



Building the Next Generation Public Library Website with Drupal

John Blyberg, Darien Library, CT
Eli Neiburger, Ann Arbor District Library, MI

Drupal is a CMS

- CMS: Content Management System
 - Import or create documents
 - Fluid management of blog entries
 - Effective user management
 - Delegation of roles
 - Version control
 - Separation of form from content
 - User participation
 - Searchable

Drupal is a content management system

Most of you are familiar with CMSs. They really represent a natural progression of online publishing and, well, content management.

They allow users to create and import content with a very low technical barrier.

They generally allow you to arrange all your content in a logical and organized way.

They accommodate multiple authors with varying degrees of editorial control

Many allow you to do version tracking, giving you the ability to roll back to previous version

They separate form from content, so your designers can design and your authors can write

They invite and facilitate user participation such as comments and forums

And they're searchable.

As opposed to...

- Screwing around with Dreamweaver
- Fussing with FTP
- “WTF do I do with all these files?”
- Circ vs. Reference vs. Youth vs...
- (Battling for the unsustainable)

If you're not using a CMS, chances are, you're using something like Dreamweaver which, in itself, is a very good piece of software.

If you're a professional web designer. In too many cases, however, an institution's strategy for distributing content development has been to distribute copies of dreamweaver around to whoever was involved. For those of you who are familiar with using dreamweaver in this kind of environment, you know that there are several problems that inevitable emerge.

FTP is actually a very robust protocol, it's been around far longer than http. The problem is that it's now so poorly understood by most systems administrators. As a result, accessing files via FTP can be very problematic, especially when one or more firewalls, or even simple routers stand between Dreamweaver and the server.

Then once you do get in, you're presented with an ungodly manifest of files that, over time, has grown so out of control, nobody knows what to do with them all.

And to make matters worse, turf wars erupt between departments over content, design, format. These veiled resentments manifest themselves over time in the form of animated gifs, and grainy low-res pictures of generic brown books that look like they were lifted from an old version of print shop.

In the end, it all amounts to a political entanglement over an unsustainable model. The irony here being that this outdated model is what often makes the library website such a magnet for strife.

It's 2007, does your website still suck?

Until you're 100% sure that it doesn't, it probably does.

You wouldn't be attending this session if you weren't sure.

That's ok, if you can admit to the problem, and admit that you're powerless over it, then there may be some hope for your web site.

The problem I see with 90% of all library web sites is that they range from banal to mediocre. I know that is a harsh assesment, but unfortunately it's true.

There is no reason why we cannot have world-class web sites.

Our websites need to be the public representation of a cohesive and comprehensive technology program. [say again]

If your website is awful, it's probably because your library's technology program is neither cohesive nor comprehensive.

So I'm telling you two things right now:

- 1) Don't bother with a new website unless you're completely and absolutely committed to making it excellent.
- 2) Don't bother trying to make an excellent web site, until your technology is excellent.

And when I say "Excellent technology" I'm not talking about top-shelf equipment. I'm talking about implementation. Even a small, lightly-funded library can have a robust, high-performance network build on old PCs. Open source allows us to do this, but it takes expertise.

What makes a good PL Website?

- A firm commitment to not settle
- Single sign-on
- Integrated OPAC
- Significant quantities of content generated every day
- Usefulness
- Understand your community
- Youth
- Staff buy-in
- The website is an extension of the library experience, not a resource

But what are some of the key features or components of a good public library website?

Well, first you need to ditch the expectation that you may have to compromise, or settle. You wouldn't just settle for a mediocre building if you were building a new library, why would you settle on something that gets even more visitors?

Single sign-on is a critical, critical component of customer service, and we're all still grappling with a jarring separation between our OPACs and our web sites. Our users don't care that it's difficult, all they know (either consciously or subconsciously), is that it's awkward and inconvenient.

Also, a good website generates content on a daily basis. This can (and should) be blog posts done by staff and patrons. Your front page should not be a mishmash of information, it should be content. Information should be easily available through elegant user interface design.

Of course, you need to be providing content and services online that your community wants and can use.

I'm a firm believer that anytime we can get our youth involved in the library in some productive way, no matter how loud or raucous, we're infusing our institutions with priceless energy and verve while instilling a sense of stewardship in our future patrons.

And it's important to realize that your website will fall flat if you don't get buy-in from your entire staff. If you're a techie, you know that you can provide the best tool possible for a job, but if people aren't interested in it, it will stagnate. We see that with 99% of all mediawiki installations in the world.. not enough interest, not enough participation.

Finally, let's stop treating our websites as though they are resources. They do, indeed,

Drupal is a means to an end, not the end itself

Before I get into some of the specifics on why I love Drupal, let me be clear that Drupal is not going to solve your problems. If you come away from this session thinking that installing Drupal will result in an award-winning website, we will have done you a big disservice.

Drupal will give back to you what you put in to it. That doesn't just mean you, the IT guy, or you the highly-motivated-but-solitary blogger, but you--your entire organization.

