Using Custom Post Types to Control Access

Building a collaborative web site by segmenting a site into areas of responsibility and control.
Custom Post Types - CPTs

• What
  – Features and Attributes of CPTs
  – How they can be used for access control

• Why
  – The problem we are trying to solve
  – Why CPTs meet our needs

• How
  – Implementing CPTs
  – Simple Case Study
Background

• A bit about the organization
  – All about bicycle touring
  – Tours are developed and run by volunteers
  – Local groups in several various cities across Canada

• What we used to have

• Where we wanted to go
  – Modern web software
  – Break the single person bottle neck
Breaking the Single Person Bottleneck

• Different approaches
  – Make everyone an admin
  – Multiple contributors – central publication
  – Multiple restricted editors

• Restrict access for security purposes

• Restrict access to enhance simplicity
Simple Restricted Dashboard
Simple Restricted Post

Add New Comox Post

First Post

Permalink: http://localhost/cccts/comox-post/first-post/

This is my first Comox Post
Why use CPTs

• Alternative architectures
  – Restricted categories
    • Addresses only security requirements not simplicity
    • Doesn’t fully solve security issues
    • Much less elegant than capabilities
  – Multi-site installation
    • Would probably work
    • Adds unnecessary complexity
Users, Roles and Capabilities

• Users
  – Must have one or more roles

• Roles
  – Standard System Roles
    • Administrator, Editor, Author, Contributor, Subscriber
  – Custom Roles
    • Club Document Editor, Rides Director

• Capabilities
  – read, delete_posts, edit_posts
What are Post Types

• Great Concept – bad name
• Default System Post Types (i.e. non-custom post types)
  – Post
  – Page
  – Attachment
  – Revision
  – Navigation Menu Item
Post Type Common Features

• A set of fields
• A set of capabilities
• A set of taxonomies
• A set of attributes
  – Hierarchical ?
  – Archive page ?
What are Taxonomies

• A list of terms
• Can have a parent (category) or not (tags)
• Associated to 0 or more CPTs (many-to-many)
• Capabilities
  – Manage_terms
  – Edit_terms
  – Delete_terms
  – Assign_terms
Case Study

• Local Editor
  – Comox_editor, Victoria_Editor, Ottawa_editor
  – CPT for each local editor restricts his area of authority

• Club Document Editor
  – CPT to restrict area of authority
  – Custom Taxonomy to categorise different document types

• Ride Director
  – CPT to restrict area AND add custom fields
  – Uses built-in tax taxonomies
Plugins Employed

• Pods – Content Types and Fields
  – “Custom Post Type UI” is a possible alternative

• Members
  – “Roles & Capabilities” component of Pods is an alternative

• Post Type Switcher
  – Essential for converting an existing site

• User Switching
  – Not strictly required, but very useful for testing
Setting up a Local Editor

• Create the CPT “Comox Posts”
  – PODS plugin
• Create the “Comox Editor” role
  – Members plugin
  – Users | new Role
• Setup menus to Reference CPT’s
  – Create a Custom Link menu item
  – URL: site-url/?post_type=comox_post
Create the CPT

Pods are content types that you can customise and define fields for based on your needs. You can choose to create a Custom Post Type, Custom Taxonomy, or Custom Settings Pages for site-specific data. You can also extend existing content types like WP Objects such as Post Types, Taxonomies, Users, or Comments.

Not sure what content type you should use? Check out our [Content Type Comparison](#) to help you decide.

- **Create New**
  
  Create entirely new content types using Post Types, Taxonomies, or Custom Settings Pages.

- **Extend Existing**
  
  Extend any existing content type within WordPress, including Post Types (Posts, Pages, etc), Taxonomies (Categories, Tags, etc), Media, Users, or Comments.
Create the CPT cont.

Creating a new Content Type allows you to control exactly what that content type does, how it acts like, the fields it has, and the way you manage it.

Create a New Content Type

- Content Type: Custom Post Type (like Posts or Pages)
- Singular Label: Comox Post
- Plural Label: Comox Posts
Create the CPT cont.
Create the CPT cont.
Create the CPT cont.
Create “Comox Editor” Role

<table>
<thead>
<tr>
<th>All</th>
<th>Capability</th>
<th>Grant</th>
<th>Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wordpress</td>
<td>edit_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Posts</td>
<td>edit Others_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Pages</td>
<td>publish_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td>read_private_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Pods</td>
<td>delete_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Pod Fields</td>
<td>delete private_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td></td>
<td>delete published_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td></td>
<td>edit_private_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td></td>
<td>edit published_comox_posts</td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

Add new Role
More on Users and Role

• Add the new CPT capabilities to the Administrator

• Create new user “Joe Comox”
  – Optional user for testing
  – Assign “Comox Editor” role

• Confirm Role and CPT are working
  – Switch user to “Joe Comox”
  – Create a new Comox Post
Creating a *Comox Post*
Menu Link to Comox Post Archive
What’s Next?

