Start

Definition

That is the first step of the workflow. At that step, you could define the fields you want to show when the end users will go to make a request.

Steps Tabs

Task details

This process type does not have task details for the start step.

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, and an additional field called **Action**. You can choose the fields you want to show when the end-users, 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.

The Action field is a droplist that will allow end-users to select one of the different options to perform. The available actions, defined by default on the Attributes tab:

- **Add user**: action uses to generate a task to create a new identity.
- **Enable user**: action uses to create a task to enable an identity who is disabled.
- **Modify user**: action uses to create a task to modify identity attributes.
- **Disable user**: action uses to create a task to disable identity.

To enable, modify or disable an identity, you need to add a field with the name **userSelector**, defined on the Attributes tab. That field will be available, to end-users, to select an existing identity when selecting one of that options. When you select an identity, Soffid will show all the

attributes defined on the form to the end user.

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.

Validation examples

```
if (value == null || value.equals(""))
  throw new Exception("The userName is mandatory");
else
  return true;
```

It is also allowed in the following manner:

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

Validate that a certain field is not repeated:

```
userList = serviceLocator.getUserService().findUserByJsonQuery("attributes.field_XX eq \"" + value +"\"");
if (!userList.isEmpty() {
  return "the field field_XX is associated to another user";
}
return true;
```

Visibility example

```
user = serviceLocator.getUserService().getCurrentUser();
if ("admin".equals(user.userName))
  return false;
```

SCIM filter example

```
userType eq "E"
```

Triggers

On the trigger tab, you could define different triggers using custom scripts. Those triggers will be launched with the events you will define.

- **onLoad**: you can use that trigger to perform some actions before the execution of the step.
- **on PrepareTransition**: you can use that trigger to perform some actions after the execution of the step and before starting a transition to another step.
- **onChange**: you can use that trigger to perform some actions when the value of the attribute is changed. You could choose the field from a list.

Example

1. Calculate the email when firstName or lastName changes and depending on the userType:

```
firstName   = (inputFields.get("firstName")!=null) ? inputFields.get("firstName").value : null;
lastName    = (inputFields.get("lastName")!=null) ? inputFields.get("lastName").value : null;
userType    = (inputFields.get("userType")!=null) ? inputFields.get("userType").value : null;

if (firstName!=null && !firstName.trim().isEmpty() &&
    lastName!=null && !lastName.trim().isEmpty() &&
    userType!=null && !userType.trim().isEmpty()) {

  emailAddress = firstName + "." + lastName;
   if ("E".equals(userType)) {
   emailAddress = emailAddress + ".ext@soffid.com";
   } else {
     emailAddress = emailAddress + "@soffid.com";
   }
  inputFields.get("emailAddress").value = emailAddress;

}
```

> You can find more information about StandardUserWindow.java on Github.

2. Load the user data into the form.

```
user = serviceLocator.getUserService().getCurrentUser();
task.getVariables().put("action", "M");
task.getVariables().put("userSelector", user.userName);
workflowWindow.fetchUserAttributes()
```

Incoming transitions

This process type does not have task details for the start step.

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

Check if there are any similar identities:

```
firstName = executionContext.getVariable("firstName");
birthDate = executionContext.getVariable("birthDate");

df = new java.text.SimpleDateFormat("yyyy-MM-dd");
query = "firstName co \""+firstName+"\" and attributes.birthDate sw \""+df.format(birthDate)+"\"";

users = serviceLocator.getUserService().findUserByJsonQuery(query);
if ( !users.isEmpty()) {
  throw new es.caib.bpm.toolkit.exception.UserWorkflowException("Your identity is probably registered. Please,
contact your system administrator");
}
```

Revision #40
Created 4 June 2021 06:43:33
Updated 30 October 2023 09:58:27