approval

Purpose

The approval macro defines a (one of optionally multiple) named approvals for page or a workflow. Once a page has met all approvals the page advances to the state named by the approved parameter of the surrounding state macro. The approval macro must be used inside a state macro.

Syntax

{approval} Syntax

```
{state:state_name|approved=state_name1[|rejected=state_name2]}
    {approval:approval_name[|user/group/selectedapprover/selectedapprovers][|minimum=#][|weight=%]
[|label=label_name][|assignable=true/false] [|rememberassignees=true/false][|credentials=0/1/2]}
{state}
```

Parameters

Parameters	Mandatory	Default	Description
unnamed first parameter	Yes		Name of the approval. Names must be unique withing a state macro and can use any character set supported by Confluence expect the special characters used by the workflow syntax and Confluence (e.g. {. |}, ,|, etc.).
`user`[1]	No		A comma-separated list of specific user names or Value References.[2]
`group`[1]	No		A comma-separated list of specific group names or Value References.[2]
`exclude`[3]	No		A comma-separated list of specific user names or Value References who cannot approve or reject the approval.
`selectedapprover`[1]	No		A comma-separated list of specific user names, group names, or Value References to choose from, when a *single* approver is to be selected.
`selectedapprovers`[1]	No		A comma-separated list of specific user names, group names, or Value References to choose from, when one or more approvers are to be selected.
`weight`	No	40	A numeric value used to determine the order in which multiple approvals within a given state appear. The *lightest* weight approvals are shown first, and the *heaviest* last. Valid values range from 1 to 2^32^.
`label`	No		Sets the label which must be present on the page or news/blog post to make the approval active.
`assignable`	No	`false`	If set to `true`, the user could assign the approval to another user or users.
`rememberassignees`	No	`false`	If set to `true` any subsequent times the workflow transitions to a state with this approval, it will automatically be assigned to the same users who were assigned last time.
`minimum`	No	1	A numeric value or Value Reference that sets the minimum number of users that have to approve the approval.
`credentials`	No	0	If set to 1, then approvers have to provide their password; if set to 2, approvers need to provide both their user name and password. See Credentials below.
`approvelabel`	No	`Approve`	Sets the text that will be displayed in the approval "approve" action button.
`rejectlabel`	No	`Reject`	Sets the text that will be displayed in the approval "reject" action button.
Conditions	No		Sets the conditions that have to be met in order to enable this approval. If no conditions are set, then any user who has permission to see draft versions can approve or reject.

Table Notes

1. These parameters are mutually exclusive. If none of these parameters are set, then any user authorized to edit the page can give the approval.

2. If the list is preceded by a '&' then all the users must give the approval. This mode is incompatible with the *assignable* param.

3. Excluded users from an approval cannot approve/reject or be assigned to that approval. However, they still can assign other users to it (if *assignable* param is true).

Approver Selection

There are several ways to allow users to assign approvers:

- To require users be assigned before they can approve the approval, use either the `selectedapprover` or `selectedapprovers` parameter. The only difference is that `selectedapprover` only allows 1 user to be assigned and `selectedapprovers` allows 1 or more users to be assigned.
- To allow users to approve the approval without being assigned but still allow users to be assigned to the approval, use the `assignable=true` parameter. To limit who can approve the approval or be assigned to the approval use either the `user` or `group` parameter. If neither the `user` nor `group` parameters are used, any user will be able to approve the approval or be assigned to the approval.

Reset or Ignore on Page Updates

When an approval requires multiple users to approve, or there is more than one approval in a state they can be configured to reset on page updates or keep their status. This is a global (or space) configuration option, not a parameter on the approval macro.

Examples

General Example

An approval must be named, even if there is only one approval in a state. In this case we recommend naming the approval "*State_Approved", e.g. for _Sign Off* state "Sign Off Approved". Where multiple approvals are prerequisites for overall approval in a state (but no sequence is first necessary) then name the approvals accordingly, e.g. for multiple Editorial qualities of a piece of text in a review state you might have "clearApproval", "conciseApproval" and "compellingApproval".

```
{workflow}
   {state:Draft|approved=Editorial Review}
        {approval:Draft Approved}
    {state}
    {state:Editorial Review|approved=Sign Off|rejected=Draft}
        {approval:clear|group=reviewers}
        {approval:concise|group=reviewers}
        {approval:compelling|group=reviewers}
    {state}
    {state:Sign Off|approved=Published|rejected=Editorial Review}
        {approval:Sign Off Approved|user=@Editor in Chief@}
    {state}
    {state:Published|final=true}
    {state}
    {trigger:pageupdated|state=Published}
        {set-state:Draft}
    {trigger}
{workflow}
```

Approvals Conditional on an Earlier Approval

In this example, the page can get the *Reviewer* approval/rejection (by users belonging to the group `reviewers`) only if it has already received the *Author* approval.

```
{approval:Reviewer|hasapproval=Author|group=reviewers}
```

In the example below, the *Author* approval has to be given first before the *Reviewer* approval. In the example, the *Reviewer* approval will be visible but not active until the *Author* approval has been given.

```
{state}
{approval:Author}
{approval:Reviewer|hasapproval=Author|group=reviewers}
{state}
```

In the next example, the *Author* approval can be given only by members of the `content-providers` user group. The *Peer* approval can be given by either of users `emendator` or `primus`, with the user giving the *Author* approval making the selection. The *Reviewers* approval must be given by all the members of the `reviewers` user group (as indicated by the proceeding ampersand), after the *Peer* review approval is given.

```
{state}
{approval:Author|group=content-providers}
{approval:Peer|selectedapprover=emendator,primus|hasapproval=Author}
{approval:Reviewers|user=&reviewers}
{state}
```

Approval Requiring User Credentials

To create an approval which requires users to provide their user name and password:

```
{approval:Review|credentials=2}
```

Security Warning

The user name and password are sent to the server in clear text (the same way it is done when logging in to Confluence); therefore, it is recommended only in a trusted environment and/or using HTTPS.

Logging Invalid Approval Authentication

If you need to log when an approval, rejection or invalid attempt for an approval with a credentials param is made you can enable INFO level logging on the `com.comalatech.confluence.workflow.DefaultApprovalManager` class. This will produce logging lines similar to

```
[INFO] [talledLocalContainer] 2013-05-15 14:51:31,035 INFO [http-1990-6] [comalatech.confluence.
workflow.DefaultApprovalManager] approvePage user: admin pageId: 983156 approval: Review approved: true
error: Invalid credentials
```

which can be directed to another file depending on your logging configuration.