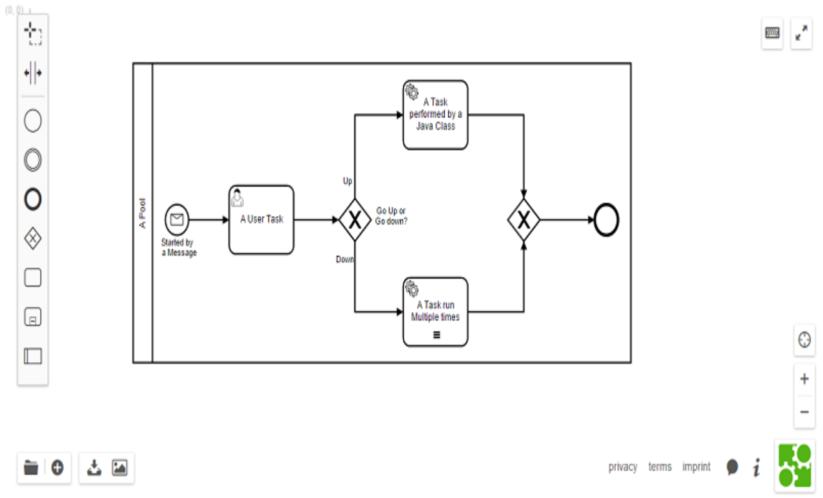
BPMN - Enactment

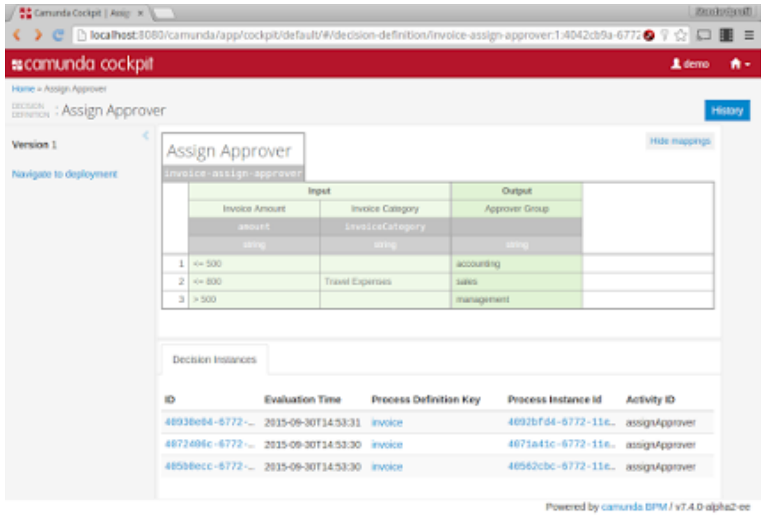
Camunda

2 Steps

- Process Modeling



- **Deployment and Test**



Camunda Cockpit | Assig
localhost:8080/camunda/app/cockpit/default/#/decision-definition/invoice-assign-approver:1:4042cb9a-6772
camunda cockpit
demo
Home » Assign Approver
DECISION DEFINITION
History
Hide mappings

: Assign Approver

Version 1

Navigate to deployment

Assign Approver

invoice-assign-approver

	Input		Output
	Invoice Amount	Invoice Category	Approver Group
	amount	invoiceCategory	
	string	string	string
1	<= 500		accounting
2	<= 800	Travel Expenses	sales
3	> 500		management

Decision Instances

ID	Evaluation Time	Process Definition Key	Process Instance Id	Activity ID
48938e84-6772-...	2015-09-30T14:53:31	invoice	4892bfd4-6772-11e...	assignApprover
4872486c-6772-...	2015-09-30T14:53:30	invoice	4871a41c-6772-11e...	assignApprover
485b8ecc-6772-...	2015-09-30T14:53:30	invoice	48562cbc-6772-11e...	assignApprover

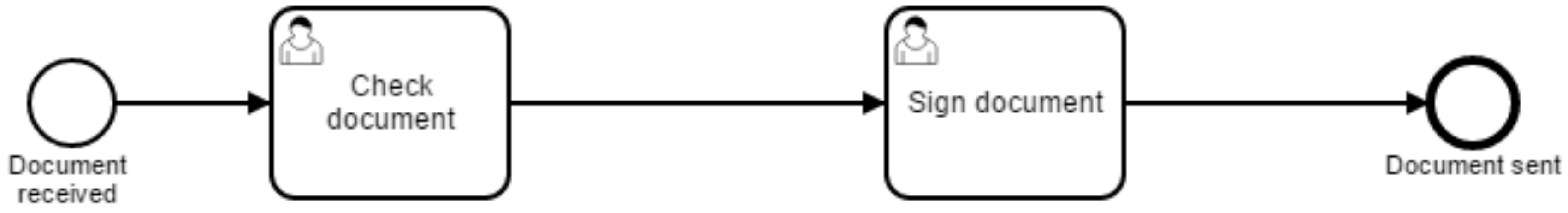
Powered by camunda BPM / v7.4.0-alpha2-ee

