E-Dev Documentation

Processing

index.php
- Primary runtime file
- header/header.inc.php
  - Primary function calling file
  - Calls system config file, common function files
    - postgresql.inc.php – database wrapper
    - function.inc.php – generic functions for e-dev
    - module.inc.php – module loading functions for e-dev
    - array_function.inc.php – generic array functions
    - common.inc.php – common functions for application (can vary from app to app
    - date_function.inc.php – date/time manipulation functions
    - form.inc.php – html form creating functions
    - display_function.inc.php – output template creation functions
    - legacyxml.inc.php – XML parsing functions
  - Configures site module layout
  - Connects to database
  - Sets permission defines for checking later
- include/app_preauth.inc.php (optional)
  - Perform pre-authorization processing.
  - This file is setup on a per-app basis
- auth/auth.inc.php
  - Processes the following...
    - If the site requires authorization, it checks to see if the user is already authorized
    - If a user is authorized, it returns their account information
    - It a user tries to login or logout, it processes the information from the login form and logs them in (or denies entry)
  - Auth Failed
    - show_login_form variable is set, user login form is displayed
  - Auth passed
    - Check to see if the user has permissions to access current module
- include/app_postauth.inc.php (optional)
  - Perform post-auth processing
  - File is setup on a per-app basis
- Module Permissions Check
  - Module Permissions Fail
    - Skip processing and echo permissions failure for module
  - Module Permissions Pass
    - Load module files from module directory
    - These files are discussed later
- include/log.inc.php (optional)
  - If access logging is done on the site, this file can be called to perform it
- Theme path is set from config file into THEME_PATH define
- header/left.inc.php (optional)
  - Load modules for the left column of the site. All output content is stored in

$leftColumnContent variable
- header/right.inc.php (optional)
  - Load modules for the right column of the site. All output content is stored in $rightColumnContent variable
- If $show_login_form is set, show the login form, otherwise load the DISPLAY_MODULE file
- If successMessage or errorMessage has been returned from our module, set the siteMessage variable with the appropriate css class for display later
- If there is a navbar.inc.php file in our theme's layout, load it for sit enavigation later
- We have three possible display files, call the appropriate one:
  - theme_path/theme_name/layout/noheader.inc.php
    - Usually used in popup windows. Shows only the center column, no left or right column and no site logo.
  - theme_path/theme_name/layout/logo.inc.php
    - Can be used for login pages. Shows the login form and the logo at the top of the site
  - theme_path/theme_name/layout/body.inc.php
    - The actual site display template. Shows all columns, logos, and the center.
  - All of the display files call required css and javascript files. The css, javascript, and layout files are stored in the theme_path/theme_layout directory

Modules

- Modules are individual components which make up the actual functionality of the application.
- Modules allow for functionality of an application to be sorted on a file-system hiearchy to allow for easier editing.
- Allow for the easy addition, exchange, or removal of features from an application during deployment.
- All modules are located in the modules/ directory. Currently, there are three different types of modules. Center, left, and right column modules.
- Left
  - modules loaded and displayed on the left column of the application. If no module is found, no output is displayed
  - Located under the modules/left/ directory
  - Loaded by the header/left.inc.php file.
  - All output must be stored in the "$leftColumnContent" variable for display by the template
- Right
  - modules loaded and displayed on the right column of the application. Stylesheets may be set to automatically alter the center column to expand and fill the page when no right column is available.
  - Located under the modules/right/ directory.
  - Loaded by the header/right.inc.php file
  - All output must be stored in the "$rightColumnContent" variable for display by the template
- Center
  - The center column of the page
  - located under the modules/center/ directory

    - loaded by index.php
    - All output must be stored in the "$siteContent" variable for display by the template.
- Common
    - Located in the modules/common/ directory
    - These are functions or includes used by more than one module in the system. They can be placed here for easy access by other modules.
    - Generally, functions are placed here that are used by two or more modules, but not used frequently enough to be migrated to the include/common.inc.php file.

