*Pubblicato da Drupal Italia (**www.drupalitalia.org**) per gentile concessione di Antonio Santomarco*

Capitolo 2.

2. Creazione di moduli per Drupal: un tutorial.

2.1. Introduzione.

Come detto in precedenza, un modulo non è altro che un insieme di funzioni che forniscono nuove funzionalità a Drupal o adattano quelle già esistenti ai propri scopi. Questo tutorial descrive come iniziare a creare moduli; naturalmente esso non copre tutti gli aspetti della programmazione di moduli, ma vuole essere un punto di partenza per entrare nel meccanismo di progettazione. Nell'esempio che segue verrà creato un modulo che mostra un blocco all'interno del quale sono visualizzati dei links a contenuti creati negli ultimi sette giorni e che si chiamerà "*lastweekposts*".

La prima cosa da fare è quella di creare un nuovo file PHP con un qualsiasi editor di testo e di salvarlo con il nome che si vuole dare al modulo (nell'esempio: "*lastweekposts*") e con estensione "*.module*": questa, infatti, è l'unica estensione che Drupal riconosce per i suoi moduli; il file, in definitiva, si

chiamerà "*lastweekposts.module*". All'interno di tale file, tutte le funzioni php che verranno create saranno chiamate "*lastweekposts_hook()*" dove "hook" dovrà essere di volta in volta sostituito col nome effettivo della funzione (ulteriori dettagli sono stati forniti nel capitolo 1 del presente lavoro).

2.2. Primo obiettivo: descrivere il modulo.

Il primo hook che è bene scrivere è quello che permette a Drupal di avere informazioni riguardo al modulo che si sta scrivendo: tale hook si chiamerà "*lastweekposts_help()*"; il codice per tale hook è il seguente:

```
<?php

function lastweekposts_help($section='') {
  $output = '';
  switch ($section) {
  case "admin/modules#description":
    $output = t("Displays links to nodes created on the last week");
    break;
  }
  return $output;
}

?>
```

La variabile *"$section"* individua il contesto nel quale sarà visualizzata la stringa di testo presente all'interno della variabile *"$output"*: nel caso dell'esempio tale stringa viene visualizzata nella pagina che descrive i compiti dei moduli, ovvero *"admin/modules#description"*. Il modo più opportuno per processare la variabile "$section" è quello di utilizzare un costrutto "*switch-case*"; in tal modo è facile aggiungere altri contesti in cui mostrare altre stringhe d'informazione/aiuto: in

particolare *"admin/help#lastweekposts"* visualizzerà un output sulla pagina d'aiuto principale individuata appunto dall'URL *"admin/help"*.

2.3. Chi può usare il modulo: i permessi.

La funzione successiva da scrivere è quella che definisce i permessi; tale funzione non assegna i permessi, ma semplicemente definisce quali permessi sono disponibili per il modulo all'interno del quale essa si trova: l'accesso ai contenuti basato su tali permessi sarà configurato altrove. Il codice per tale funzione è il seguente:

```
function lastweekposts_perm() {
  return array('access content', 'access lastweekposts', 'administer lastweekposts');
}
```

Ovviamente tale codice va inserito tra i tag php già presenti nel file. Un accorgimento importante è quello di usare stringhe di permesso uniche all'interno del modulo per evitare che vengano mostrati più volte gli stessi permessi nella pagina dei permessi (*"admin/access"*; dalla barra di navigazione scegliere *"administer"* e poi *"access control"*). E' utile anche che tali stringhe contengano il nome del modulo a cui fanno riferimento per evitare conflitti con altri moduli.

2.4. Creazione dei menu.

Affinché Drupal sia a conoscenza di ogni funzione, c'è bisogno di creare un hook che permetta di definire l'associazione tra un'URL e la funzione che dovrà creare il contenuto per quella stessa URL: l'hook che permette di fare ciò è "*lastweekposts_menu()*". Il codice php per tale modulo è il seguente:

```
function lastweekposts_menu() {
  $items = array();
  $items[] = array('path' => 'lastweekposts',
                   'title' => t('last week posts'),
                   'callback' => '_lastweekposts_all',
                   'access' => user_access('access content'),
                   'type' => MENU_CALLBACK);
  return $items;
}
```

Quello che si sta indicando a Drupal è che se l'utente va alla pagina *"lastweekposts"*, il contenuto generato da "*_lastweekposts_all()*" sarà visualizzato nella stessa pagina che avrà come titolo *"last week posts"*. La voce '*type*' associata a "*MENU_CALLBACK*" fa in modo che il link alla pagina *"lastweekposts"* non sia visualizzato nel menu dell'utente, ma, nel momento in cui si accede a tale pagina, sia visualizzato il contenuto di cui sopra. Se si desidera visualizzare anche il link nel menu all'interno del blocco di navigazione bisogna usare "*MENU_LOCAL_TASK*".