A little about Drupal

- Open Source
- Written in PHP
- Relatively low hardware requirements
- Can be run on Open Source platforms
- The software is free (the time is not)
- Pronounced “Droo-pull”

Taxonomies

- Classify content
- Site organization
- Cross-post stories, blog entries, etc
- Can be extended to custom “nodes”

I want to talk for a moment about Drupal’s taxonomy system which is really rather unique. Drupal uses a classification system for all content in its system that can allow you to place content virtually anywhere on the site at any given time.

When you create what Drupal calls a “node” which is basically a unit of content, such as a blog post or a static page, you can put that content in as many taxonomy categories as you like. These taxonomies are predetermined by the site administrators, and really, the site designers.

A good way to think about Drupal Taxonomies is that they are like tags, but with authority control.

Web site content and behavior can then be custom tailored around whatever taxonomy is active on the page.

For instance, if a user went to a blog post about a community fund drive, that post may be a member of the “community” taxonomy. You could write a very simple, custom module that would aggregate local news, events, and announcements from virtually any source, and present it on-the-fly in a sidebar or even embedded in the node itself.

And because the taxonomy system is built over the node system, you can define custom nodes and have them naturally fall into the same classification system. A custom node is basically a custom content-type that maintains its own unique data fields.

The theme engine

- Separates form from content
- Closely integrated with Drupal's API
- Supports multi-site or civic spaces
- Completely customizable
- Several templating engines available
 - PHPTemplate, Smarty...

For those of you who are more aesthetically inclined, Drupal's theme engine is very comprehensive.

I mentioned before that the CMS allows us to separate form from content. Well, it's the theme engine that is responsible for that. It means that a single change to your CSS or HTML template affects the entire site.

Drupal's theme engine is very closely integrated with its API (which I'll talk about in a minute). This means that you can do some very neat custom things.

If you're like Darien, where we host a number of community web pages, Drupal is very good at hosting what are called "civic spaces". Civic spaces are essentially independent organizations that have agreed to allow their web content to fall under the auspices of a single CMS. As libraries, we're perfectly positioned to provide that service, and from an archival standpoint, we benefit greatly. From a purely technical standpoint, the participating organizations benefit.

Drupal gives you option to use several different templating engines such as PHPTemplate and Smarty. These both have grown to be de-facto standards in the PHP web development arena, so the templates you create just happen to be very portable--you could move them out of Drupal and employ the same template engine.

The API

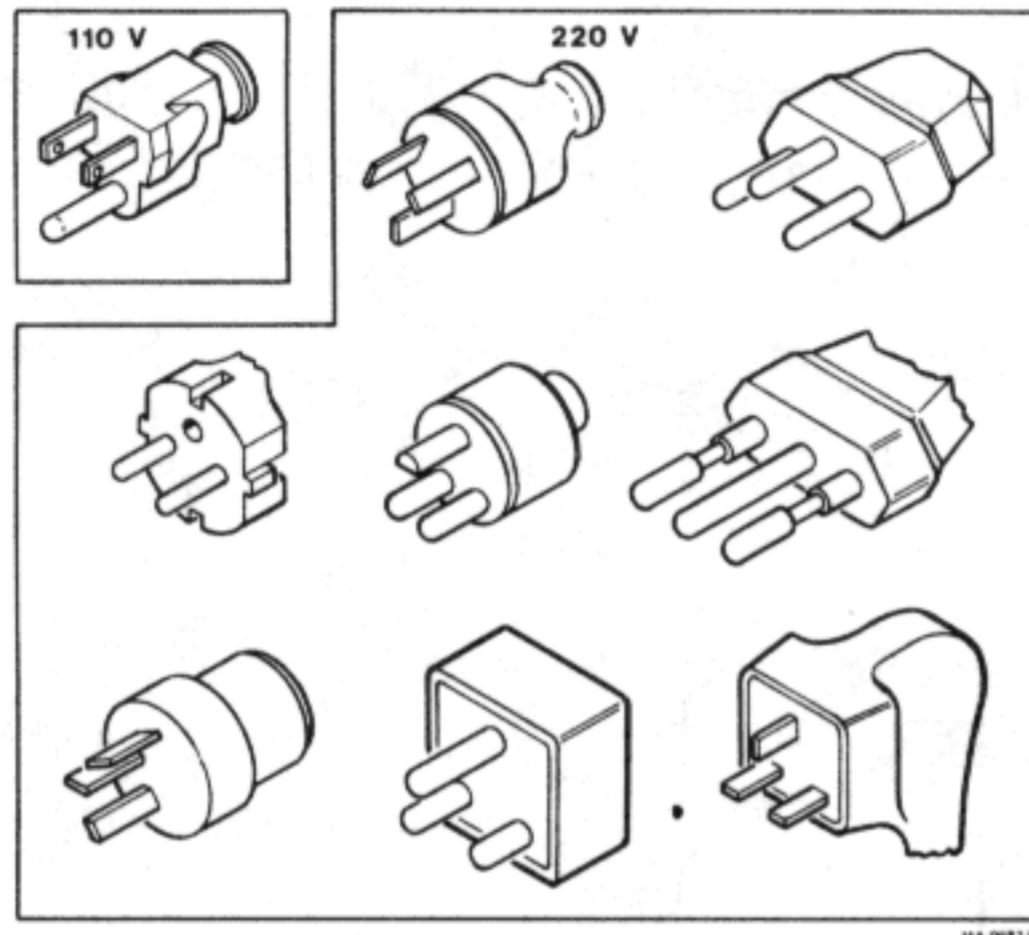


Now, I want to spend some time talking about Drupal's API.

