

# **Bonita Camp**

## **Exercises**

---

# Bonita Camp

Copyright © 2019 Bonitasoft

---

---

# Preface

The goal of the exercises is to create a basic leave request process.

The process will execute in the following way:

1. An employee requests a leave
2. The employee's manager validates the request

In order to implement this process we are going to follow an iterative approach mixing theoretical content and practical exercises. Each exercise will allow you to build an executable and more complete version of the process.

Each exercise is divided in 3 sections:

- **Goal** - Presents the objectives of the exercise.
- **Instructions overview** - Contains an overview of the main steps required for completing the exercise.
- **Step by step instructions** - Contains a detailed description of the steps required for completing the exercise.

---

# Environment setup

## 1. Overview

This chapter explain how to setup your development environment in order to be able to do the exercises described in the next chapters.

## 2. Install Java Development Kit (JDK) 8

Bonita Studio requires Java 8 to run. You can get the JDK from Oracle website [<http://www.oracle.com/technetwork/java/javase/downloads/index.html>].

Make sure that you install Java 8 32-bit (x86) or 64-bit (x64) according to your Operating System type. For Windows 10, in order to know if you are running a 32 or 64 bit version of Windows, you can go to Settings - System - About and refer to "System type" in the list of information.

If possible we recommend to uninstall any other version of Java to guaranty that the correct version is used. If not you can see the documentation to configure Bonita Studio to use Java 8 [<https://documentation.bonitasoft.com/bonita/7.8/bonita-bpm-studio-installation>].

## 3. Install Bonita Studio

Go to the download section [<https://www.bonitasoft.com/downloads>] of Bonitasoft website, click on "Customize your download" and click on the "Download" button next to "Bonita Studio zip package" to get the **zip** package of Bonita Studio.

Unzip the Studio to the folder of your choice. Preferably choose a target folder with a path that is short and that doesn't include any space such as `C:\BonitaStudioCommunity-x.y.z`.

Note for macOS users: the zip package does not support macOS. On the download page you need to select "Mac" in order to get the .dmg file.

## 4. Start Bonita Studio

Open the Studio folder and use the appropriate runtime to start it:

- Windows 64 bit: BonitaStudioCommunity64.exe
- Windows 32 bit: BonitaStudioCommunity.exe
- macOS: BonitaStudioCommunity.app
- Linux 64 bit: BonitaStudioCommunity64-linux
- Linux 32 bit: BonitaStudioCommunity-linux

## 5. Web browser for development

We recommend to use Chrome or Firefox for development. In order to test which web browser your computer is currently using click on the "Portal" button in Studio tool bar. It should open the Bonita Portal with your default web browser.

---

You can configure Bonita Studio to use an other browser when opening web user interfaces:

- In the Studio click on "Preferences" (gear) button.
- In the "Web" section click on "Browser".
- Click on "New..." button.
- Type a name and use the "Browse..." button to select the web browser runtime (e.g. C:\Program Files (x86)\Google\Application\chrome.exe for Chrome on Windows).
- Click on "OK" button.
- In the list of "External web browsers" select the one you want to use and click on "Apply" button.
- Close the preferences window.

---

# Chapter 1. Exercise: Modeling a basic process

## 1. Goal

The goal of this exercise is to create a first basic version the leave request process diagram.

At this stage the process is executable but has very limited value from the business point of view as it does not yet contain forms or data. We will extend it in the upcoming exercises.

Note that validation of diagrams in Bonita Studio needs to be triggered manually. If you just fix an error or a warning you might want to go to "Validation status" tab and click on the "Refresh" button or go to Studio menu "Diagram" and click on "Validate".

## 2. Instructions overview

In order to complete the exercise, create a "LeaveRequestDiagram" diagram in version 1.0.0 containing a "LeaveRequest" process in version 1.0.0.

The process pool should contain the following BPMN elements:

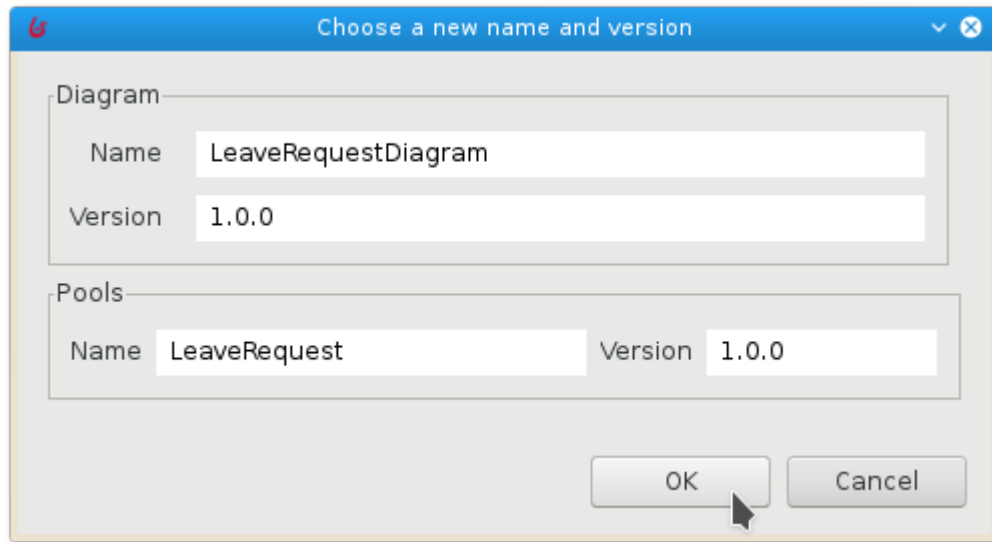
- A start event named "Fill request"
- A human task named "Validate request"
- An **exclusive** gateway named "Is Approved?"
- A service task named "Notify request approved"
- A condition always true (for testing purpose) on the transition connecting the gateway to the approve task
- A service task named "Notify request rejected"
- A default transition connecting the gateway to the reject task
- An end event named "End - Request rejected"
- An end event named "End - Request approved"

## 3. Step by step instructions

1. **Start the Bonita Studio.**
2. **Create a new process diagram.**
3. **Set the diagram and process names.**

Click on the "File / Rename diagram..." top menu.

Enter "LeaveRequestDiagram" as the diagram name, "LeaveRequest" as the pool name and set the version to 1.0.0 for both.



**4. Rename the start event into "Fill request".**

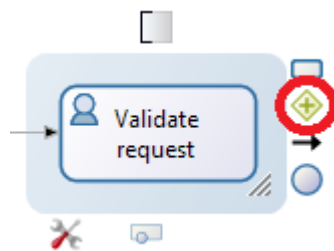
Select the start event from the diagram.

Navigate to the "General / General" tab and enter the new name.

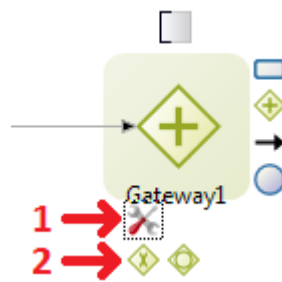
**5. Rename the human task into "Validate request".**

**6. Add an exclusive gateway named "Is approved?" next to the human task "Validate request".**

Select the "Validate request" task, perform a drag and drop from the gateway contextual icon.



Once you have created the gateway, use the context "tool" icon (1) to change its type to "Exclusive" (2).

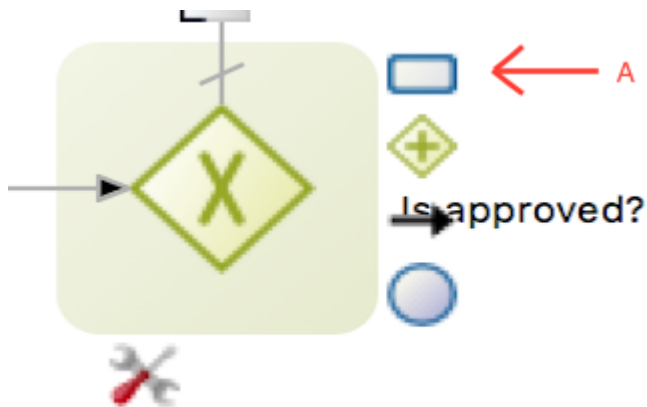


Rename the gateway into "Is approved?".

---

7. **Create a service task named "Notify request approved" after the gateway.**

Select the gateway, perform drag and drop from the contextual event icon (A).

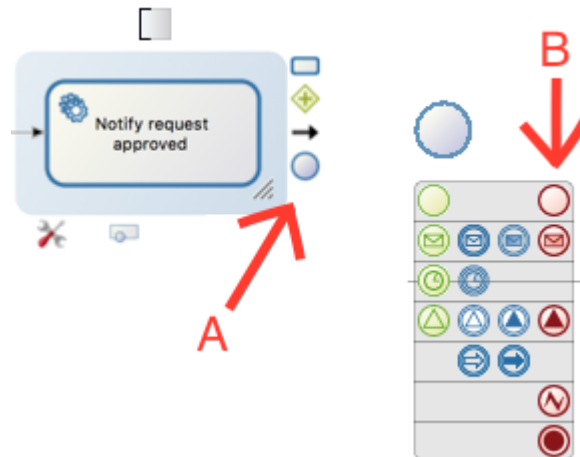


Once the service task is created, rename it.