2.5. I blocchi.

Molto spesso è utile (ma non indispensabile) creare anche i cosiddetti blocchi: essi sono mostrati sul lato sinistro o destro della pagina e contengono

piccoli contenuti inerenti al modulo cui fanno riferimento. L'hook che si occupa di definire tali blocchi è *"lastweekposts_block()"* e il suo codice è il seguente:

```
function lastweekposts_block($op='list', $delta=0) {
// listing of blocks, such as on the admin/block page
  if ($op == "list") {
    $blocks[0]["info"] = t('Last Week Posts');
    return $blocks;
  } else if ($op == "view" && user_access('access lastweekposts')){
    switch ($delta) {
      case 0:
        $block['subject'] = 'Last Week Posts';
        $block['content'] = _lastweekposts_block_content();
        break;
    }
    return $block;
  }
}
```

Il contenuto del blocco viene in genere creato richiamando una funzione che si occupa solo di questo: nell'esempio bisogna quindi implementare *"_lastweekposts_block_content()"*; prima di questo, però, diamo un'occhiata ai parametri in ingresso all'hook; il parametro *"$op"* descrive il tipo di operazione da effettuare: se il suo valore è *'list'* (valore di default), viene mostrata la lista dei blocchi del modulo in questione nella pagina *"admin/block"*; se invece il suo valore è *'view'*, allora il blocco viene visualizzato a sinistra o a destra della pagina (se la visualizzazione è stata precedentemente abilitata in *"admin/block"* e se si dispone dei permessi necessari); il parametro *"$delta"* invece definisce il numero del blocco: è possibile infatti definire anche più blocchi all'interno della funzione e ognuno di essi è individuato da un numero a partire da zero (che è il valore di default). Nel caso in cui il valore di "$op" è '*view*', la funzione deve ritornare un array con gli elementi *'subject'* e *'content'*: se non s'includono entrambi questi

elementi il blocco non sarà visualizzato correttamente. "$op" può assumere altri due valori che sono *'configure'* e *'save'*: entrambi servono per la configurazione dei blocchi (per maggiori dettagli si visiti http://drupaldocs.org/). Tornando al contenuto del blocco, l'obiettivo nell'esempio è quello di visualizzare una serie di links che si riferiscono a contenuti generati nell'ultima settimana. Come descritto nel capitolo 1, tutti i contenuti sono memorizzati all'interno del database sotto forma di "nodi"; ad ogni nodo è inoltre associata la data e l'ora della sua creazione. Per raggiungere lo scopo, allora, bisognerà effettuare un'interrogazione al database e sfruttare i dati riguardanti proprio la data e l'ora. La funzione è la seguente:

```
function _lastweekposts_block_content() {
  $block_content = '';
  // Get today's date
  $today = getdate();
  // calculate midnight one week ago
  $start_time = mktime(0, 0, 0,$today['mon'],($today['mday'] - 7), $today['year']);
  // we want items that occur only during the last week, so calculate 7 days
  $end_time = $start_time + 604800;
  // 60 * 60 * 24 * 7= 604800 seconds in a week
  $query = "SELECT nid, title, created FROM " . "{node} WHERE created >= '" . $start_time
."' AND created <= '". $end_time . "'";
  // get the links
  $queryResult = db_query($query);
  while ($links = db_fetch_object($queryResult)) {
    $block_content .= l($links->title, 'node/'.$links->nid) .'</a><br>';
  }
  // check to see if there was any content before setting up the block
  if ($block_content == '') {
    // no content from a week ago, return nothing.
    return;
  } else {
    return $block_content;
  }
}
```

Si noti che il codice della funzione precedente è indipendente da Drupal (ad esclusione della chiamata alla funzione *"l()"* all'interno del costrutto *"while"*); inoltre con la query utilizzata si seleziona ogni tipo di contenuto (blog, forum,

pagine...) presente nella tabella "*node*" che è la tabella principale per i contenuti di Drupal; per questo tutorial d'esempio la cosa è accettabile, ma nei moduli reali è opportuno selezionare solo i contenuti inerenti al modulo in questione aggiungendo la colonna '*type*' e una clausola '*WHERE*' che faccia il check su tale colonna. Altra cosa importante da notare è che la funzione non si occupa dell'apertura e chiusura della connessione col database: ciò non è necessario perchè di questi compiti si occupa direttamente Drupal grazie al livello di astrazione dal database di cui al capitolo 1.

