Simplicity, Abstraction, and the Modern Website

Kevin Pittman

Welcome!

Begin Presentation

Introduction

- My Background
  - Senior Web Developer: Georgia Tech Ivan Allen College of Liberal Arts
  - Systems and Web App Administator for Sixteen Years Prior
  - No prior dedicated developer for the college or its six schools
- Inherited thirty-something existing websites by many different developers
  - Faculty / Staff / Students (Graduate & Undergraduate)
  - Small outside development companies

Recurring Problems in Existing IAC Sites

- In reviewing these thirty-something sites, I found:
  - No standardization between sites, or even within sites
  - Too much complexity / functionality / data
  - Kludged, uneditable landing pages
  - No documentation
  - Cluttered, unorganized backend filesystems and databases
- We needed to rethink the way we were approaching website design!

Where Visual and Functional Design Meet and Clash

- What is at the root of the problem?
  - Too commonly, visual design drives the project, resulting in a functionally poor site
- Causes:
  - Wanting to please the client too much
  - Not recognizing the limitations of the chosen CMS

Reality of Visual and Functional Design

- Visual and functional design must work in **harmony** - one cannot forcibly dictate the other
  - Visual design needs to leverage capabilities of back end without overtaxing it
  - Back end needs to properly support visual design without hacks
- Websites need well-designed structures to support their visible skins

Rethinking Website Design

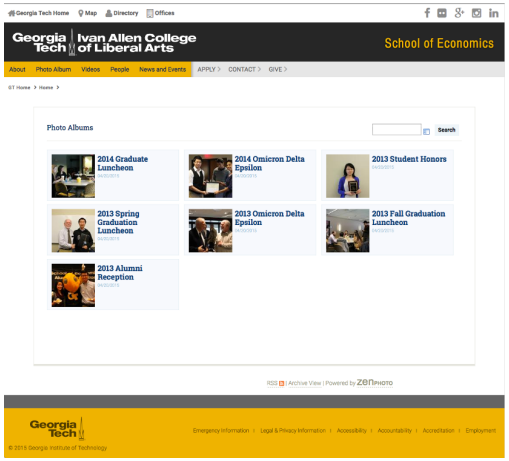
- Rebuilding half-a-dozen of our websites led to a new design philosophy
  - The Strategy of Abstraction-First Design
  - The Art of Simplistic and Intuitive Landing Pages
  - The Virtue of Documenting Your Site
  - The Joy of Streamlining Back End Systems

The Strategy of Abstraction-First Design

- What can be better handled by an external web application?
- CMS's may be the swiss-army knives of the web, but ...
- Better to do a few things well than a lot of things poorly.

Abstraction-First Examples: Photo Albums

- Extracted several hundred photos from custom CMS site to ZenPhoto
- Setup ZenPhoto in directory under Drupal installation - /albums
- Created ZenPhoto GT theme and CAS integration - looks like part of Drupal 7 site
- Could also have used hosted app (Flicker, Photobucket) or even social media (Instagram, Facebook)



Abstraction-First Examples: Faculty Profiles

- Faculty Profiles: Very complex implementation of a curricula vitae
  - Not your typical everyday user profile
- Every school website and many project center sites had local profiles
  - Redundant data, and redundant code implementations
- Abstracted this functionality out to a central faculty profile server
- Central server integrates with Drupal websites via one custom module

Abstraction-First Examples: Resource Savings

- Single Drupal 6 Site with local faculty profiles, news, and current events:
  - 1000 nodes, 93 non-core modules
- After extracting profiles, news, and events to central servers:
  - 85 nodes, 26 non-core modules
- New site has all needed functionality, runs much faster, and is much easier to maintain

Arguments for the Abstraction First Concept

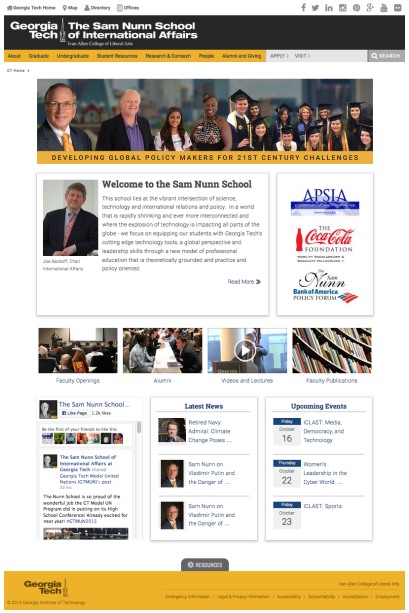
- Better security for main and satellite sites
  - Finer grained control over what users can access
  - Fewer internal dependencies make patching quicker and easier
- More uptime for main and satellite sites
  - Individual sub-sites can be patched without bringing down any others
- Multiple simplified sites save time when major upgrades are needed
  - Frankensites never get upgraded!