8. **Create a service task named "Notify request rejected" after the gateway.**

9. **Add an end event named "End - Request approved" next to the newly added service task "Notify request approved"**

Select the task "Notify request approved", perform drag and drop from the contextual event icon (A) and select the end event (B).



Once the event is created, rename it.

10 **Add an end event named "End - Request rejected" next to the service task "Notify request rejected"**

11 **Name the two transitions going out of the gateway**

Select the transition.

Navigate to the "General / General" tab and enter the name.

The transition leading to the "Notify request approved" task should be named "Yes" and the other "No".







---

**12 Configure condition on the "Yes" transition so the flow always goes this path.**

Select the transition and type "true" in the condition field ("General / General" tab). If you don't see the condition field it is probably because you defined a parallel gateway instead of an exclusive one.

Condition ☒ Use expression ☐ Use decision table

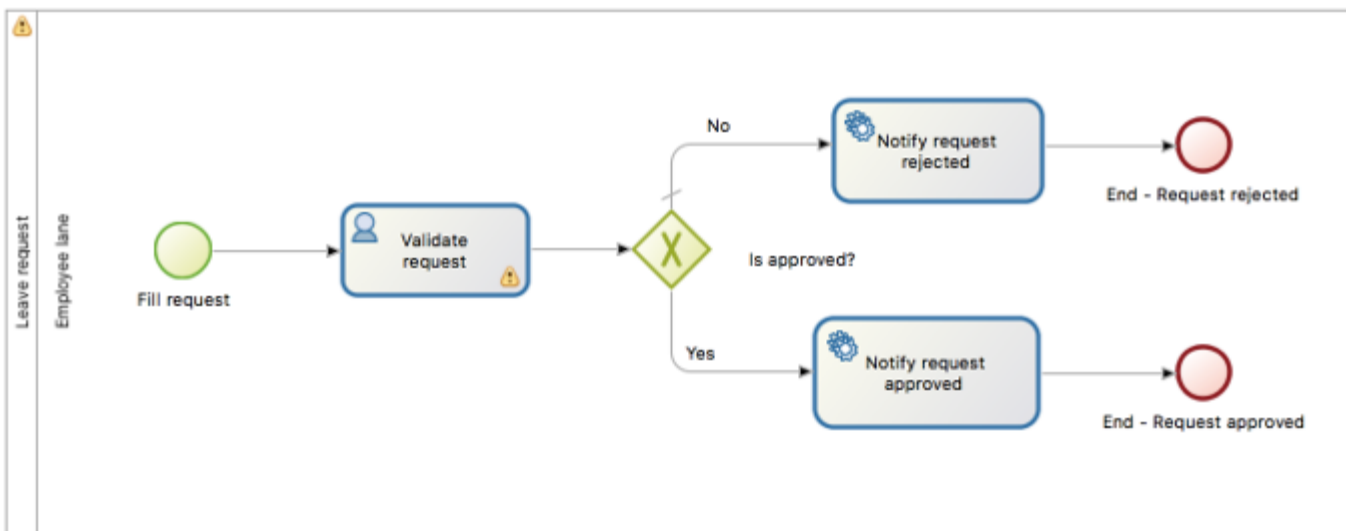
**13 Set the "No" transition as the default flow.**

Select the "No" transition and check the "Default flow" property.

**14 Validate the diagram.**

Once the transition conditions are properly set, you can re-validate the process diagram. To do so, navigate to the "Validation status" tab and click on "Refresh". You may safely ignore the 3 warnings related to the UI Designer.

**15 Make sure that your diagram matches this:**



**16 Save the process.**

Click on the "Save" button from the Studio's top menu bar.

**17 Run the process to test it.**

Select the process pool (the rectangle shape that includes tasks and events).



Click on the  button of the Studio's top menu bar (if button is disabled make sure the process pool is actually selected).

Your browser should open and display an automatically generated case start form. Submit the form.

You should be redirected to the task inbox of the Bonita Portal. Select the "Validate request" task and click on the "Take" button.

The screenshot displays the Bonitasoft web application interface. The top navigation bar is red with the Bonitasoft logo on the left and user information (Welcome: Walter Bates, User, Settings) on the right. Below the navigation bar, there are tabs for 'Tasks', 'Cases', and 'Processes'. The 'Tasks' tab is active.

On the left side, there is a sidebar with a 'To do' section (1 item) and 'My tasks' and 'Done tasks' sections. The main content area is divided into two panels. The left panel shows a 'Filters' section with a 'Process' dropdown set to 'All' and a search bar. Below this is a 'Task list' section with a table of tasks.

The 'Task list' table has the following columns: Task Id, Task name, Case Id, Process name, Last update, and Assigned on. There is a 'TAKE' button and a 'RELEASE' button above the table. The table contains one task:

Task Id	Task name	Case Id	Process name	Last update	Assigned on
4	Validate request	2	Leave request	Nov 14 2:27 PM	-

The right panel shows a task detail view for the 'Validate request' task. It has tabs for 'Form', 'Comments', and 'Overview'. The 'Form' tab is active. Below the tabs, there is a message: 'To fill out the form, you need to take this task. This means you will be the only one able to do it. To make it available to the team again, release it.' There is a 'TAKE' button next to this message. Below the message, there is a large heading 'Validate request' and a green 'Execute' button.

Submit the form of the "Validate request" task by clicking on the "Execute" button to complete your case.

---

# Chapter 2. Exercise: Adding data and specifying contracts

## 1. Goal

The goal of this exercise is to continue the leave request process implementation by:

1. setting up a Business Data Model (BDM)
2. specifying contracts for the case instantiation and human tasks

Once completed, the process will be executable with automatically generated forms.

### Warning

It is mandatory to strictly observe the case and syntax of the technical names provided in the instructions. Failing to do so will result in errors.

## 2. Instructions overview

Duplicate the process diagram from the previous exercise to create a 2.0.0 version.

Create a "LeaveRequest" BDM with the following attributes (don't use the "Multiple" option):

Name	Type	Mandatory
requestorId	Long	Yes
leaveStart	Date only	Yes
dayCount	Integer	Yes
isApproved	Boolean	No

Declare a business variable "request" of type "LeaveRequest" on your pool.

Using the Studio assistant (i.e.: "Add from data..."), generate the case instantiation contract, as well as the initialization script, from the "request" data with the following mandatory items:

- leaveStart
- dayCount

Add these two constraints on the case instantiation contract:

- "leaveStart" must be in the future
- "dayCount" must be strictly greater than zero

Initialize the "request" business variable.

Add a step contract on the "Validate request" task use the Studio assistant with the following item:

- isApproved

### 3. Step by step instructions

#### 1. Duplicate the process diagram from the previous exercise to create a 2.0.0 version.

In the Studio's top menu, click on "File / Duplicate diagram...".

Update the process diagram AND pool version numbers.

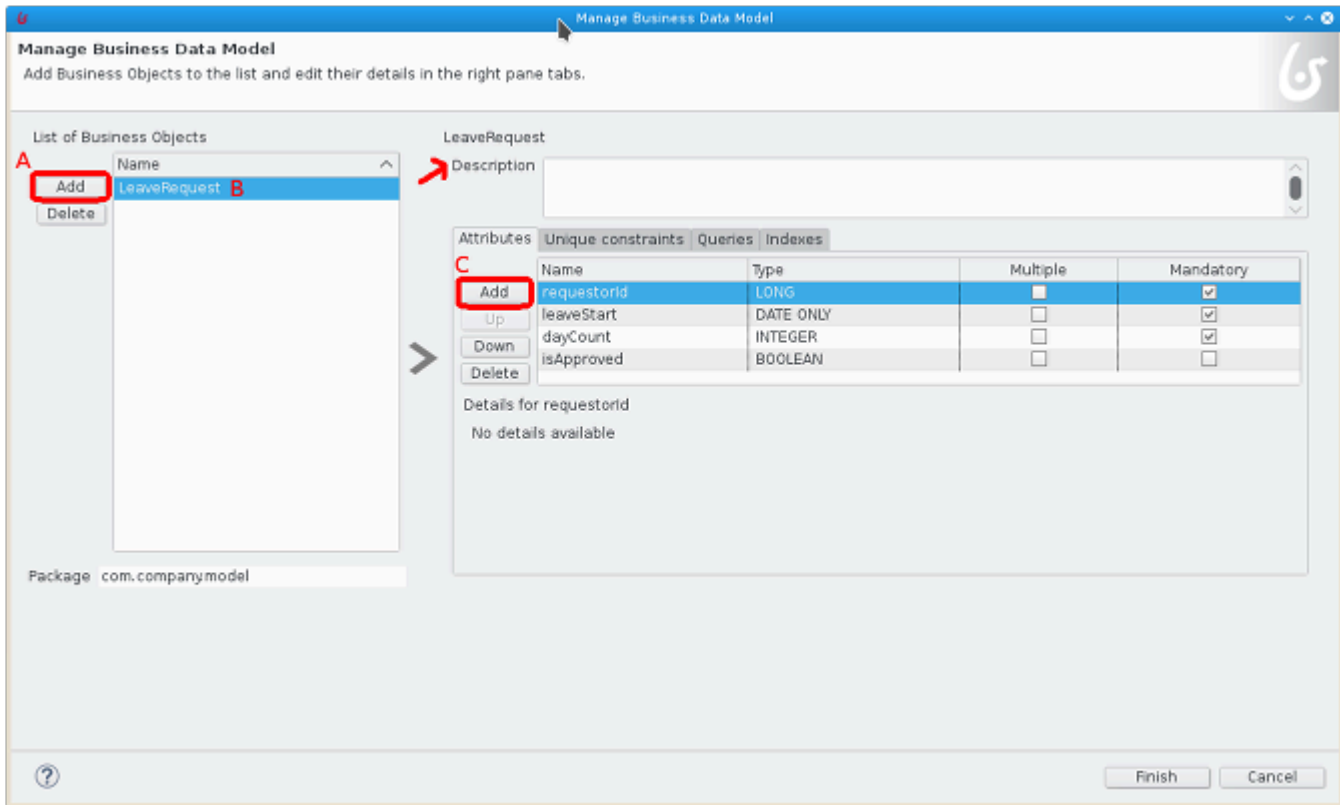
#### 2. Create the "LeaveRequest" BDM

Navigate to the "Development / Business Data Model / Define..." top menu.

