Drupal7Eshop

php-Drupal7 - 7.8

version :1.0.0-SNAPSHOT

Drupal is a general-purpose server-side scripting language originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications.PHP can be deployed on most web servers and as a standalone interpreter, on almost every operating system and platform free of charge.

COREMODULE

file

The File module enables you to upload and attach files to content and to manage these uploads if you have the appropriate permissions. This module is responsible for validating file content and managing uploaded files. It also provides options for displaying file content. As a site administrator you will be able to control what type of files can be uploaded and their maximum size.File module provides its functionality by defining a File field type for the Field module

options

Defines selection, check box and radio button widgets for text and numeric fields.

field_sql_storage

The Field SQL Storage module provides data management for the Field module. This module uses the Field API to generate dynamic data tables for entities and fields. While this is a required core module, it may be replaced by other contributed modules that implement The Field Storage API. The Field SQL Storage module provides data management for the Field module. This module uses the Field API to generate dynamic data tables for entities and fields. While this is a required core module, it may be replaced by other contributed modules that implement The Field Storage API.

Field Module :-The Field module allows custom data fields to be attached to Drupal entities (content nodes, users, taxonomy vocabularies, etc.) and takes care of storing, loading, editing, and rendering field data. Most users will not interact with the Field module directly, but will instead use the Field UI module user interface. Module developers can use the Field API to make new entities "fieldable" and allow fields to be attached to their entities.

Field API:-The Field API allows custom data fields to be attached to Drupal entities and takes care of storing, loading, editing, and rendering field data. Any entity type (node, user, etc.) can use the Field API to make itself "fieldable" and thus allow fields to be attached to it. Other modules can provide a user interface for managing custom fields via a web browser as well as a wide and flexible variety of data type, form element, and display format capabilities.

The Field API defines two primary data structures, Field and Instance, and the concept of a Bundle. A Field defines a particular type of data that can be attached to entities. A Field Instance is a Field attached to a single Bundle. A Bundle is a set of fields that are treated as a group by the Field Attach API and is related to a single fieldable entity type.

Field Module :-The Field module allows custom data fields to be attached to Drupal entities (content nodes, users, taxonomy vocabularies, etc.) and takes care of storing, loading, editing, and rendering field data. Most users will not interact with the Field module directly, but will instead use the Field UI module user interface. Module developers can use the Field API to make new entities "fieldable" and allow fields to be attached to their entities.

Field API:-The Field API allows custom data fields to be attached to Drupal entities and takes care of storing, loading, editing, and rendering field data. Any entity type (node, user, etc.) can use the Field API to make itself "fieldable" and thus allow fields to be attached to it. Other modules can provide a user interface for managing custom fields via a web browser as well as a wide and flexible variety of data type, form element, and display format capabilities.
The Field API defines two primary data structures, Field and Instance, and the concept of a Bundle. A Field defines a particular type of data that can be attached to entities. A Field Instance is a Field attached to a single Bundle. A Bundle is a set of fields that are treated as a group by the Field Attach API and is related to a single fieldable entity type.

field
The Field API allows custom data fields to be attached to Drupal entities and takes care of storing, loading, editing, and rendering field data

text
Defines simple text field types.

EXTERNALMODULE

Chaos tool suite
This suite is primarily a set of APIs and tools to improve the developer experience. It also contains a module called the Page Manager whose job is to manage pages. In particular it manages panel pages, but as it grows it will be able to manage far more than just Panels.A library of helpful tools by Merlin of Chaos.

Rules
The rules module allows site administrators to define conditionally executed actions based on occurring events (known as reactive or ECA rules). It's a replacement with more features for the trigger module in core and the successor of the Drupal 5 workflow-ng module.The rules modules allows site administrators to define conditionally executed actions based on occurring events

Views
This module provides a flexible method for Drupal site designers to control how lists and tables of content are presented.

Entity API
It enables modules to work with any entity type and to provide entities.

