Committers Call 2011-04-04 - Progressive Enhancement

Participants

Julie Meloni
Joe Gilbert
John Scofield
Jessie Keck
Chris Fitzpatrick
Rick Johnson
Dan Brubaker-Horst
Bess Sadler

Agenda: Progressive Enhancement & Unobtrusive Javascript

Graceful Degradation vs. Progressive Enhancement

Progressive Enhancement:
Fully functional & fully tested without any javascript, then add JS behaviors

Graceful Degradation:
Starting from full JS functionality and making it degrade gracefully

We want to use the Progressive Enhancement approach.

Current State of the Hydra Javascript

There is a lot of javascript. Too much javascript

- takes too long to load into the page
- difficult to debug
- hard to maintain/extend/refine

Many javascript features do not degrade gracefully. Even less (none?) are progressively enhanced.

Basic observation: There are 4 different approaches at play in the code (confusing, hard to debug)

- Some inline javascript
- Event-driven javascript – ok in spots, but not ok as overall arch
- Rails-ish, page-specific javascript controller (ie. edit.js, show.js)
- Plugins – ie hydraMetadata

Other observations:

- Submission form currently requires ajax. Would require multiple submit buttons if ajax disabled.
- Fluid Framework is problematic

New Approach in Blacklight: (Jessie will follow up with more info)

What should be done?

- All features progressively enhanced
- Less (lines of) Javascript
- No inline javascript
- Put code in consistent places
- No javascript includes in view templates. They should go in Rails controllers
- Javascript should not change css

Accessibility

- Ensure Valid Markup!
- Section 508
- ... Julie will provide more reference info

Q: Where should you put the code that specifies which js to load for a particular page?
A: Hold off on that question. Wait and see what conventions come out of Libra.

Should we have a cucumber step definition for "And the html should be valid"?
This would rely on a network connection. Should keep that in a separate cucumber namespace. Possibly define a single features/valid_html.feature that runs through all of the pages and runs the validity check...

**Getting rid of Fluid Framework**

This is revisiting an old topic. Yes we should get rid of all Fluid dependencies. They have already been removed from Libra. They should be removed from Hydrangea (and any other Hydra heads) in anticipation of the next Libra merge.

**How will it be done?**

**Using Libra as a straw man for identifying conventions**

Libra goes into production this week. UVa has cleaned up many parts of the javascript

- reduced the overall amount of javascript loaded
- removed dependency on Fluid framework
- squashed bugs
- cleaned up html
- addressed some accessibility issues

They will be documenting what they changed, how and why. We can use this code as a straw man for identifying conventions around javascript behavior.

**In the meantime, Hydrus / Hypatia development proceed with NO javascript**

Hydrus & Hypatia developers will

- remove the javascript *includes* from Hydrus & Hypatia
- leave the javascript *files* unouched -- allows us to retain any potentially important javascript customizations
- implement & test all features with javascript disabled

This provides the foundation for using progressive enhancement approach in all features of all hydra heads. Once we decide on javascript conventions, we will enhance these features according to those conventions.

**Meanwhile, work on a solution for Automated Testing of JS**