Click on "Add" (A) in the "List of Business Objects" and name the object "LeaveRequest" (B) (this is a technical name so it should not contain spaces or special characters).

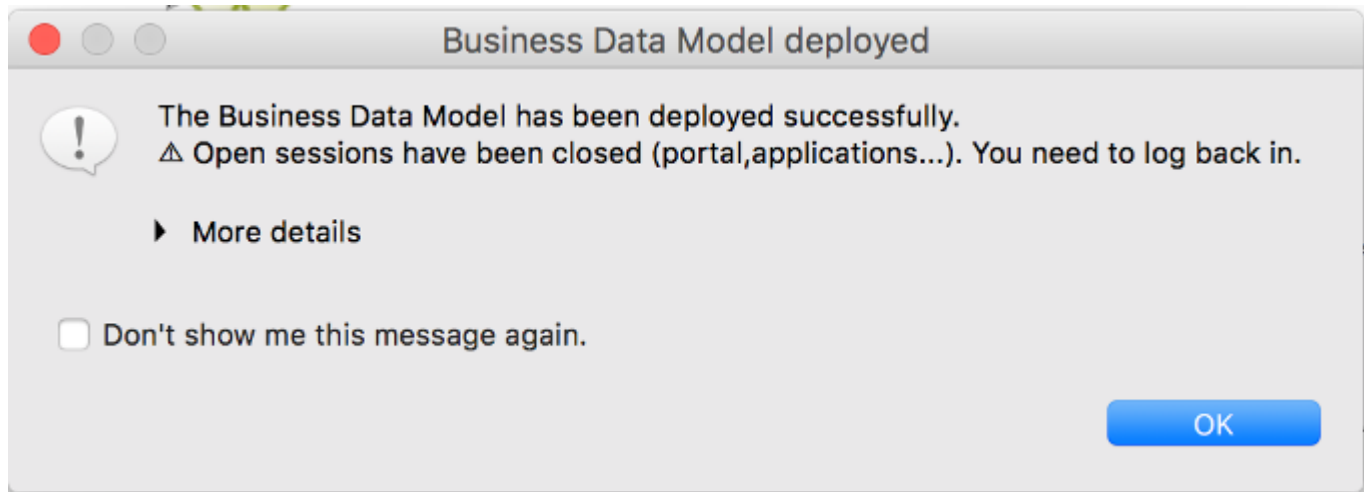
With the "LeaveRequest" object selected, add the following attributes (C):

Name	Type	Mandatory
requestorId	Long	Yes
leaveStart	Date only	Yes
dayCount	Integer	Yes
isApproved	Boolean	No



Click on "Finish".

You should be displayed with the following message that confirm the deployment of the BDM. Check the box "Don't show me this message again" and click on "Ok".



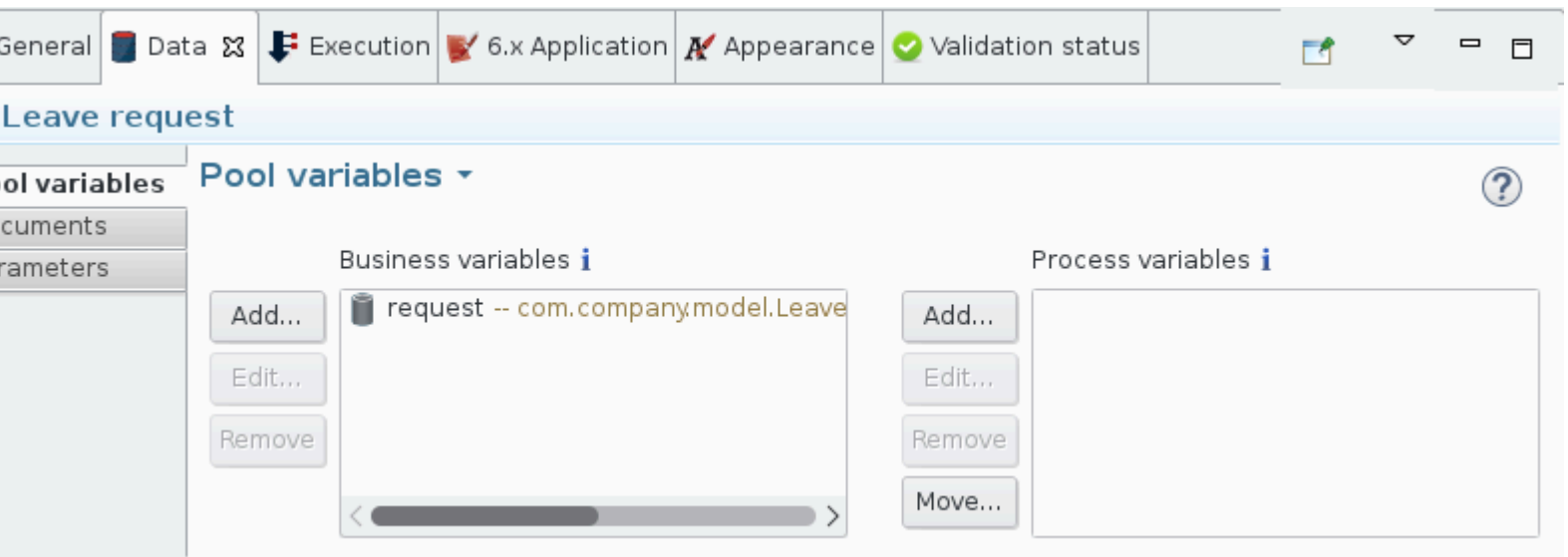
### 3. Declare an instance of the "LeaveRequest" BDM on your process

Select the process pool, and navigate to the "Data / Pool variables" tab.

In the "Business variables" section, click on "Add..."

Name the variable "request" and select the "LeaveRequest" business object.

Click on "Finish".



### 4. Set up the case instantiation contract

Select the process pool, and navigate to the "Execution / Contract / Inputs" tab.

Click on "Add from data..."

Select the "request" business data, keep the default input name "requestInput" and click on "Next".

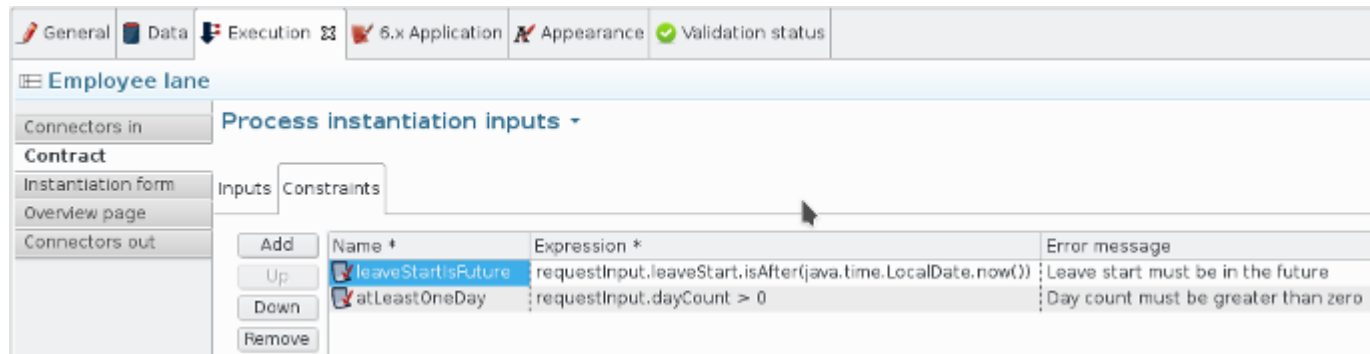
Uncheck the "requestorId" and "isApproved" attributes.



Property	Value
Name	leaveStartIsFuture
Expression	requestInput.leaveStart.isAfter(java.time.LocalDate.now())
Error message	Leave start must be in the future

Add a second constraint:

Property	Value
Name	atLeastOneDay
Expression	requestInput.dayCount > 0
Error message	Day count must be greater than zero



## 5. Update the "request" BDM initialization

Select the process pool, and navigate to the "Data / Pool variables" tab.

Select the "request" business variable and click on "Edit...".

Click on the "Pencil" icon next to the "Default value" field to open the expression editor.

Clear all of the generated code and replace it by the following:

```
def leaveRequestVar = new com.company.model.LeaveRequest()
leaveRequestVar.leaveStart = requestInput.leaveStart
leaveRequestVar.dayCount = requestInput.dayCount

// Retrieve current process instance
def processInstance = apiAccessor.processAPI.getProcessInstance(processInstanceId)
// Add requestor id to the new request
leaveRequestVar.requestorId = processInstance.startedBy

return leaveRequestVar
```

This will initialize the BDM from the contract data and set the process initiator as the request author.

