Grant approval

Description

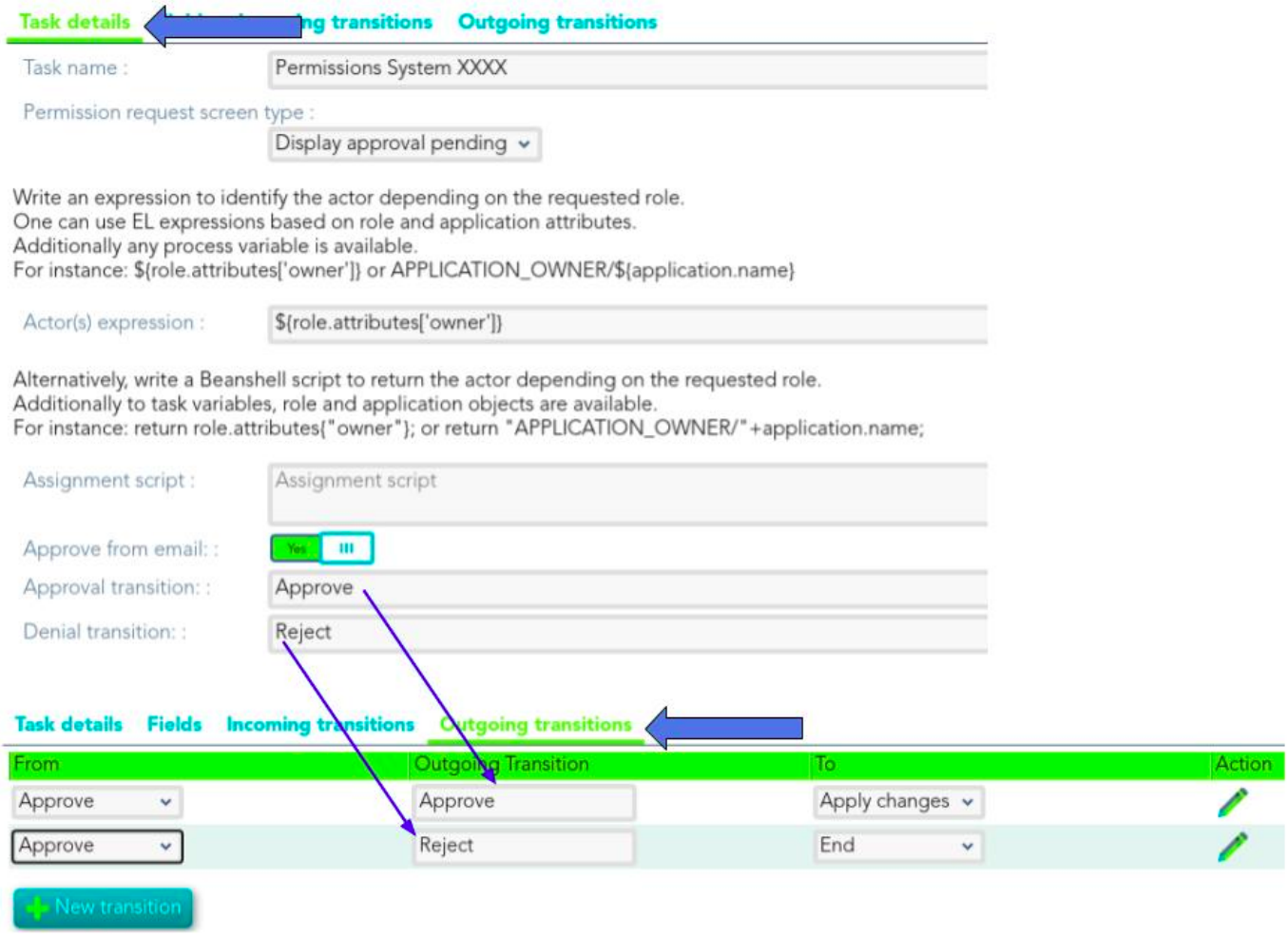
This step is used to define the custom form that will be used by the users who have to approve or reject the generated task. To configure that step will be necessary to determine the fields that will be shown to the users, and the actions that these users could perform.