An API, or Application Programming Interface, can be likened to a wall outlet. On one side of the outlet (the outside) you presumably have an appliance you'd like to plug in. That appliance could be very complex, like a computer, or very simple, like an iron.

But you've got to somehow get the electricity on one side of that outlet into the appliance before it'll work. That's where the outlet comes in.

The API



But your outlet cannot just be any old outlet, the shapes of the holes cannot be arbitrary. There has to be some kind of standard--and there is.

Here we see a plug that fits a 110v outlet. And we know that when we plug it in, that 110 current will have been conditioned to 60 Hz on the pole outside our home, and that the plastic used in the construction of the outlet meets the specifications set out by the Underwriters Laboratory, and that if the outlet isn't grounded, it has a built in ground-fault interruptor.

Well, maybe you're not thinking about all that when you plug something in, but that's the point. The outlet is a killer app, using it is such a seamless, transparent experience because of these standards.

And we see here, right away, the benefits that the 110 standard has over the multiple 220 standards. For those of you who have travelled overseas, you know the problems you run into sometimes.

In terms of software, the API allows you get move data, seamlessly, back and forth between two separate pieces of software, that's it. Drupal allows you to do that in several different ways.

Hooks

(Do stuff when something happens)

Drupal's hooks allow you to programmatically define a response to a particular event. In other words, do stuff when something happens.

There are two components to the hook.

First, the event and the event handler

and second, all the data and metadata associated with that events

This means, you can program Drupal to sync up a library card number with your ILS on a new-user registration event. Or you can sync their password on a change password event, or update their personal information on that event. You can do stuff when someone searches the site, or when they post a story, or when the nightly scheduled maintenance is done.

Hooks are a very powerful tool in terms of housekeeping for Drupal, but they significantly extend it's scope beyond that of a standard CMS.

Search Functions

Create custom content types (node types) that are instantly searchable!

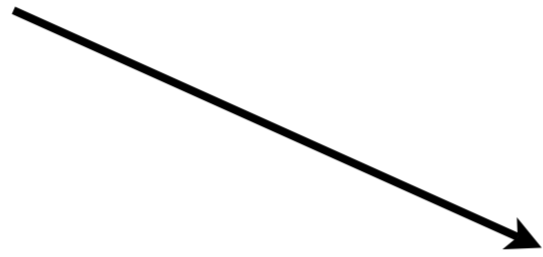
Use the search hook to query external databases, or just about anything...

Like I previously mentioned, you can create custom content types in drupal which is a very powerful feature, indeed. But you can then explicitly define that new content type in such a way that the search system understands perfectly how to search it.

You can also extend your site's search capability by hooking into a search query while it's happening, and aggregate all sorts of external sources into the result as well.

Form Generation / Validation

Create forms rapidly



Validate forms easily

Many of you who work on websites know that form generation and form processing can be very tedious, repetitive work.

Drupal offers an extensive forms API which is closely intertwined with the theme engine. That means that you can quickly create forms that use the site's theme without having to worry about a single line of HTML.

You can then have drupal verify the validity of those forms using built-in or custom validation functions.

This is one way that Drupal assists in the pursuit of rapid development.

Menu System

- Menus are complex, even when they're not
- Menus are contextual

Manage menus easily

Create on-the-fly menu options

Button placement can be dynamic

Like everything else in Drupal, the menu system is entirely customizable and dynamic.

Menu items can be added and removed easily. They can even be generated on-the-fly from module code.

Buttons can also be given a weight attribute that can be changed based on runtime criteria, so their placement can be customized based on context.

Pretty cool.

api.drupal.org

Covering Drupal's entire API is beyond the scope of this talk, so I'd suggest you follow up here.

You'll find here the myriad of functions available to you as a developer nicely organized into topics.

I think it's important to remember that because Drupal is open source, the developers are able to present to you the very same functions that they themselves used to build drupal.

You can do just about anything with Drupal

You really can do pretty much anything with Drupal, but I want to bring you back around to the thought that your website is only a component within your larger information architecture.

At the end of the day, we want our data to manifest itself, not just in our website, but throughout the entire organization, whether it be on electronic signage, printed receipts, fundraising letters, self-checks, or computer workstations.

This takes a great deal of forethought and planning. It takes a fair amount of coordination with our vendors.

And it takes a lot of imagination to envision a library where our physical spaces are infused by our electronic resources.

Thank-you.