Click on "OK" button to close the expression editor.

Click again on "OK" button to confirm the modification of the business data.

## 6. Set up the "Validate request" step contract

Select the "Validate request" task and navigate to the "Execution / Contract / Inputs" tab.

Click on "Add from data..."

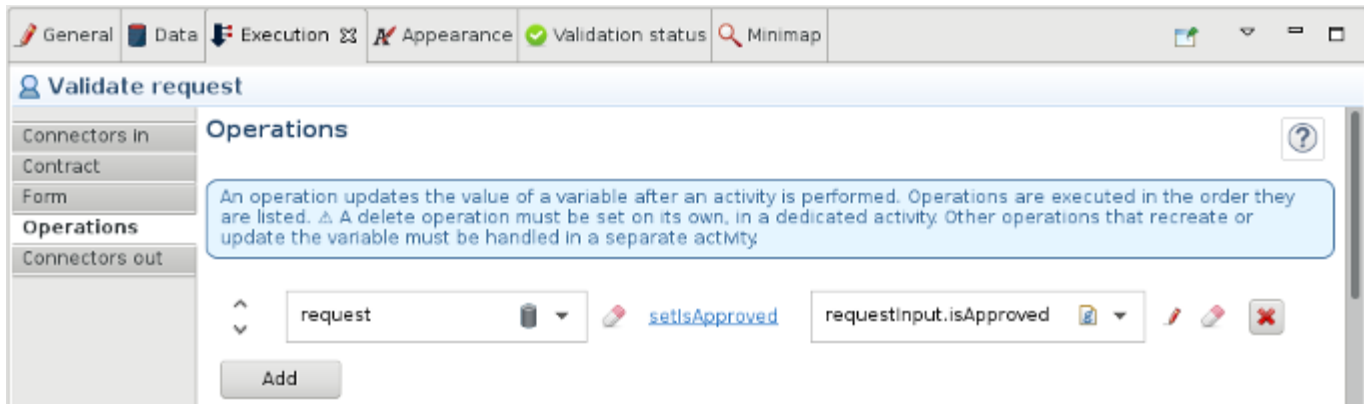
Select the "request" business data, keep the default input name "requestInput" and click on "Next".

Check only the "isApproved" attribute.

Click on "Finish" (not on "Finish & Add") and dismiss the warning message about the incomplete initialization of the business variable.

**7. Note that an operation on "Validate request" to update the request has been automatically generated:**

With the "Validate request" task selected, navigate to the "Execution / Operations" tab.

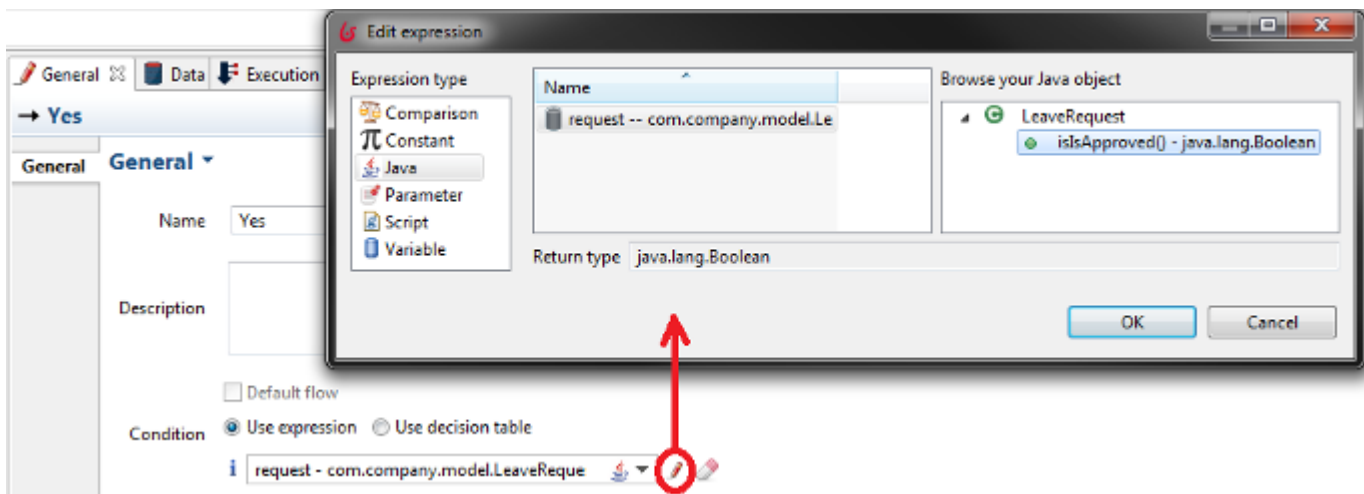


**8. Use the business data to dynamically control the process flow**

Select the transition "Yes" and navigate to the "General / General" tab.

Click on the pencil icon next to the "Condition" drop-down to open the expression editor.

In the expression editor, select "Java" as the expression type, select the "request" variable and the "isApproved" method.



**9. Save the process.**

Click on the "Save" button from the Studio's top menu bar.



---

## 10 Run the process to test your contract constraints.

Run the process and use the automatically generated forms to test your constraints.

### Tips:

- for the "leaveStart" input, enter a date in the YYYY-MM-DD format.
- for the "isApproved" input, enter a boolean value: either "true" or "false".

**Attention** : this is a temporary form generated automatically for testing. Before you put your process into production, create and map the necessary forms.

# Leave request

**requestInput** - *<no description defined in contract for this input>*

**leaveStart** - *<no description defined in contract for this input>*

Expecting a LOCALDATE value.

Example: 2018-08-06

**dayCount** - *<no description defined in contract for this input>*

Expecting a INTEGER value.

Start

---

# Chapter 3. Exercise: Creating forms

## 1. Goal

The goal of this exercise is to provide user friendly forms for the execution of the process.

## 2. Instructions overview

Duplicate the process diagram from the previous exercise to create a 2.1.0 version.

Create the following forms:

- A case instantiation form at pool level that sets the "leaveStart" and "dayCount" data.
- A form for the "Validate request" task that displays the "requestor" user details, the "leaveStart" and "dayCount" data in read-only mode and that sets the "isApproved" data.

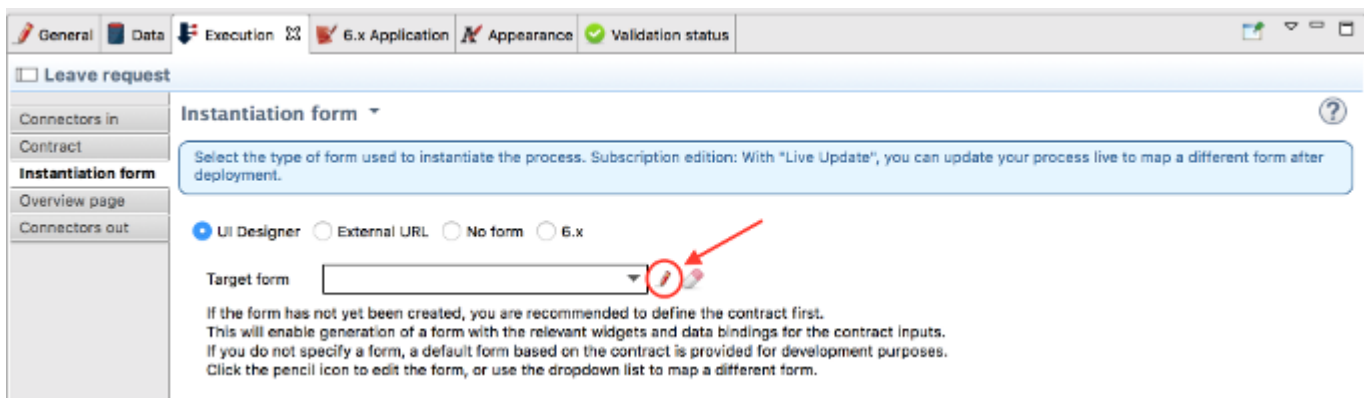
## 3. Step by step instructions

1. **Duplicate the process diagram from the previous exercise to create a 2.1.0 version.**

2. **Create a case instantiation form.**

Select the process pool, and navigate to the "Execution / Instantiation form" tab.

Click on the pencil next to the Target form input.



This will open the UI Designer in a browser with a form automatically generated from your instantiation contract.

3. **Rename the form.**

Use the text field on top of the screen to rename the form into "fillLeaveRequest" (this is a technical name so it should not contain spaces or special characters).

Click on "Save".

4. **Update the form title**

---

Select the top most widget "Request Input".

In the section on the right edit the following properties:

Property	Value
Text	Fill a new leave request
Text level	Level 2
Alignment	Center

#### 5. Update the "Day Count" widget

Select the "Day count" widget and edit the following properties:

Property	Value
Label	Number of days
Placeholder	Number of leave days
Min value *	1

\* not to be confused with the "Value min length" property.

#### 6. Clear the initial form values (this will show the placeholders at runtime)

In the lower "Variables" section, click on the "Pencil" icon for the "formInput" variable.

