Form API 2.0: The Cheat Sheet

(For Drupal 5 and later)

FormAPI Functions

drupal_get_form(*$form_id, [...]*)
Retrieves and builds the specified form, then validates, processes, and displays it as appropriate.

drupal_execute(*$form_id, $form_values, [...]*)
Retrieves the specified form, populates it with the *$form_values*, then validates and processes it.

form_set_value(*$field, $value*)
Change the submission value of a given form *$field*. Should only be used in a *form_id*_validate() function.

form_set_error(*$field_name, $message*)
Mark a given form field as containing an error.

form_get_error(*$field*)
Get the error(s) for a specific form field.

form_get_errors()
Get all errors for the form currently being processed.

FormAPI Callbacks and Hooks

***form_id*_validate($form_id, $form_values)**
Check form input for errors, then call form_set_error() if any are found.

***form_id*_submit($form_id, $form_values)**
Process input for a given form (called if no errors are found). Should return a path to redirect to when submission is complete.

theme_*form_id*($form_id, $form)
Override the default for a given collection of form elements. Should return the fully-rendered HTML to represent the contents of the *$form* parameter.

hook_form_alter(*$form_id, $form*)
Add, remove, or modify fields for any form handled by Drupal. Check *$form_id* to see whether the form you're given is the one you want to modify.

hook_forms(*$form_id, [...]*)
Return a list of form_ids implemented by your module, and the functions to use when building and validating them. Most modules will return a static list of forms they implement, but a module generating forms on the fly may examine the incoming *$form_id*, responding with data when it sees a *$form_id* it can handle.

Sample FormAPI Definition

Drupal forms are defined as structured arrays, then processed.

```
function mymodule_form_page() {
  return drupal_get_form('mymodule_form');
}

function mymodule_form() {
  $form = array();
  $form['fields']['field1'] = array(
    '#type' => 'textfield',
    '#title' => 'A text field',
    '#description' => 'Enter text here',
    '#default_value' => 'Untitled node',
  );
  $form['fields']['field2'] = array(
    '#type' => 'radios',
    '#title' => 'Radio buttons',
    '#options' => array(
      1 => 'Option 1',
      2 => 'Option 2',
      3 => 'Option 3'
    ),
    '#default_value' => 1,
  );
  $form['fields']['#tree'] = TRUE;

  $form['buttons']['submit'] = array(
    '#type' => 'submit',
    '#title' => 'Submit this form'
  );
  return $form;
}
```

Standard Drupal Element Types

Basic HTML elements:
textfield, textarea, password, select, radios, checkboxes, radio, checkbox, file, hidden, fieldset, submit, button.

password_confirm
Presents two password fields, and throws a validation error if the data entered into both fields doesn't match.

weight
Presents a combo box of numbers from -10 to 10. Useful for specifying Drupal sorting weights.

date
Presents day/month/year options with date validation.

value
Embeds a value in a form defintion that *does not* appear in the final rendered HTML, but can be read by other form manipulation functions (validation, submission, form_altering, etc.)

FormAPI "Magic Properties"

#submit
A list of functions to be called to process the form after successful validation. Each function name should be the key to an array of parameters. For example:

```
$form['#submit']['my_function'] = array();
```

#validate
A list of functions to be called when validating form submissions. Follows the same format as the #submit property.

#tree
Preserve the hierarchial structure of the form's fields when building the *$form_values* array. Defaults to FALSE, and can be applied to the entire form or to specific form elements.

#parents
A list of form element ids that controls where an element's value appears in the *$form_values* collection if **#tree** is TRUE. Handled automatically, but overridable.

#post
Holds the the current set of $_POST values for the form.

#theme
The name of the theming function used to render a form array to HTML. Can be set for the entire form, or for individual form elements. Defaults to theme_*form_id*().

#base
When set, this value is used instead of *$form_id* to locate the default validation, submission, and theming functions for the form.

#value
A hard-coded value that overrides user input entered into a form element. Applies only to individual elements, and should not be confused with #default_value.

#parameters
A saved copy of the parameters passed into the form's builder function. Useful in hook_form_alter().

#multistep
Activates more complex validation code for forms that submit to themselves repeatedly (multi-page wizards or forms with'click for more fields' buttons, for example).

#redirect
Overrides the redirection path returned by the *form_id*_submit() function. Can be set to FALSE to prevent redirection entirely.