* Template
  + The carousel needs to treat odd-sized photos better, the left/right arrows do not nicely align with some viewports
  + Need to fix selection box with fields
  + Need to better model selection choices  
    Need to improve visual display on non-mobile viewports before finishing
  + Add view for “Loading/Error” when waiting on API call to return
* Controller
  + Need to Angular-ify google charts. This includes moving it to it’s own directive and getting the functions out of a controller as it currently is; it goes highly against Angular practice
  + Need to better model the selection of which test the user is viewing
    - Need to remove the function of getTestById by accomplishing this, again, Angular practices
    - Code throughout the controller and template will be extremely cleaned up by this act.
  + Need to produce a call to the service and a listen $on for when a user changes a graph axis. And a ng-change in the template for changing
  + Move the call to load the axis from on success of test list, to inside the test list and shove it into the JSON array.
  + Add readability for the “code” portion of the selTest data. (Change it from an integer of state to a string saying good/error/load/start)
* Service
  + More function calls for above.
* APIs
  + getTestFields, make it grab a dynamic directory and make it so the user cannot select a bad directory (or shove this file completely into getEventList)
  + getEventList, grab all field names from the csv files here instead of later so we limit ourselves to one call for loading and lazy-load the remaining graphed data.
  + getDataFromField, a new PHP file that will take a field and a test id, verify that they are both valid, and return data points for the requested field. This should be paired with the service, controller, and view.
* Charts
  + Need to Angular-ify chart so that it has its own directive, controller, and potentially service.
  + Needs to display better (perhaps no bootstrap styling or get it centered somehow), and needs to be able to be handled more MVC-ly
* Other
  + Consider implementing native AngularJS routing
  + Consider applying a styling guide for consistency across all files (some “ & ‘, etc)
  + Link photo with a selected dot in the plot and vice versa.
  + Consider an admin/webmaster page with a simple login, so that a professor can potentially add a future test.
  + Ensure that all SQL & file-grabbing has sanitized user inputs