Replace the JSON values with this:

```
{
  "requestInput" : {
    "leaveStart" : null,
    "dayCount" : null
  }
}
```

#### 7. Add a variable to store potential form submit errors

In the lower "Variables" section, click on "Create a new variable".

Set the variable name to "error", leave its type to "String" and its value empty then click on "Save".


#### 8. Update the "Submit" widget

Set the "Failed response value" property to "error" (this will store the submit errors in the variable "error" if any).

#### 9. Dynamically display the error

Drag a "Text" widget from the palette and drop it under the "Submit" button.

Set the "CSS classes" property to "text-danger".

Click on the "binding" icon  next to the "Hidden" property.

---

Enter the following content in the text field that just appeared (this display the widget only when there is an error):

```
!error
```

Replace the "Text" property with the following content:

```
<b>Error :</b> {{error | json}}
```

## 10Check the form's appearance

Make sure that the form looks like this in the editor:

The form is titled "Fill a new leave request". It contains two input fields and a submit button. The first input field is labeled "Leave Start \*" and has a value of "data:formInput.requestInput.leaveStart", a "Today" button, and a calendar icon. The second input field is labeled "Number of days \*" and has a value of "Number of leave days". Below the input fields is a blue "Submit" button. At the bottom of the form, there is an error message: "Error : {{error | json}}".

Click on the "Save" button.

Click on the "Preview" button.

The form should look like this:

---

# Fill a New Leave Request

## Leave Start \*

Enter a date (mm/dd/yyyy)	Today	
---------------------------	-------	---

## Number of days\*

Number of leave days
----------------------

Submit

Verify the following points:

- the widget presenting the error is not visible
- the "Submit" button is disabled by default (this is due to the validation provided by the form container)
- the "Submit" button is enabled when the form is valid

**Note:** the form cannot be submitted from the preview mode even if it is valid.

Close the preview window.

### 11 Create a form for the "Validate request" task.

In the Studio, select the "Validate request" task and navigate to the "Execution / Form" tab.

Click on the pencil next to the Target form input to open a new form in the UI Designer.

### 12 Rename the form

In the UI Designer, rename the form into "validateLeaveRequest" and save it.

### 13 Remove unnecessary variables

In the lower "variables" section of the screen, remove the following variables:

- formInput
- formOutput

### 14 Retrieve business data from the ongoing request

Click on the "Create a new variable" button and configure the variable with the following properties:

Property	Value
Name	request
Type	External API
API URL	{{context.request_ref.link}}

**Note:** here, we are calling the Bonita REST APIs to fetch our "request" business variable value. We are using the "context" variable that exposes links to the process instance business variables including our request in the form of "request\_ref". We can then retrieve the request thanks to the "link" attribute which provides the URL that needs to be called to retrieve the object.

#### 15 Retrieve the requestor's user data

Create a new variable with the following properties:

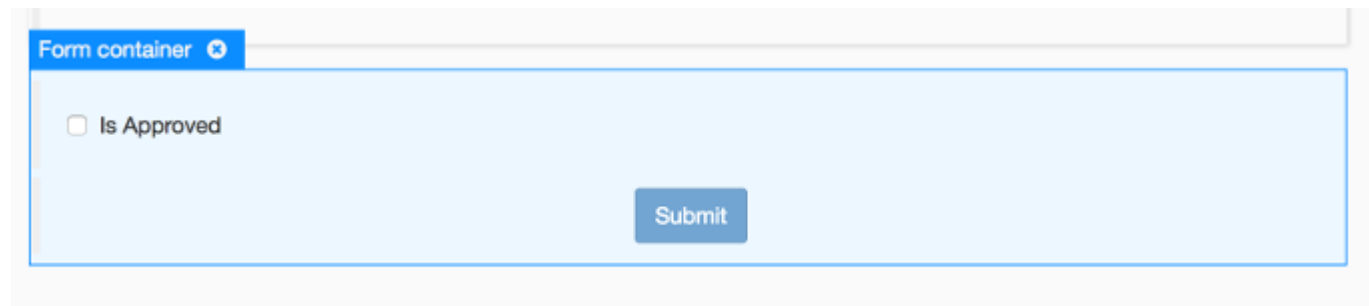
Property	Value
Name	requestor
Type	External API
API URL	API/identity/user/{{request.requestorId}}

#### 16 Add a variable for error handling

Create a new variable named "error", leave its type to "String" and its value empty then save it.

#### 17 Remove the "Is Approved" checkbox and the "Submit" button

Select the Form container that contains the checkbox and button widgets, then delete it by using the X button that appears when hovering over the container with the cursor.



#### 18 Modify the form title

Select the existing "Title" widget and configure it as following:

Property	Value
Text	Validate a leave request
Title level	Level 2

#### 19 Add a widget to display the requestor

Drag an "Text" widget from the palette and drop it on a new row under the form title.

Configure the widget as following:

Property	Value
Text	<b>Requestor:</b> {{requestor.firstname}} {{requestor.lastname}}

#### 20 Add a widget to display the leave start date

Drag an "Text" widget from the palette and drop it under the requestor widget.

Configure the widget as following:

Property	Value
Text	<b>Leave start date: </b>{{request.leaveStart   date : 'dd/MM/yyyy'}}

## 21Add a widget to display the number of days

Drag an "Text" widget from the palette and drop it on a new row under the "Leave start date" widget.

Configure the widget as following:

Property	Value
Text	<b>Number of days: </b> {{request.dayCount}}

## 22Add a form container

Drag an "Form container" widget from the palette and drop it on a new row under the "Number of days" widget.

## 23Add a widget to reject the request

Drag a "Button" widget from the palette and drop it in the form container (highlight with a dashed border).

Configure the widget as following:

Property	Value
Width	6
Label	Reject
Alignment	right
Style	danger
Data sent on click	{ "requestInput" : { "isApproved" : false } }
Failed response value	error
Target URL on success	/bonita

## 24Add a widget to approve the request

Drag a "Button" widget from the palette and drop it on the "6 column" zone located on the right of the "Reject" button.


Configure the widget as following:

Property	Value
Label	Approve
Style	success
Data sent on click	{ "requestInput" : { "isApproved" : true } }
Failed response value	error
Target URL on success	/bonita

## 25Dynamically display potential submit errors

Drag a "Text" widget from the palette and drop it on a new row at the bottom of the form.

Set the "CSS classes" property to "text-danger".

Click on the "binding" icon  next to the "Hidden" property.

---

Enter the following content in the text field that just appeared (this display the widget only when there is an error):

```
!error
```

Replace the "Text" property with the following content:

```
<b>Error :</b> {{error | json}}
```

## 26Check the form's appearance

Make sure that the form looks like this in the editor:

The screenshot shows a form titled "Validate a leave request" in a design editor. The form has a light gray background and a white border. It contains the following elements:

- A title "Validate a leave request" in a large, bold, dark gray font.
- A text field with the placeholder text "{{ task.displayDescription }}" in a dark gray font.
- A text field with the label "Requestor:" in bold and the placeholder text "{{requestor.firstname}} {{requestor.lastname}}".
- A text field with the label "Leave start date:" in bold and the placeholder text "{{request.leaveStart | date : 'dd/MM/yyyy'}}".
- A text field with the label "Number of days:" in bold and the placeholder text "{{request.dayCount}}".
- A dashed rectangular box containing two buttons: a red button labeled "Reject" and a green button labeled "Approve".
- A text field at the bottom with the label "Error :" in bold and the placeholder text "{{error | json}}".

## 27Save the form

Use the top "Save" button to save the form.

## 28Test the process execution

Run the process from the Studio and execute all of its steps.

Validate that the proper execution path was taken at the end of process execution by looking at the case history in the Bonita Portal.



---

# Chapter 4. Exercise: Configuring actors

## 1. Goal

The goal of this exercise is to add collaboration to the existing process by dispatching the forms between two actors: a requestor and a validator.

## 2. Instructions overview

Duplicate the process diagram from the previous exercise to create a 3.0.0 version.

Add a "Validator" lane to the diagram and move the "Validate request" task in it.

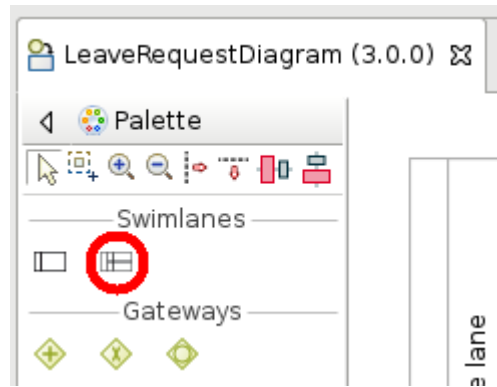
Add an actor filter of type "Initiator manager" on the "Validator" lane.

## 3. Step by step instructions

1. **Duplicate the process diagram from the previous exercise to create a 3.0.0 version.**

2. **Add a "Validator" lane to the process.**

Select the "Lane" element from the BPMN palette and click in the process pool.



With the lane selected, navigate to the "General / Lane" tab and set its name to "Validator".

3. **Rename the other lane into "Requestor".**

4. **Drag and drop the "Validate request" task into the "Validator" lane.**

5. **Add an "Initiator manager" actor filter on the "Validator" lane.**

Select the "Validator" lane and navigate to the "General / Actors" tab.