Module Files

- Files loaded when a module is called.
- All are optional. No error will occur if a module does not exist
- function.php – Contains functions used by the module. This is not a class, so all functions are still in the global space
- javascript.js – Contains javascript functions used by the module. Including your javascript here instead of inline in display.php will cause the javascript to be cached by the browser
- stylesheet.css – Contains stylesheet information used by the module. Including your css here instead of inline in display.php will cause the stylesheet to be cached by the browser
- display.php – Responsible for formation of html to be passed to the display template. All html code must be stored in the "$siteContent" variable.
- Image – An image may be placed in the module directory for display in module lists. This will be discussed in more detail later.

Module Images

- Currently used to display module icons in module lists
- Not called module.png because of browser caching
- Image Locations, in order of preference:
    - <module_path>/<link_name>.png (in module directory)
    - THEME_PATH/images/modules/<link_name>.png (current theme)
    - themes/default/images/<link_name>.png (default theme)

Module.xml

- XML Configuration file for module
- Format:

```
<module>
        <parameter1>Value</parameter1>
        <parameter2>Value</parameter2>
</module>
```

- Parameters
    - module_name – Proper name for module. Used when displaying a module to a user
    - module_description – Description of a module used when displaying module lists

- link_name – Reference to module. This is used when calling a module in the url using the "module=link_name". The link_name must also match the folder name the module resides in
- permissions – Permissions that must be met by the user before the module can be viewed. This tag is optional, and there can be multiple occurrences within the same module file. The settings must match the defines used in the permissions.xml file
- custom_perm – If a second set of permissions is created which are not editable in the app, they would be set here. See custom permissions for more information
- hidden – Whether or not a module is hidden from a user when displaying a module list. This does not prevent a user from accessing the module
- auth_only – Prevents a user from accessing a module unless they are logged into the application
- perm_error (depreciated) – The error message to be displayed when a user does not have permissions to access a module
- sort_order – The order the module appears in when displaying a module list

Module Loading & Caching

- Ordinarily, a function is run when the app is first accessed by a user to load the site's module layout.
- This function scans the appropriate module directory for all files named module.xml, and loads the module properties into two arrays.
- The layout is stored in two session variables:
  - $_SESSION["siteModList"]
    - stores the modules in the $_SESSION["siteModList"]["property_name"]["link_name"] layout where link_name corresponds to the "link_name" property set in the module's module.xml
    - Good for creating site module lists or finding a set property for all modules
  - $_SESSION["siteModInfo"]
    - stores in the modules in the $_SESSION["siteModInfo"]["link_name"]["property_name"] layout where link_name corresponds to the "link_name" property set in the module's module.xml
    - Good for returning all properties of a particular module.
  - If a module's properties are changed or a module is added/removed, the user must login/logout or close the browser window to queue the site loading function to run again (because the session has been deleted)
- For higher-traffic sites, the module layout can be cached into a single file. This prevents the initial file system scan for all.
- The cache file is named "module-cache.xml" and is created in the root directory of the module type (I.E., a module-cache.xml is created under modules/center, modules/left, and modules/right.
- To create the cache, just run the scripts/createcache.php file
- To recreate the cache and load any module changes into it, just rerun the above script. You can delete the cache files at any time to go back to the old method.

Permissions

There are two different types of permissions available to be set on a module, generic permissions and custom permissions. Generic permissions are used at any point in the application. Custom permissions are used when a user is accessing an object through a view or property update.

Generic permissions are permissions set directly on an account in the account profile (Insert Object, Manage Users, etc). These are listed in the config/permissions.xml file. They are referenced in the module.xml file by using the value in "define_name". They are referenced with the tag <permissions></permissions> in module.xml. For example, to only allow an administrator to access a certain module, I would add "<permissions>ADMIN</permissions> to that module's module.xml file.

Custom Permissions are permissions for an account on an object, in this case a file, collection, or url. These can be set directly in the object's permissions page. Some are also inherited based on the status of the object. These are set in the "include/app_postauth.inc.php" file only if an object is being accessed by a user. This could be during a view or property adjustments. These are listed in "config/customperm.xml", and are set via the tag <custom_perm></custom_perm> in moudle.xml.

It should be noted that a change in permissions will only be recognized after a log off/log on. Specifying multiple permissions of either type will act in an "OR" fashion. For example, setting ADMIN and MANAGE_USERS in <permissions> will allow a user with either to access a module. However, if you set permissions and custom_perm, both requirements must be met before the user can access the module.

License