The Art of Simplistic and Intuitive Landing Pages

- Follow the flow of your CMS and selected theme
  - Don't force the CMS or theme to do something it's not designed to do
  - Today's hack will always end up being tomorrow's massive headache
  - Be gently honest with the client about requests that aren't feasible
- Avoid unnecessary eye candy, bells and whistles
- Never put content directly into template files

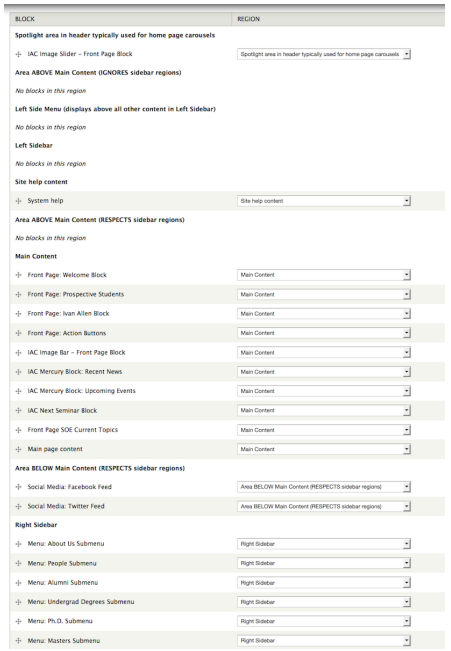
Drupal Specific Landing Page Tips

- Always assume visual block orderings will change
  - Use independent blocks for each landing page visual block
  - Start with standard styles on all landing page blocks: margins, borders, padding, etc.
- Keep images / content files with landing page blocks by using the *nodeblock* module
- Provide contextual links to a block's data source
  - Visit http://webdev.iac.gatech.edu/samw/ for in-depth details



The Virtue of Documenting Your Site

- Why document? You **will** forget!
- Create self-documenting designs
  - Use clearly descriptive names and identifying prefixes
    - Sidebar Menu: Academic Programs
    - Front Page Block: Upcoming Events
  - Avoid uninformative titles like 'About'
- Put additional documentation in unpublished pages



BLOCK	REGION
Spotlight area in header typically used for home page carousels	
IAC Image Slider – Front Page Block	Spotlight area in header typically used for home page carousels
Area ABOVE Main Content (IGNORES sidebar regions)	
No blocks in this region	
Left Side Menu (displays above all other content in Left Sidebar)	
No blocks in this region	
Left Sidebar	
No blocks in this region	
Site help content	
System help	Site help content
Area ABOVE Main Content (RESPECTS sidebar regions)	
No blocks in this region	
Main Content	
Front Page: Welcome Block	Main Content
Front Page: Prospective Students	Main Content
Front Page: Ivan Allen Block	Main Content
Front Page: Action Buttons	Main Content
IAC Image Bar – Front Page Block	Main Content
IAC Mercury Block: Recent News	Main Content
IAC Mercury Block: Upcoming Events	Main Content
IAC Next Seminar Block	Main Content
Front Page SOE Current Topics	Main Content
Main page content	Main Content
Area BELOW Main Content (RESPECTS sidebar regions)	
Social Media: Facebook Feed	Area BELOW Main Content (RESPECTS sidebar regions)
Social Media: Twitter Feed	Area BELOW Main Content (RESPECTS sidebar regions)
Right Sidebar	
Menu: About Us Submenu	Right Sidebar
Menu: People Submenu	Right Sidebar
Menu: Alumni Submenu	Right Sidebar
Menu: Undergrad Degrees Submenu	Right Sidebar
Menu: Ph.D. Submenu	Right Sidebar
Menu: Masters Submenu	Right Sidebar

The Joy of Streamlining Back End Systems

- Keep workspace organized from the start
  - Avoid temptation to clean it all up at the end - you never have enough time
- When migrating/upgrading a site, clone old site and clean it out first
  - At start of projet, your mind is clearer and you're not under deadline pressure
  - You have entire development period for client to notice missing content
  - Not just pages, but taxonomies, user accounts, modules, themes, etc.

Streamlining Tips: Modules & Plugins

- Don't install anything until you know you need it
  - Not only cleaner, but more secure, and speeds up site
  - Drupal: drupal_get_path() scans every module in every directory!
- Develop a standard organization for add-on components at start and stay with it
  - Keep all 3rd party add-ons together and use scripted upgrade tools to manage them
  - In Drupal, use *libraries* module to group libraries separate from modules

Streamlining Tips: Utilities and Content

- Setup private folder outside webroot for archive files, external tools, scripts, etc.
  - Nothing should be shared via the webroot that doesn't need to be shared
- Never load content files directly into the filesystem
  - Attach files to pages within the site
  - Use redirects to preserve old URL paths

Conclusion

- My Philosophy in Summary:
    - The Strategy of Abstraction-First Design
    - The Art of Simplistic and Intuitive Landing Pages
    - The Virtue of Documenting Your Site
    - The Joy of Streamlining Back End Systems

Presentation Resources

http://webdev.iac.gatech.edu/samw/

Questions?