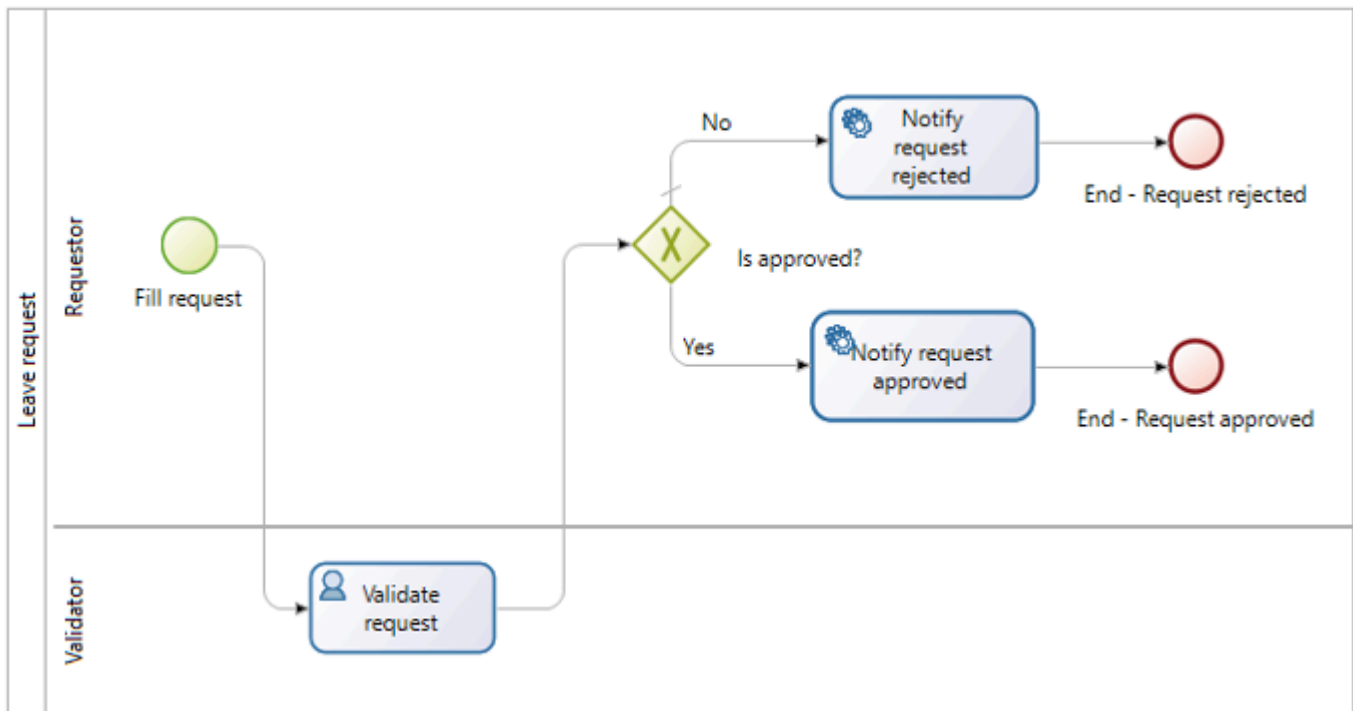
Select the "Employee actor" from the drop-down menu.

Click on the "Set..." button of the actor filter.

Select an "Initiator manager" filter and click on "Next".

Name the filter "requestorManager" and click on "Finish".

6. Make sure that the diagram looks like this:

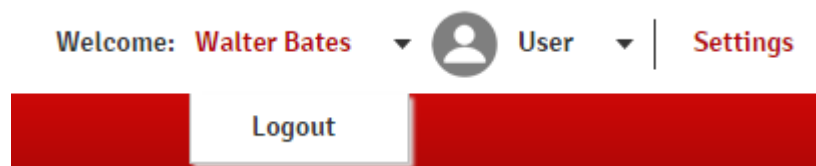


7. Execute the process with the two actors.

Run the process from the Studio (user "Walter Bates" will be used).

Submit the "Fill leave request" form. If the actors are properly set, the "Validate request" task should not be available anymore.

Disconnect from the Portal by clicking on the user name in upper right corner of the screen and clicking on "Logout".



Connect as user "helen.kelly" with "bpm" as password.

If the actor filter ran correctly, the "Validate request" tasks should now be available in the inbox.

---

# Chapter 5. Exercise: Using a connector to send an email

## 1. Goal

The goal of this exercise is to demonstrate an interaction between Bonita and an external system with the use of a connector. In this context, we will notify the leave request initiator about his request status with an email connector.

### Warning

Depending on your network configuration, your firewall or the security settings of your email server you may not be able to send an email from Bonita.

In order to get rid of those technical constraints, we will use a program that simulates an email server (FakeSMTP).

## 2. Instructions overview

Obtain and start the FakeSMTP server.

Duplicate the process diagram from the previous exercise to create a 3.1.0 version.

Add an email connector on the "Notify request approved" and "Notify request rejected" service tasks. These will send an email to the requestor with the request validation status.

The following code will be used to retrieve the requestor's email in the connector:

```
BonitaUsers.  
    getProcessInstanceInitiatorProfessionalContactInfo(apiAccessor,processInstanceId)  
    .email
```

## 3. Step by step instructions

### 1. FakeSMTP setup

Download FakeSMTP from this link: <http://nilhcem.github.com/FakeSMTP/downloads/fakeSMTP-latest.zip>

Unzip the file and run FakeSMTP by either double-clicking on the JAR file or running this shell command:

```
java -jar fakeSMTP-2.0.jar
```

Once the user interface is displayed, set the listening port to 2525 and click on the "Start server" button.

### 2. Duplicate the process diagram from the previous exercise to create a 3.1.0 version.

### 3. Test the email connector to obtain the right SMTP configuration.

Navigate to the "Development / Connectors / Test connector" top menu.

Select the "Email (SMTP)" connector by using either the text filter or the "Messaging" category and click on "Next".

Set these connection parameters (username and password remain empty):

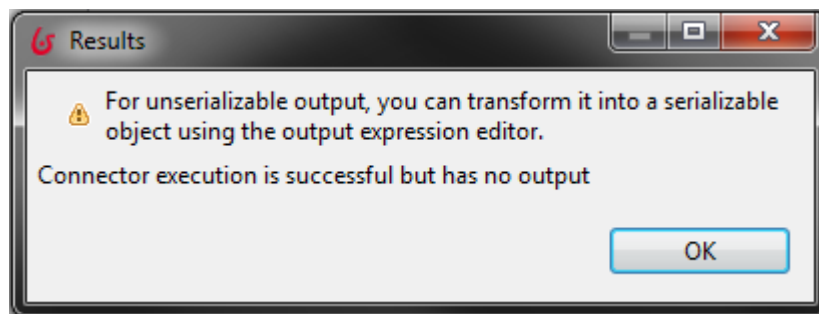
Property	Value
SMTP host	localhost
SMTP port	2525 (the port number specified in FakeSMTP)
SSL (in the "Security" section)	unchecked

Click on "Next".

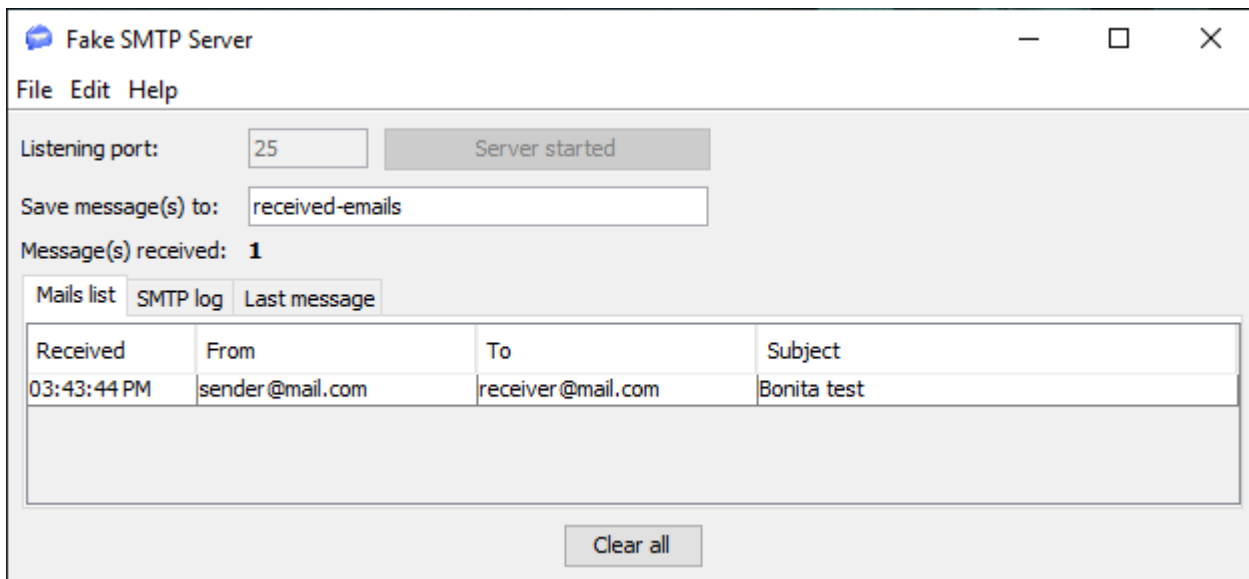
Enter some email addresses (not necessarily existing addresses) in the "From" and "To" fields and click on "Next".