2.6. Configurazione del modulo.

Le funzioni precedenti sono sufficienti per il funzionamento del modulo; è utile, però, aggiungere un'ulteriore funzione che permette di configurare il modulo affinché il suo comportamento risponda alle proprie esigenze. Questa funzione è *"lastweekposts_settings()"* ed è opportuno che l'accesso a tale sezione sia riservato soltanto agli amministratori. Nel caso in esame, verrà semplicemente definito il numero massimo di links da mostrare all'interno del modulo. L'hook da inserire è il seguente:

```
function lastweekposts_settings() {
  // only administrators can access this module
  if (!user_access("admin lastweekposts")) {
    return message_access();
  }
  $output .= form_textfield(t("Maximum number of links")," lastweekposts_maxdisp",
                            variable_get("lastweekposts_maxdisp", "5"), 2, 2,
                            t("The maximum number of links to display in the block."));
  return $output;
}
```

La funzione genera un form che permette all'amministratore di fissare il numero di links da mostrare all'interno del blocco. Non ci si deve preoccupare di creare il form: questo lavoro è fatto automaticamente da Drupal grazie a *"form_textfield()"*; accanto a questa esistono tante altre funzioni del core che permettono di creare in automatico altri tipi di campi per i form (per maggiori dettagli, è utile, ancora una volta, consultare http://drupaldocs.org/). Inoltre nella funzione viene utilizzata *"variable_get()"* che permette di recuperare il valore della variabile di configurazione di sistema chiamata *"lastweekposts_maxdisp"*; se tale variabile non esiste essa sarà creata e le sarà assegnato un valore di default pari a 5. Affinché queste modifiche abbiano effetto, bisogna anche modificare la funzione che si occupa della creazione del contenuto del blocco: in particolare bisogna aggiungere la riga

```
$limitnum = variable_get("lastweekposts_maxdisp", 5);
```

prima di

```
$query = "SELECT...
```

e modificare quest'ultima nel modo che segue:

```
$query = "SELECT nid, title, created FROM " ."{node} WHERE created >= '" . $start_time .
        "' AND created <= '". $end_time . "' LIMIT " . $limitnum ;
```

A questo punto, dopo aver abilitato il modulo (vedi oltre), è possibile configurare le impostazioni per il modulo andando alla pagina *"admin/settings/lastweekposts"*. Nei moduli reali è opportuno prevedere forme di validazione dell'input: nell'esempio, infatti, se fosse inserita una lettera al posto di un numero all'interno del form la funzione per la creazione del contenuto del blocco non funzionerebbe correttamente.

2.7. Creazione di pagine.

A questo punto la domanda più ovvia che potrebbe venire in mente è: "Non sarebbe utile mostrare i links anche in una pagina, oltre che nel blocco, in modo da non limitarne il numero?" In effetti questo è il passo successivo. Per espletare questo compito bisogna aggiungere la funzione *"_lastweekposts_all()"* che è apparsa nella funzione che si occupa della creazione dei menu (si noti che il suo nome comincia con il simbolo "_"; questa è una convenzione interna a Drupal la quale distingue le funzioni che non devono essere richiamate all'esterno del modulo in cui sono scritte). Il codice di tale funzione è il seguente:

```
function _lastweekposts_all() {
  // content variable that will be returned for display
  $page_content = _lastweekposts_page_content();
  // check to see if there was any content before setting up the block
  if ($page_content == '') {
    // no content from a week ago, let the user know
    print theme("page", "No posts left on this site during last week.");
    return;
  }
  print theme("page", $page_content);
}
```