PROCESS MODELING

A very simple process

When a document is received by the administrative office, it is first checked for errors, and then is signed by a secretary before it is sent out again.

Create the model of the process using **Camunda Modeler** (https://camunda.org/download/modeler/) .

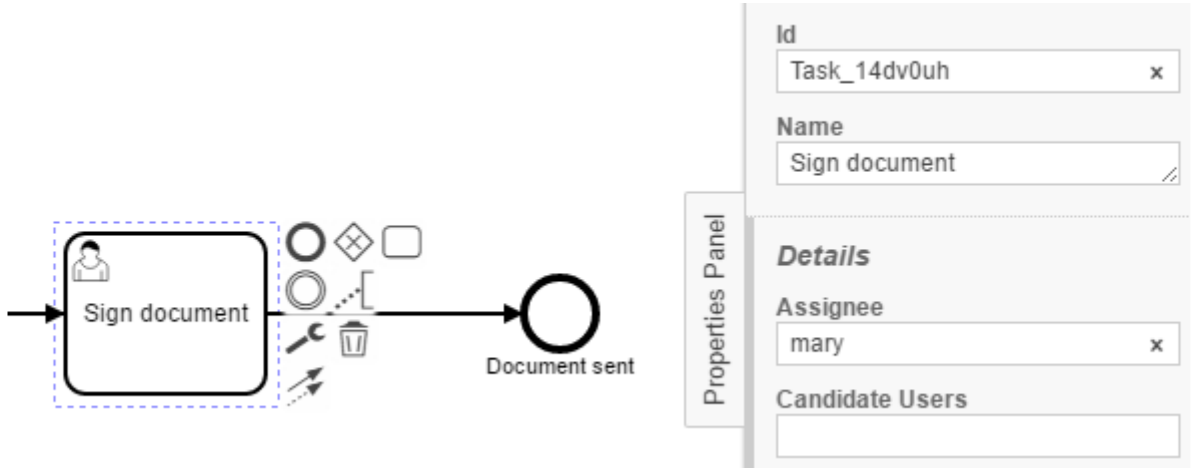


Since the tasks will be performed by human users, set them as user tasks from the "Change Type" menu.

Set tasks assignees

Tasks must be assigned to the employees of our organization. We will use the default names provided by the Camunda BPMN.

From the **Properties Panel**, set john as the assignee for the first task, and mary as the assignee for the second one.



Id
Task_14dv0uh

Name
Sign document

Details

Assignee
mary

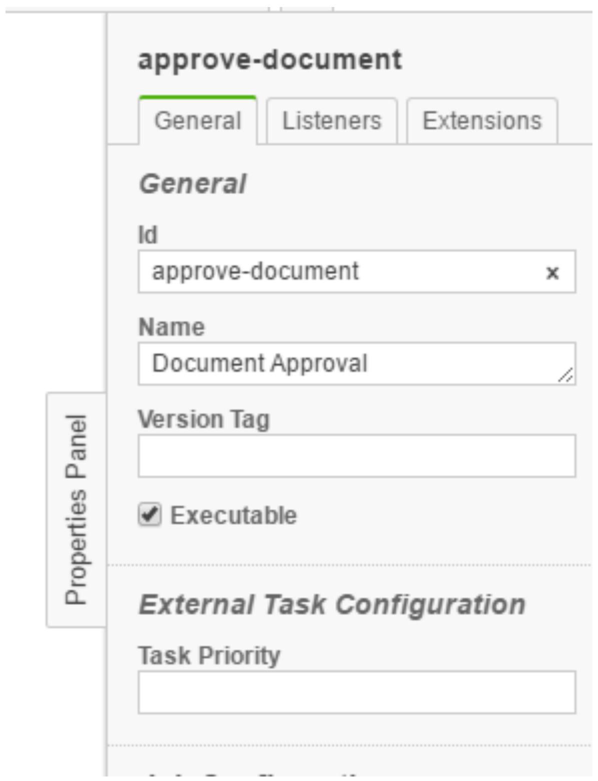
Candidate Users

Setup process properties

Click on any blank place in the canvas and set the process-wide properties from the **Properties Panel**.