Enter "Bonita test" as the subject, click on the "Test" button and confirm the test without checking any dependency.

At this stage, you should get a message similar to this one:



Make sure that the email is received by FakeSMTP as shown below:



Once the configuration is valid, click on .

Name the configuration "emailConfig" and save it.

Close the connector test interface.

---


#### 4. Add an email connector on the "Notify request approved" task.

Select the task, navigate to the "Execution / Connectors in" tab and click on "Add...".

Select the "Email (SMTP)" connector and click on "Next".

Name it "sendRequestApprovedEmail" and click on "Next".



Do not fill in the parameters but click on .

Select the "emailConfig" and move to the "Email addressee" configuration page.

Enter "hr@acme.com" in the "From" field.

Use the "pencil" icon to edit the expression of the "To" field.

Set the "Expression type" to "Script", name the script "getRequestorEmail" and paste the following in the code edition zone:

```
BonitaUsers.  
  getProcessInstanceInitiatorProfessionalContactInfo(apiAccessor,processInstanceId)  
    .email
```

Move to the next page and set "Leave request approved" as the subject.

Click on "Finish".

#### 5. Add an email connector on the "Notify request rejected" task.

Repeat the previous steps by naming the connector "sendRequestRejectedEmail" and setting "Leave request rejected" as the subject.

Alternatively you can use the feature that let you create a copy of an already configured connector and add it to another task.

#### 6. Test the process

Execute the process twice to test the different paths and check that FakeSMTP receives the right emails.

---

# Chapter 6. Exercise: Build a leave request application

## 1. Goal

The goal of this exercise is to build an application for users to create and manage their leave requests.

## 2. Instructions overview

Open the UI Designer and create a new "Application page" named "LeaveRequestStatus" to follow the progress of the connected user leave request.

This page will contain one "multiple container" that list the on-going leave requests. For each request, the start date, number of days and status will be displayed.

Export the page as a zip file and deploy it in the portal using the "Resources" menu of the "Administrator" view.

Create a new application. Then add the "LeaveRequestStatus" page in the application.

Access the application using the generated URL.

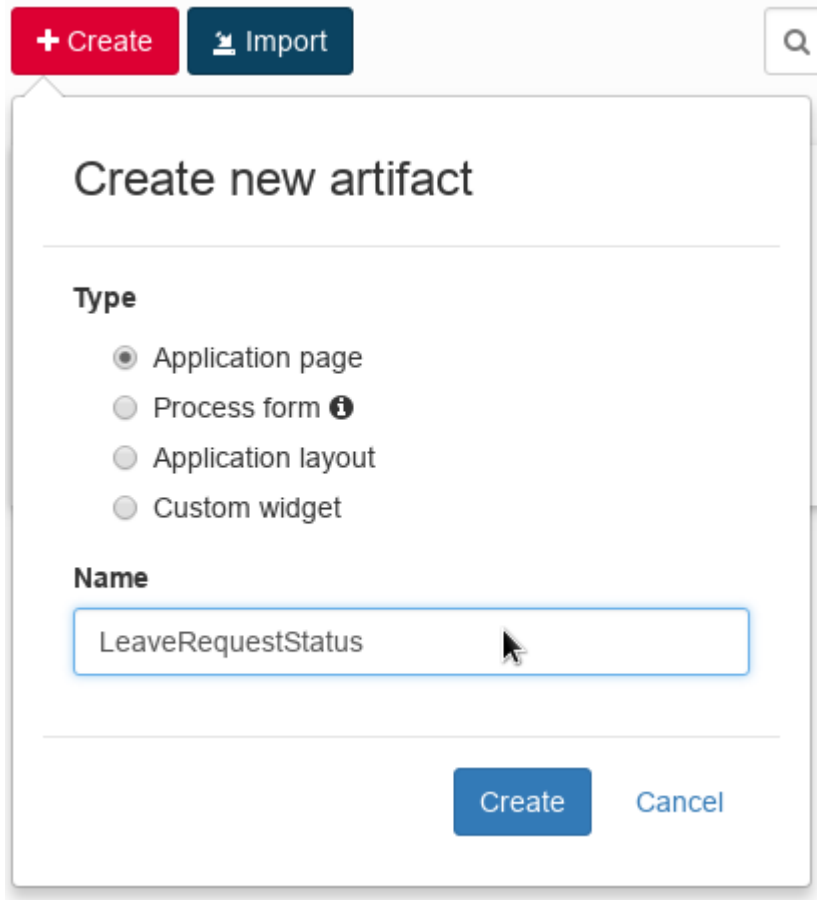
Optional: Add a date widget and an input widget to collect new leave request information in the page. Then add a submit button to start a new leave request.

## 3. Step by step instructions

### 1. Create a new page.

In the Studio, click on the "UI Designer" button.

Click on the "Create" button, select "Application page", set the name to: LeaveRequestStatus. Then click on "Create".



**+ Create** **Import**

## Create new artifact

**Type**

- ☒ Application page
- ☐ Process form ⓘ
- ☐ Application layout
- ☐ Custom widget

**Name**

LeaveRequestStatus

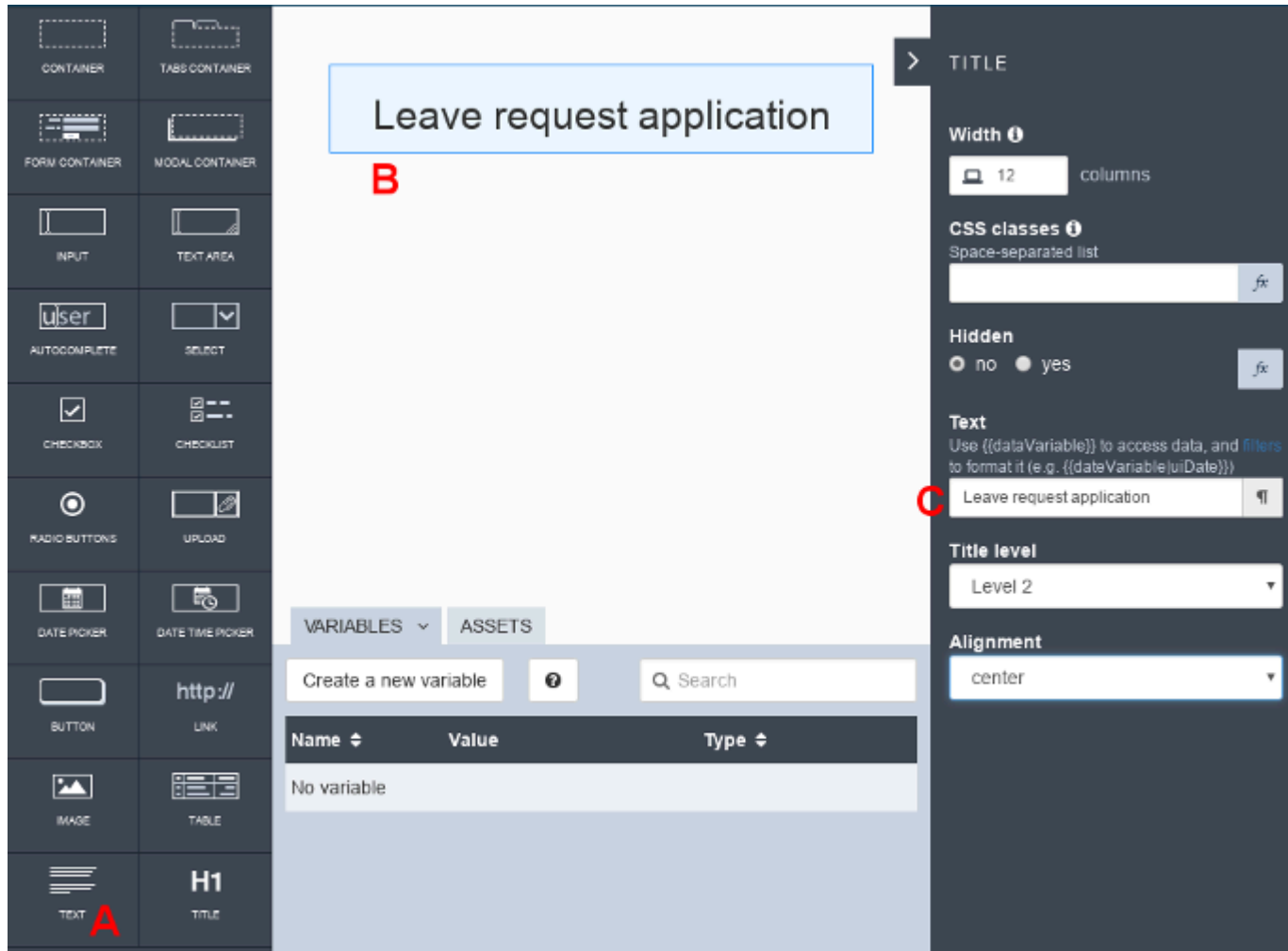
**Create** **Cancel**

You should now be on the designer page.

## 2. Add a title to your page.

Drag the widget Title (A) from the palette to the top of the page (B).

Select the widget. On the right panel, enter "Leave request application" in the "Text" field (C). Select "center" for alignment.



**3. Add another title below.**

Name it "Leave request status".

Select "Level 4" from the Title level drop-down list. Select "center" for alignment.