This module extends the entity API of Drupal core in order to provide a unified way to deal with entities and their properties. Additionally, it provides an entity CRUD controller, which helps simplifying the creation of new entity types.

Features

This is an API module, so it doesn't provide any end-user features. However, users may enable the entity tokens module, which provides token replacements for all entity properties that have no tokens and are known to the entity API.

Overview

The module provides API functions allowing modules to create, save, delete, view or to determine access for any entity, i.e. entity_create(), entity_save(), entity_delete(), entity_view() and entity_access().

The entity API introduces a unique place for metadata about entity relationships and entity properties: hook_entity_property_info(). This information about entity properties contains the data type and callbacks for how to get and set the data of a property. Modules may rely on this information in order to support any entity property, e.g. Rules and the Search API build upon that.

Furthermore the module provides data wrappers that make use of the available information to provide a simple and unified access to entities and their properties. For usage examples have a look at the README or the provided tests.

Beside that, the module helps you defining a new entity type. For that, it provides an entity controller, which implements full CRUD functionality for your entities. Optionally, entities may be created based on classes derived from the provided Entity class.

The entity API cares about creating fieldable entities as well as exportable entities, however it does not yet support revisions. Additionally it supports implementing bundle entities, i.e. bundle objects (like node types) for fieldable entities implemented as (exportable) entities, for which the appropriate field API callbacks get automatically invoked.

For entity types implemented based upon the provided CRUD API the API is providing additional module integration too, i.e. Rules events are provided for all CRUD-related hooks, some basic entity property information for hook_entity_property_info() is provided and exportable entities are automatically integrated with the Features module.

These module integrations are implemented in separate controller classes, which may be separately overridden or enabled/deactivated.
Optionally, the entity API also helps providing an administrative interface for managing entities, e.g. the UI for managing profile types of Profile 2 is built with that.

Taxonomy Manager

This module provides a powerful interface for managing taxonomies. A vocabulary gets displayed in a dynamic tree view, where parent terms can be expanded to list their nested child terms or can be collapsed.

The Taxonomy Manager has following operations and key features:

- dynamic treeview
- mass deleting
- mass adding of new terms
- moving of terms in hierarchies
- merging of terms (using the Term merge module in 7.x)
- fast weight changing with up and down arrows (and AJAX saving)
- AJAX powered term editing form
- simple search interface
- CSV Export of terms
- i18n support for multilingual vocabularies (per language terms)
- Double Tree interface for moving terms in hierarchies, adding new translations and switching terms between different vocabularies

For using the Taxonomy Manager you should have JavaScript and automatically load of images enabled in your browser.

This is a Google Summer of Code 2007 project. Read my proposal and my status reports for more information.

Taxonomy Manager Drupal 5 and JQuery Upadate module:
In case you are using the JQuery Update module v2.x on your D5 installation, the Taxonomy Manager won't work. See this issue for more information.
To fix the problem download the tree.js file and replace it with the one included in the taxonomy manager folder under /js.

Taxonomy Manager Drupal 6 and Panels 3 / Taxonomy Breadcrumb:
Taxonomy Manager and other taxonomy modules might conflict, if both try to override the taxonomy/term/% path. Taxonomy Manager uses this to redirect previously merged terms to their new term. If you do not need this feature or if you want to used Panels 3 / Taxonomy Breadcrumb, you can disable it under Administer > Site Configuration > Taxonomy Manager. See this issue for more information.
Note: since 6.x-2.1, the redirect of taxonomy/term is disabled by default

Taxonomy subterms

Taxonomy Subterms is a simple module to force display child taxonomy terms on the parent taxonomy page of nodes belonging to the current term or/and the children of the current term.

Ubercart

Ubercart out of the box is a great solution for stores that sell physical goods, digital product downloads, and site memberships.

shopmenu

Provideds shopping cart menu and contains ubercart module altered forms.

the "menu" hook (shop_menu). For each merchant from whom I'm using their datafeed, I make code like this in the shop_menu function: function shop_menu($may_cache) { $items = array(); if ($may_cache) { ....