Id and **Name** are the way the process will be identified in the Camunda BPM platform, for managing and launching it.

The **Executable** checkbox allows the process to be launched by the BPM platform.



approve-document

General

Id: approve-document

Name: Document Approval

Version Tag:

✓ Executable

External Task Configuration

Task Priority:

LIBRARIES AND ENVIRONMENT

Libraries and Environment

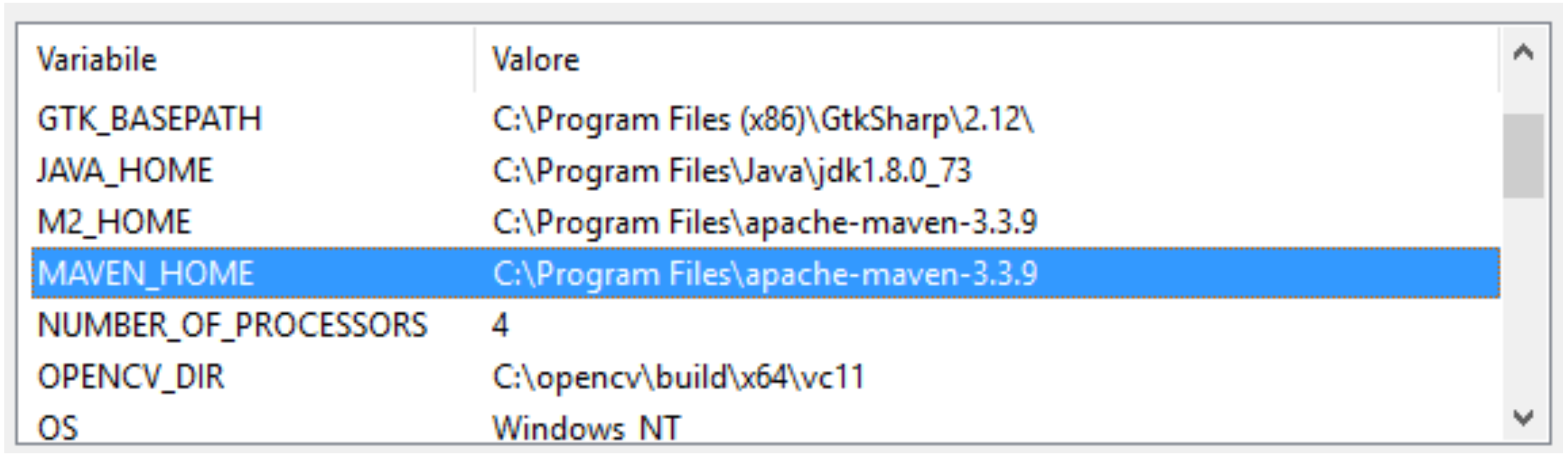
In order to make the Camunda BPM platform work, a Java environment, equipped with Maven, is needed.

Make sure that JDK 1.8 and Apache Maven libraries are installed:

- download latest JDK from http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- download latest Maven version from http://maven.apache.org/install.html and install it in any folder
- download and install Eclipse: https://eclipse.org/downloads/

Libraries and Environment

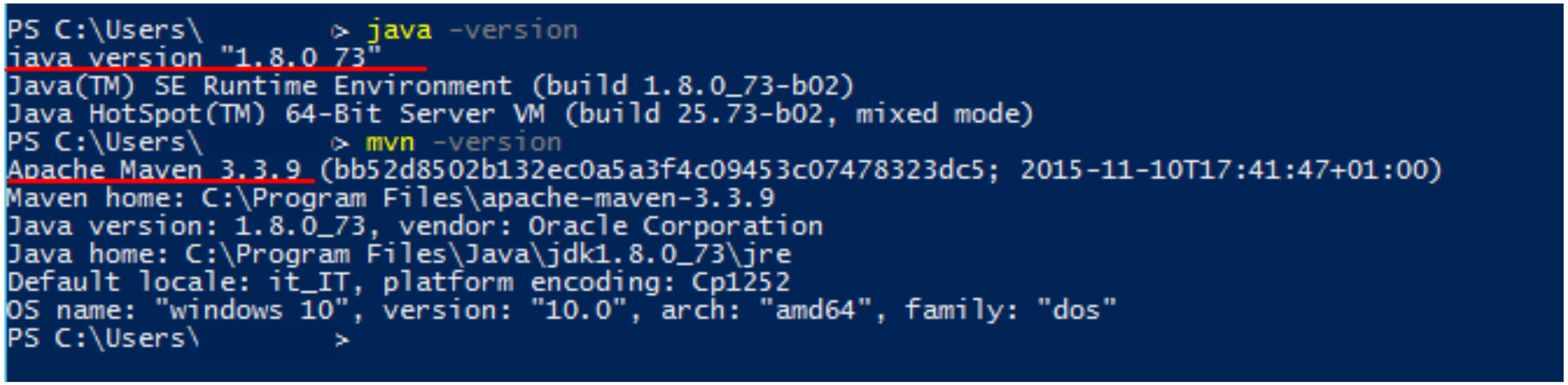
Set environment variables for Java Home and Maven Home, to the folders where the libraries have been downloaded.



