for the user's simulated location. A button to pan the map to that location, another one to refresh manually (now only automatically every 30sg).... the list goes on