• Use Pods “duplicate” functionality to create additional similar CPT’s
• Create a corresponding role for each CPT
• Add users to roles
• Add menu items for each CPT archive
• Deploy
Club Document Editor Case Study

• Slightly more complicated example than the Local Editor
• Adds a custom taxonomy to categorize different types of documents
• First we will Create the custom taxonomy using Pods
• Then we will create our “Club Document” CPT as in the previous example
Create a Custom Taxonomy

Creating a new Content Type allows you to control exactly what that content type does, how it acts like, the fields it has, and the way you manage it.

Create a New Content Type

Content Type: Custom Taxonomy (like Categories or Tags)

Singular Label: Document Type

Plural Label: Document Types

Start Over Next Step
Create Taxonomy cont.
Create Taxonomy cont.
Create “Club Document” CPT
Create “Club Document” CPT cont.
Create Role and User

• Add new role: Club Document Editor
  – Assign all *Club Document* capabilities
• Assign all *Club Document* capabilities to administrator
• Create new user: Sally Secretary
  – Assign *Club Document Editor* Role
• Populate our *Document Types* Taxonomy
  – Add: “Board Minutes”, “General Meeting Minutes” and “uncategorized”
Test: Add New Club Document
Why Can’t Sally Select a Document Type?

• Hint: If I switch back to my administrator role everything works fine.
• But we assigned Sally all Club Document capabilities!
• *Document Types* is a taxonomy not a CPT
• Where do we set taxonomy capabilities?
Setting Taxonomy Capabilities

### Edit Capabilities: Taxonomies

<table>
<thead>
<tr>
<th></th>
<th>Capability</th>
<th>Grant</th>
<th>Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>All</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>General</strong></td>
<td>manage_categories</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Posts</strong></td>
<td>assign_categories</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Pages</strong></td>
<td>assign_post_tags</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Media</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>manage_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>edit_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>delete_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>assign_document_type_terms</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
Taxonomy Capabilities

• Grant `assign_document_type_terms` to Club Document Editor role

• Grant all 4 `document_type` taxonomy capabilities to administrator

• Flush the Pods cache:
  – `Pods Admin | Settings | Clear Pods Cache`
Menu Links to Archives
Ride Director Case Study

- Our most complicated CPT
- Numerous custom field using table based storage
  - But that is a topic for another day
- Custom post type that uses the built-in taxonomies *category* and *tags*
Create Rides CPT
Rides: Roles & Capabilities

• Create new “Rides Director” role
  – Grant all Rides capabilities
• Grant all Rides capabilities to Administrator
• Create user “Ron Rider” with Ride Director role.
Test: Add New Ride
We Have Been Here Before

• Why can’t Ron assign *categories* or *tags*?
• Again we have a capability problem
• Same problem (slightly) different solution
  – *Create custom capabilities*
    • *assign_categories, assign_post_tags*
  – *Grant both capabilities to Rides Director & Administrator*
  – *Bind the taxonomy using a filter in functions.php*
  – *Flush the Pods cache*
Creating Custom Capabilities

<table>
<thead>
<tr>
<th>Role</th>
<th>Grant</th>
<th>Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WordPress</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Posts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Appearance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Club</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Edit Capabilities: Taxonomies

<table>
<thead>
<tr>
<th>Capability</th>
<th>Grant</th>
<th>Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td>manage_categories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>assign_categories</td>
<td><strong>✓</strong></td>
<td></td>
</tr>
<tr>
<td>assign_post_tags</td>
<td><strong>✓</strong></td>
<td></td>
</tr>
<tr>
<td>manage_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>edit_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>delete_document_type_terms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>assign_document_type_terms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Custom Capability:

- **assign_post_tags**

Role:

- Users: 1
- Granted: 13
- Denied: 0

Update
Binding Built-in Taxonomy Capabilities

/* In order to use a builtin taxonomy inside a CPT we need
 * to setup capabilities for the taxonomy that are enabled
 * in the role for that CPT
 */
add_action('init', 'set_builtin_tax_caps');
function set_builtin_tax_caps() {
    $tax = get_taxonomy('post_tag');
    $tax->cap->assign_terms = 'assign_post_tags';
    $tax = get_taxonomy('category');
    $tax->cap->assign_terms = 'assign_categories';
}
Summary

• Post Types (built-in or custom) each have an associated set of capabilities.
• By taking advantage of that functionality we can segment our site into discreet areas of responsibility
• Restrict access to only essential functionality
• Simplifies the interface by limiting to a specific area
Hopefully this discussion has provided a better understanding of how Custom Post Type can provide the infrastructure for building a collaborative web site.

QUESTIONS?