**4. Create a variable to store the session information.**

Click on "Create a new variable" and choose "External API" for the type.

Name it "sessionInfo"



In the field API URL, enter:

API/system/session/unusedId

**Create a new variable**

**Name \***

sessionInfo

**Type**

External API ▼

**API URL**

../API/system/session/unusedId

💡 **Tip:** You can use data in the URL, by using the syntax `{{dataName}}`

💡 **Tip:** You can extend our REST API capabilities by creating *REST API extensions* in the *Studio Development* menu and use them here

[See examples](#)

**Save** **Cancel**

**5. Create a variable to list the leave request.**

Click on "Create a new variable" and choose "External API".

Name it "leaveRequestStatus".

In the field API URL, enter: (no line break)

```
API/bdm/businessData/com.company.model.LeaveRequest?  
q=findByRequestorId&p=0&c=100&f=requestorId={{sessionInfo.user_id}}
```

Be careful when you copy/paste the URL that there is no whitespace left.

**6. Define a new JavaScript expression to prepare the list.**

Click on "Create a new variable" and choose "JavaScript expression".

Name it "updateLeaveRequestStatus".

---

Replace the existing value with the following script:

```
for (let line of $data.leaveRequestStatus) {  
  if (line.isApproved === null) {  
    line.isApprovedLabel = "In progress";  
  } else if (line.isApproved) {  
    line.isApprovedLabel = "Approved";  
  } else {  
    line.isApprovedLabel = "Rejected";  
  }  
}
```

## 7. Create a multiple container.

Drag the widget container from the palette and place it below the Title "Leave request status".

Select the container and on the right panel, enter "leaveRequestStatus" in the field "Collection".

## 8. Add 4 widgets in the container.

One "Input" widget with the following options:

Property	Value
Width	3
Read-Only	Yes
Label	Num
Value	\$index + 1

One "Date picker" widget with the following options:

Property	Value
Width	3
Read-Only	Yes
Label	Start date
Value	\$item.leaveStart
Show Today button	no

One input widget with the following options:

Property	Value
Width	3
Read-Only	Yes
Label	Number of days
Value	\$item.dayCount

One "Input" widget with the following options:

Property	Value
Width	3
Read-Only	Yes
Label	Status

---

Property	Value
Value	\$item.isApprovedLabel

Save the page. The page should look like this:


Leave request application

Leave request status

Num

data:\$index+1

Start date

data:\$item.leave:

Number of days

data:\$item.dayCount


Status

data:\$item.isApproved

You can preview the page at anytime by clicking on preview.

**Tip:** If you are logged in the Portal in the same browser, the current leave request will be displayed.

9. **Export the page.**

Click on the export button  to download the page in ZIP format.

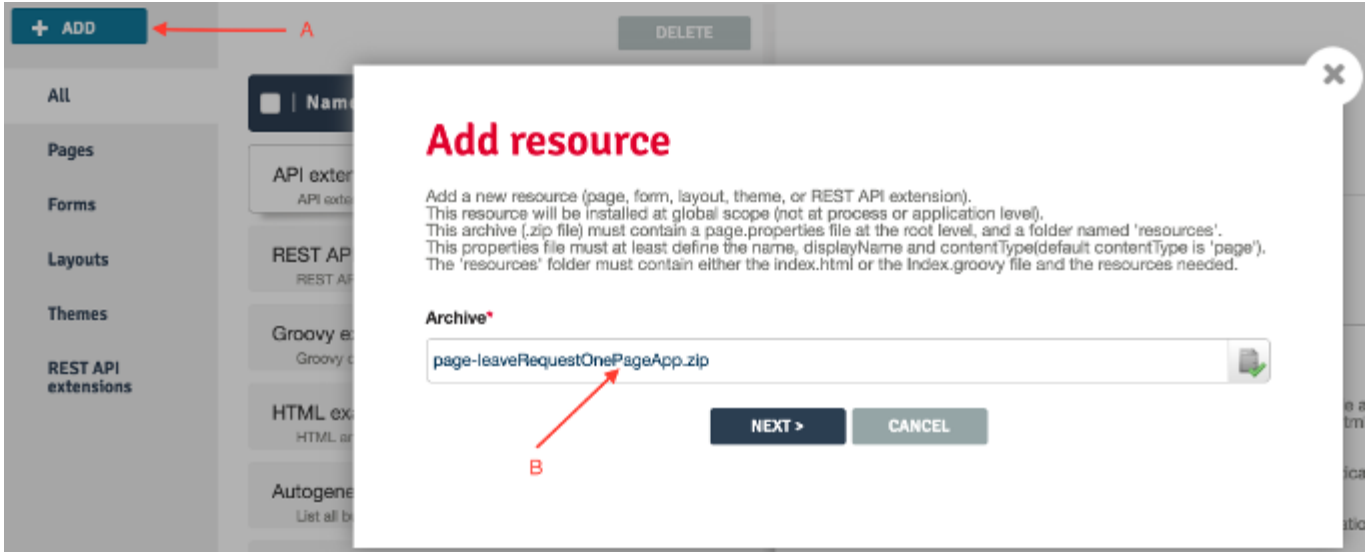
10**Import the page in the Portal.**

Log in the Bonita Portal with a Administrator user.

Change the profile to "Administrator". Go to Resources tab.

Click on the button "Add" (A)

Select (B) the ZIP file and click on Next. Then confirm on the next screen.



The page should now be available in the Pages list.

## 11 Create a new application.

In the Portal, navigate to the Applications tab.

Click on New.

Enter "Leave request app" in the field "Display name" (A).

Enter "leave-request" in the field "URL" (B).

Leave the defaults value for the Version (C) and Profile (D).

## Create an application

Display name \*

A Leave request app

URL \*

B ../apps/ leave-request

Version \*

C 1.0

Profile \*

D User

Description


CREATE

CANCEL

Click on "Create" to create the application.

Click on "..." to access the configuration page.

Find the Pages section on the bottom left part of the screen and Click on "Add". Select the page LeaveRequestStatus and enter "status" in the URL field.

Define the leaveRequest page as the homepage by clicking on the "house" icon .

Once done, click on the trash icon  to delete the default homepage.

The configuration page should now look like that:

[BACK](#)[EDIT](#)

## Leave request app (1.0)

**URL** [../apps/leave-request](#)  
**Profile** User

**Creation on** 2018-08-07 09:58:47  
**Created by** Walter Bates

**Updated on** 2018-08-07 09:58:47  
**Updated by** Walter Bates

## Look & Feel

**Layout** Default layout

**Theme** Bootstrap default theme

## Pages

[ADD](#)

status - LeaveRequestStatus page



## Navigation

[ADD](#)

Specify the menu structure.  
You must add a page before you can reference it in a menu.

Click on the link "apps/leave-request" to load the application.

Num	Start date	Number of days	Status
1	11/14/2016	5	In progress
2	12/26/2016	2	In progress

At this point, the exercise is done. However, if you have more time you can follow the next steps to add more features to your application.

### 12 Add a new form container.

Go back to edit your page in the UI Designer.

Drag a form container from the palette and place it between the two titles.

### 13 Create a new variable to store the new leave request values.

Click on "Create a new variable" and choose "JSON".

---

Name it "formInput".

Enter the following script in the value text field:

```
{
  "requestInput" : {
    "leaveStart" : null,
    "dayCount" : null
  }
}
```

#### 14 Create a new variable to store the process information.

Click on "Create a new variable" and choose "External API".

Name it "processDefinitionInfo".

In the field API URL, enter: (no line break)

API/bpm/process?p=0&c=100&o=version%20DESC&f=name=LeaveRequest

Be careful when you copy/paste the URL that there is no whitespace left.

#### 15 Add 2 widgets in the form container.

One date picker widget with the options: Width: 6. Value: formInput.requestInput.leaveStart. Label: Leave start date

One input widget with the options: Width: 6. Value: formInput.requestInput.dayCount. Label: Number of days

#### 16 Add a submit button in the form container.

Drag the button widget from the palette and place it in the form container below the two widgets.

Enter "Create a new request" in the field Label.

Select POST in the Action drop-down list.

Click on "fx" to switch the "Data sent on click" field mode and then enter "formInput".

In the field URL to call, enter:

API/bpm/process/{ {processDefinitionInfo[0].id} }/instantiation

In the field Target URL Success, enter:

/bonita/apps/leave-request

Save the page. The page should look like that:

# Leave request application

Leave start

data:requestInput.requestInput.leaveStart



Number of days

data:requestInput.requestInput.dayCount

Create a new request

## Leave request status

Num

data:\$index+1

Start date

data:\$item.leave



Number of days

data:\$item.dayCount

Status

data:\$item.isApproved

You can preview the page to make sure it works as expected.

### 17 Update the page in the portal.

Export the updated page.

Log in the portal and edit the previous page.

Select the new zip file.

Refresh your application, the change should be visible.

localhost:8080/bonita/apps/leaveRequest/status/

Apps Tesla logo JBoss REST eXo Bonita blog NYC guitar Daylighted Tomato Timer Adoption Perso Order-demo Soccer Set Platform

Leave Request

### Leave request status

Leave start: Enter a date (mm/dd/yyyy) [Calendar icon]

Number of days: [Input field]

Create a new request

Leave request table

Num	Start date	Number of days	Status
1	11/14/2016 [Calendar icon]	5	In progress
2	12/26/2016 [Calendar icon]	2	In progress