

Rules

New opportunities for site builders!

Outline

- Module overview - What is it and why do I need it?
- Advanced features: Rule Sets and scheduling
- Usage example: Build a publishing workflow
- How modules can use the rules API to extend it.
- Benefits of developing with rules
- Current state of the module
- What's next?

Who am I?

- Wolfgang Ziegler aka fago
- Part-time drupal developer from Vienna, Austria
- Studying "Information & Knowledge Management" and "Computational Intelligence" at the Vienna University of Technology
- Joined the drupal community in 2006
(Google Summer of Code project "Node Profile")
- Contributed modules: Content Access, Auto Nodetitle, Fieldgroup, Workflow-ng, Content Profile, Node Profile, Node Family, ..

Module overview – What is it?

- define conditionally executed actions based on occurring events (ECA-rules)
- a replacement with more features for the trigger module in core
- the successor of the drupal 5 compatible workflow-ng module

Triggered Rules

- Users surfing on your drupal site generate events.
- When the event is triggered associated rules are evaluated.
- E.g. rules:
 - “After saving new content” show a message to the user.
 - When a “User has logged in” and the “User has not the role admin”, “Redirect him to 'dashboard'”

Some features...

- Import / Export
- a flexible scheduling system
- a modular input evaluation system (token, ..)
- Grouping rules in “Rule Sets”
- developed with performance in mind
- A well documented and solid API, which allows modules to
 - provide further conditions, actions and events
 - configure default rules and rule sets

Example

- Inform users, when their content has been edited by another user!
- Let's do it!

Variable based configuration

- Actions and conditions need some specified arguments to work with, e.g. some content and a user account
- Events provides these arguments, e.g.
 - the updated content
 - the acting user
 - the author of the updated content
- So you can configure every condition and action for an event, if the needed arguments are available!
- Actions can provide new variables, which can be used as argument too!

Features: Input evaluation

- Rules comes with support for token replacements and PHP input evaluation
- You can use input evaluation in every textfield!
- Input evaluators can make use of all available variables like
 - the updated content
 - the acting user
 - the author of the updated content
 - the unchanged content
 - ..

Features: Rule Sets

- similar in concept to subroutines
- can be easily invoked by actions or modules
- rule sets always works upon some specified arguments, with which its rules can operate
- Define rule sets for common tasks and invoke it by action out of triggered rules when needed!

Features: Scheduling

- rules scheduler module
- Provides a new action for each rule set, which allows you to schedule the rule set execution.
- Specify dynamic scheduling dates and repeated tasks!
- Once the schedule date is reached, your rule set is invoked through cron.
- Schedule everything, by moving it into a rule set and scheduling the execution of this set!

Example: Make use of scheduling!

- A simple publishing workflow for jobs
 - Users may create job
 - Users may control whether their jobs are published
- When users leave jobs unpublished, we send them a “Reminder” that they have an unpublished job by mail!
 - A rule set is scheduled to be executed one day after the job has been created.
 - When the job isn't published, the reminder is sent and the rule-set re-scheduled.
 - So the user gets daily reminders, until he publishes or deletes the job.

Site building with rules

- Quickly build new functionality or customize your site by adding some rules
- Import/Export helps staging your rules to production sites and allows you to share your rules with others!
- Easily execute custom short code snippets with rules!
- Write new functionality by exposing new conditions, actions, events, ...

Integrate your module with rules

- Use the rules API to
 - provide further conditions, actions
 - provide further events with dynamic loading of available arguments
 - configure default rules and rule sets
 - provide new data types
 - provide further input evaluators

Example action: Set the content author

```
<?php
/**
 * Implementation of hook_rules_action_info
 */
function yourModule_rules_action_info() {
    return array(
        'yourModule_action_node_set_author' => array(
            'label' => t('Set the content author'),
            'arguments' => array(
                'node' => array('type' => 'node', 'label' => t('Content')),
                'author' => array('type' => 'user', 'label' => t('User, which is set as author')),
            ),
            'module' => 'Node',
        ),
    );
}

/**
 * Action: Sets the node author
 */
function yourModule_action_node_set_author($node, $author) {
    $node->uid = $author->uid;
    $node->name = $author->name;
    return array('node' => $node);
}
```