Variabile	Valore
GTK_BASEPATH	C:\Program Files (x86)\GtkSharp\2.12\
JAVA_HOME	C:\Program Files\Java\jdk1.8.0_73
M2_HOME	C:\Program Files\apache-maven-3.3.9
MAVEN_HOME	C:\Program Files\apache-maven-3.3.9
NUMBER_OF_PROCESSORS	4
OPENCV_DIR	C:\opencv\build\x64\vc11
OS	Windows_NT

Libraries and Environment

You can verify the currect installation of both the libraries by launching the following commands in terminal (e.g., PowerShell on Windows).



```
PS C:\Users\        > java -version
java version "1.8.0_73"
Java(TM) SE Runtime Environment (build 1.8.0_73-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.73-b02, mixed mode)
PS C:\Users\        > mvn -version
Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5; 2015-11-10T17:41:47+01:00)
Maven home: C:\Program Files\apache-maven-3.3.9
Java version: 1.8.0_73, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.8.0_73\jre
Default locale: it_IT, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "dos"
PS C:\Users\        >
```

CAMUNDA PLATFORM

Camunda Platform Setup

Download Camunda Tomcat distribution from https://camunda.org/download/;

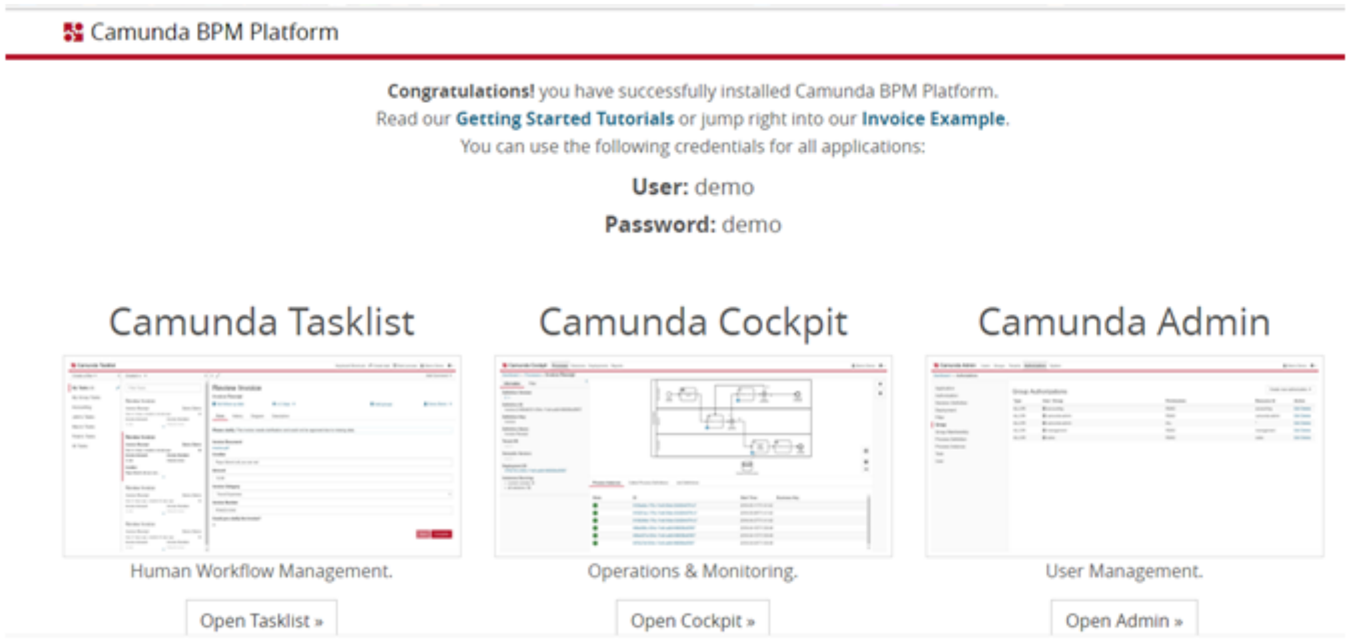
Unzip to any folder (we will call the path $CAMUNDA_HOME);