Steps Tabs

Task details

- **Task name**: identified name for the task that will be created.
- **Permission request screen type**: allows selecting the type of screen for permission request.
  - List of permissions
  - **Display approval pending**: that is the default option. When you select that option, all the approval pending will be shown to the end user.
  - Display all
  - Display approved
  - Display denied
- **Actor(s) expression**: write an expression to identify the actor depending on the requested role. One can use EL expressions based on role and application attributes. For instance: SOFFID_MANAGER/${primaryGroup}
- **Assignment script**: alternatively, write a Beanshell script to return the actor depending on the process variables. For instance: return primaryGroup.attributes{"owner"};
- **Approve from email**: checked it to allow you to send a mail to approve or deny the task. If you check that option (selected value Yes), you need to fill in the transitions to approve and deny the task, those have to match with the outgoing transitions defined for those step.
  - Approval transition: has to match with an outgoing transition.
  - Denial transition: has to match with an outgoing transition.

To send mail, you will need to configure mail server parameters. You can visit the Soffid parameters page for more information.



Example Assignment script

If a user belongs to the primary group "World", the manager of that group will be responsible to approve or deny the request. If the primary group is another, the persona who will be responsible to approve or deny will be the manager of the parent group of that group. If there is not primary group, the request will be sent to the admin user.

```
primaryGroup = executionContext.getVariable("primaryGroup");
if (primaryGroup != null && !primaryGroup.equals("")) {
  if (primaryGroup.equals("world")) {
     manager =
serviceLocator.getGroupService().findGroupByGroupName(primaryGroup).getAttributes().get("manager");
     return manager;
} else {
     group = serviceLocator.getGroupService().findGroupByGroupName(primaryGroup);
  if ( group.parentGroup != null && !group.parentGroup.equals("")) {
       manager =
serviceLocator.getGroupService().findGroupByGroupName(group.parentGroup).getAttributes().get("manager");
```

```
    return manager;
      }
    }
} else {
  return "admin";
}
```

Fields

In this tab, you could choose what fields the process form will show to the end users. You can choose these fields from all identity attributes, and from the attributes defined for the workflow on the Attributes Tab. By default, all the identity attributes will be shown. You can choose the fields you want to show, add new fields, and delete the fields that do not need to generate a task. Also, you can sort the fields, you only need to drag and drop on the Order column.

For each field, you may indicate if it is a readOnly field, and you may add a Validation script and Visibility script. The validation script allows you to define rules, the field has to comply with these rules. The visibility script allows you to define the rules to show or hide a field.

Example

```
if (value == null || value.equals(""))
  return ("The user is mandatory");
else
  return true;
```

Incoming transitions

The Incoming transitions tab displays the previous steps where the flow comes from. When you create a process from a template or from scratch default incoming transitions are defined. It is allowed to customize the default setup, add new transitions, or delete transitions.

- **From**: the previous step, where the flow comes. Allows you to select where the workflow comes from.
- **Incoming transition**: brief name to identify the transition. That is the name of the action the form will show to the final user.
- **To**: current step.
- **Action**: allows creating a custom script to perform specific actions.

When you create an incoming transition, Soffid creates the proper outcoming transition.

Example

Validation of mandatory fields:

```
a = executionContext.getVariable("firstName");
if (a==null || "".equals(a.trim()))
  throw new Exception("First name is mandatory");

a = executionContext.getVariable("lastName");
if (a==null || "".equals(a.trim()))
  throw new Exception("Last name is mandatory");

a = executionContext.getVariable("primaryGroup");
if (a==null || "".equals(a.trim()))
  throw new Exception("Primery group is mandatory");

return true;
```

To request the process is only allowed for Internal users:

```
userSelector = executionContext.getVariable("userSelector");
user = serviceLocator.getUserService().findUserByUserName(userSelector);
if (user.userType.equals("I") || user.userType.equals("S")) {
throw new Exception ("To request the process is only allowed for Internal users");
}
```

Outgoing transitions

The Outcoming transition tab displays the next steps where the flow can go from the current step. When you create a process from a template or from scratch default outcoming transitions are defined. It is allowed to customize the default setup, add new transitions, or delete transitions.

- **From**: current step.
- **Incoming transition**: name of the transition.
- **To**: the next step, where the flow goes.
- **Action**: allows creating a custom script to perform specific actions.

When you create an outcoming transition, Soffid creates the proper incoming transition.

Example

To scroll through the list of values to perform some operations.

```
grants = executionContext.getVariable("grants");
for (roleRequestInfo:grants) {
  // TO-DO
}
```

Revision #20
Created 4 June 2021 06:44:26
Updated 24 November 2022 12:09:34