

# 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.
- Comparison to the trigger module and drupal actions
- Outlook

# 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, Node Profile, Node Family, ..

# Module overview – What is it?

---

- define conditionally executed actions based on occurring events
- 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

# Argument 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!

# Example

---

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



## Editing rule Notify the content author about changes from other users

### Rule settings

**Label:** \*

Notify the content author about changes from other users

Choose an appropriate label for this rule.

**Event:** \*

After updating existing content

Select the event on which you want to evaluate this rule.

This rule is active and should be evaluated when the associated event occurs.

**Weight:**

0


Adjust the weight to customize the ordering of rules.

Save changes

### Rule elements

#### Conditions

NOT  updating user is author 

 Add a condition

#### Actions

 Inform the author by mail

 Add an action

# 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.

# 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']);
}

```

# Summary: Site building with rules

---

- Quickly build new functionality by configuring some rules
- Import/Export helps staging your rules to production sites!
- If you want version control for your rules, just provide them as default rules.
- Easily execute custom short code snippets with rules!
- Write new functionality by exposing new conditions, actions, events, ... and optimize code reuse!  
(More on this later...)

# 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 argument-based.
  - To make new features possible like
    - modular input evaluators
    - Exposing new variables to rules
- Rules actions work the same way as rules conditions

# Write core or rules actions?

---

- If it's possible to write your action as core action too, do it!
- Rules will be able to execute your action too, as well as other modules relying on core actions!
- To benefit from better integration with rules features like modular input evaluation and fancy labels, you can add some rules related stuff on top of the core action!
- Look at the documentation at <http://drupal.org/node/299055>

# Rules vs Trigger module

---

- Rules provides more features  
(Conditionals, Logical operations, Rule Sets, Scheduling, Input Evaluators, Default rules, Import/Export, ..)
- Rules works argument based!
  - Expose a new event and be able to make use of all possible conditions and actions!
  - Expose a new action and use it with every possible event or rule set!
- Trigger module does not!
  - Expose a hook and use it with which actions?

# Action code reuse and contexts ?

---

- When you write an action the goal is to be able to
  - use the action wherever possible!
  - So it shouldn't rely on any certain context!
- Trigger module unifies the context for the core actions of type user, node and comment! But..
  - The list of supported action types is fixed.
- How to make use of the context? E.g. for providing replacements when sending mail?
  - Trigger: The action has to support all possible contexts.

# Optimizing code reuse!

---

- Rules internally keeps a list of all available variables and their types.
- Events expose available variables and actions my load further variables (Load a referenced node...)
- Conditions, Actions and input evaluators can easily work with them!
- So token replacements are just added for all available variables – there is no need to support a certain context!
- When conditions or actions need them, they can be loaded dynamically!

# 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 arguments from. Just specify you need them and use them!
- Minimizes the need of custom code!
- Helps upgrading to future versions:
  - Just upgrade custom rules integration and let your rules be upgraded automatically!



# Current state of the module

---

- The first beta has just been released!
- The API is stable and modules can start adding module integration!
- The 1.0 release is expected to be seen soon, until then:
  - polish documentation
  - improve the usability (categories)
  - add views integrations to list scheduled rule sets
  - fix occurring bugs.

# Outlook

---

- Work on the CCK integration (<http://drupal.org/node/299706>)
- Port workflow-ng extension modules
  - Logging module
  - States module → CCK ?
  - CCLinks → Flags?
- Future:
  - Add support for “Data sets” and looping through them. E.g. loop over multiple referenced nodes.
  - Integrate with WebServices, perhaps by integrating with the services module.