Example: Check a content type

- Condition provided by rules
- Condition is configurable → the user selects the content types to check for.

```

<?php
/**
 * Implementation of hook_rules_condition_info()
 */
function node_rules_condition_info() {
    $items = array();
    $items['rules_condition_content_is_type'] = array(
        'label' => t('Content has type'),
        'arguments' => array(
            'node' => array('type' => 'node', 'label' => t('Content')),
        ),
        'module' => 'Node',
        'help' => t('Evaluates to TRUE, if the given content has one of the selected con
    );
    return $items;
}
/**
 * Condition: Check for content types - Configuration form
 */
function rules_condition_content_is_type_form($settings, &$form) {
    $form['settings']['type'] = array(
        '#type' => 'select',
        '#title' => t('Content types'),
        '#options' => node_get_types('names'),
        '#multiple' => TRUE,
        '#default_value' => isset($settings['type']) ? $settings['type'] : array(),
        '#required' => TRUE,
    );
}
/**
 * Condition: Check for selected content types
 */
function rules_condition_content_is_type(&$node, $settings) {
    return in_array($node->type, $settings['type']);
}

```

Developing new stuff with rules

- helps optimizing code reuse!
 - added events, conditions and actions can be used wherever possible!
- Eases and speeds up development!
 - When writing conditions, actions you don't have to care where to get your variables from.
Just specify what you need and use it!
- Minimizes the need of custom code!
- Helps upgrading to future versions:
 - Just upgrade custom rules integration and let your rules be upgraded automatically!

How does that all work?

- Rules internally keeps a list of all available variables and their types.
- Events expose available variables and actions may load further variables (Load a referenced node...)
- Conditions, actions and input evaluators just work with the list of available variables.
- So token replacements are just added for all variables provided by rules – there is no need to support a certain context!
- Variables can be loaded dynamically, so they are just loaded when something really needs them!

Comparison to core actions

- 2 different kind of actions:
 - There are rules style actions and core style actions.
 - Rules makes use of the core style actions and provides the rules style actions for some cases.
- Why another kind of action?
 - to be able to work variable-based.
 - To make new features possible like
 - modular input evaluators
 - exposing new variables to rules
- Rules actions work the same way as rules conditions

Current state of the module

- The module is beta state since August 08.
- Since that the API is stable and modules already integrate with rules (CCK, Content Profile, ...)
- It's already stable, so what's missing?
 - Polish documentation
 - Implement rule categories (tag your rules!)
 - Convert simple tests to 6.x-2.x and improve them!
 - Add views integrations to list scheduled rule sets
- The 1.0 release is expected to appear in February.

Modules supporting rules (1)

- CCK
 - Populate a field
 - Check a field's value
 - Check whether a field has changed, ...
- OG
 - React when a user joins/leaves a group
 - Subscribe / un-subscribe user to a group
 - Modify group settings of a certain group
- Token
 - Token replacements everywhere!

Modules supporting rules (2)

- Content Access
 - Set or check role based content permissions
 - With acl, set content access per user
- E-Commerce
 - extend the workflow of transaction with rules
- Content Profile
 - Check whether a user has a content profile
 - Load the content profile of a user
- Friendlist
 - A lot of events, conditions, actions: all you need!

What's next?

- Enhance rules 1.x
 - Add support for “Lists of data” and looping.
E.g. loop over multiple referenced nodes or over nodes determined by a view.
 - Help with the flag module rules integration
 - Add a rules based logging solution
 - React upon any event and log!
 - Useful for content logs or an activity stream.

What's next? Rules 2.0!

- Probably I'll work on enhancing rules for my master thesis, starting with March.
- Build rules 2.0 (d6 compatible)
 - An improved version building upon the same API (totally compatible to the 1.x API!)
- Work on
 - A human readable rules export for documentation purposes.
- Research semantic web integration possibilities (support formats like RIF, R2ML, RuleML, ..)

What's next? Rules 2.0 (continued)

- Rule-based communication between web sites!
 - Add support for web services (services module?)
 - Let drupal sites talk to each other!
 - RDF!? Other ideas?!
- Work out solutions for those use cases:
 - Distributed Content Deployment
 - Sharing a taxonomy across multiple Drupal sites
 - A workflow based review process
 - Rule based automated publishing and expiration of content

How can you help?

- Your help is needed!
- Share your rules, write tutorials!
- Create bug reports, patches, ..
- Write module integration, feature requests
- Starting points:
 - g.d.o: <http://groups.drupal.org/rules>
 - Handbooks: <http://drupal.org/node/298476>