Da notare l'utilizzo di *"theme()"*: questa funzione fa parte del core di Drupal e serve a formattare le pagine da visualizzare (http://drupaldocs.org/ per i dettagli). La funzione *"_lastweekposts_page_content()"*, tranne che per la variabile *"$query"*, è praticamente uguale a *"_lastweekposts_block_content()"*; sarebbe opportuno quindi fare in modo di utilizzare la stessa funzione per creare sia il contenuto del blocco che quello della pagina: questo obbliga ad effettuare alcune modifiche alla funzione che esulano dagli scopi del presente tutorial e pertanto sono lasciate alle abilità di programmazione del lettore. Si riporta di seguito anche il codice di *"_lastweekposts_page_content()"*:

```
function _lastweekposts_page_content() {
  $page_content = '';
  // Get today's date
  $today = getdate();
  // calculate midnight one week ago
  $start_time = mktime(0, 0, 0,$today['mon'],($today['mday'] - 7), $today['year']);
  // we want items that occur only during the last week, so calculate 7 days
  $end_time = $start_time + 604800;
  // 60 * 60 * 24 * 7= 604800 seconds in a week
  $query = "SELECT nid, title, created FROM " . "{node} WHERE created >= '" . $start_time
."' AND created <= '". $end_time . "'";
  // get the links
  $queryResult = db_query($query);
  while ($links = db_fetch_object($queryResult)) {
    $page_content .= l($links->title, 'node/'.$links->nid) .'</a><br>';
  }
  // check to see if there was any content before setting up the page
  if ($page_content == '') {
    // no content from a week ago, return nothing.
    return;
  } else {
    return $page_content;
  }
}
```

Un'ultima cosa da fare per completare il modulo e dare un senso all'elemento creato nella funzione *"lastweekposts_menu()"* è quella di aggiungere nel blocco un link che richiami la pagina *"lastweekposts"* e di conseguenza permetta l'esecuzione di *"_lastweekposts_all()"*. Per fare ciò basta aggiungere il seguente codice nella funzione che crea il contenuto del blocco, subito dopo il costrutto *"while"*:

```
// add in our block a 'more' link that displays the page with all the links
$block_content .= "<div class=\"more-link\">".l(t("more"), "lastweekposts", array("title"
=> t("More posts."))."</div>";
```

2.8. Installazione e abilitazione del modulo.

Il modulo è completo: non ci resta che installarlo ed abilitarlo in Drupal. Per installare un modulo in Drupal basta copiare il file con estensione "*.module*" nella cartella "*modules*", presente nella cartella d'installazione principale di Drupal, oppure in una sua sottocartella. Il passo successivo è quello di abilitare il modulo: per farlo bisogna accedere a Drupal come amministratore, raggiungere la pagina *"admin/modules"* (dalla barra di navigazione scegliere *"administer"* e poi *"modules"*), cercare nella lista il modulo che si desidera abilitare e mettere il segno di spunta nella casella relativa a tale modulo; dopodichè bisogna salvare le modifiche effettuate utilizzando l'apposito pulsante in fondo alla pagina. In seguito, dato che il modulo creato in questo tutorial contiene un blocco, bisogna

raggiungere la pagina *“admin/block”* (*“administer”* e poi *”blocks”* nella barra di navigazione) e abilitare il blocco relativo al modulo scegliendo anche il lato della pagina sul quale farlo visualizzare e la posizione rispetto agli altri blocchi abilitati (questo viene fatto dando un “peso” al blocco: da -10 per farlo visualizzare in alto a +10 per farlo visualizzare in ultima posizione); infine bisogna salvare la configurazione con l’apposito tasto in fondo alla pagina. A questo punto il lavoro è concluso: non resta che provare il funzionamento del modulo andando in una qualsiasi altra pagina: se nel sito esiste del contenuto creato non più di una settimana fa, nel blocco si potranno vedere alcuni dei links a questi contenuti; inoltre cliccando sul link *“more”* sarà visualizzata la pagina con i links a tutti i contenuti in questione.

2.9. Conclusioni.

Allo stato attuale il lettore dovrebbe essere in grado di cominciare a scrivere dei moduli in maniera autonoma che rispondano alle proprie personali esigenze; come detto in precedenza, nel presente tutorial sono stati affrontati solamente argomenti basilari; per approfondire si possono trovare dei manuali sul sito ufficiale di Drupal (http://drupal.org) e la lista di tutte le funzioni native di Drupal e il loro modo d’uso al sito http://drupaldocs.org.