Move with terminal or powershell to $CAMUNDA_HOME;

Launch start-camunda.bat (on Windows) or start-camunda.sh (on Linux).

Camunda Platform Setup

The Camunda Welcome homepage should open automatically in the default browser, at this point (otherwise go to localhost:8080/camunda-welcome/index.html).



Camunda BPM Platform

Congratulations! you have successfully installed Camunda BPM Platform.
Read our **Getting Started Tutorials** or jump right into our **Invoice Example**.
You can use the following credentials for all applications:

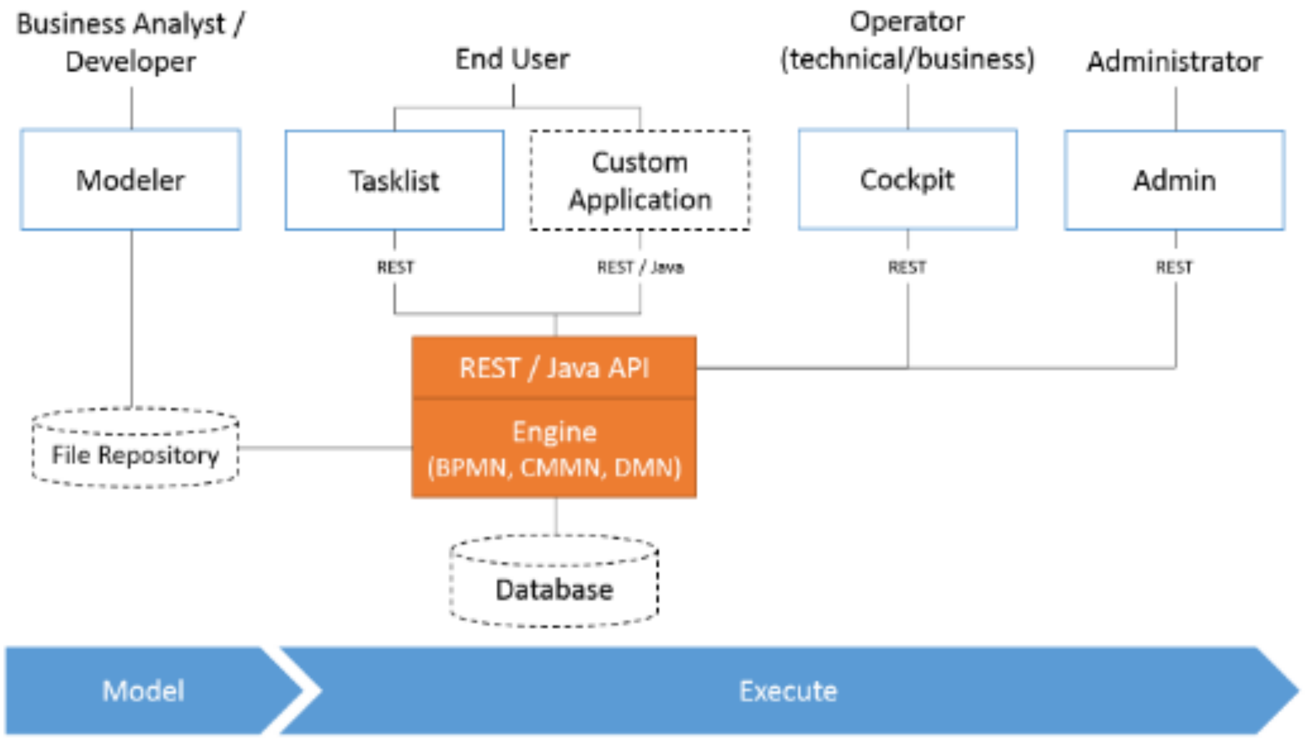
User: demo

Password: demo

SoftEng http://softeng.polito.it

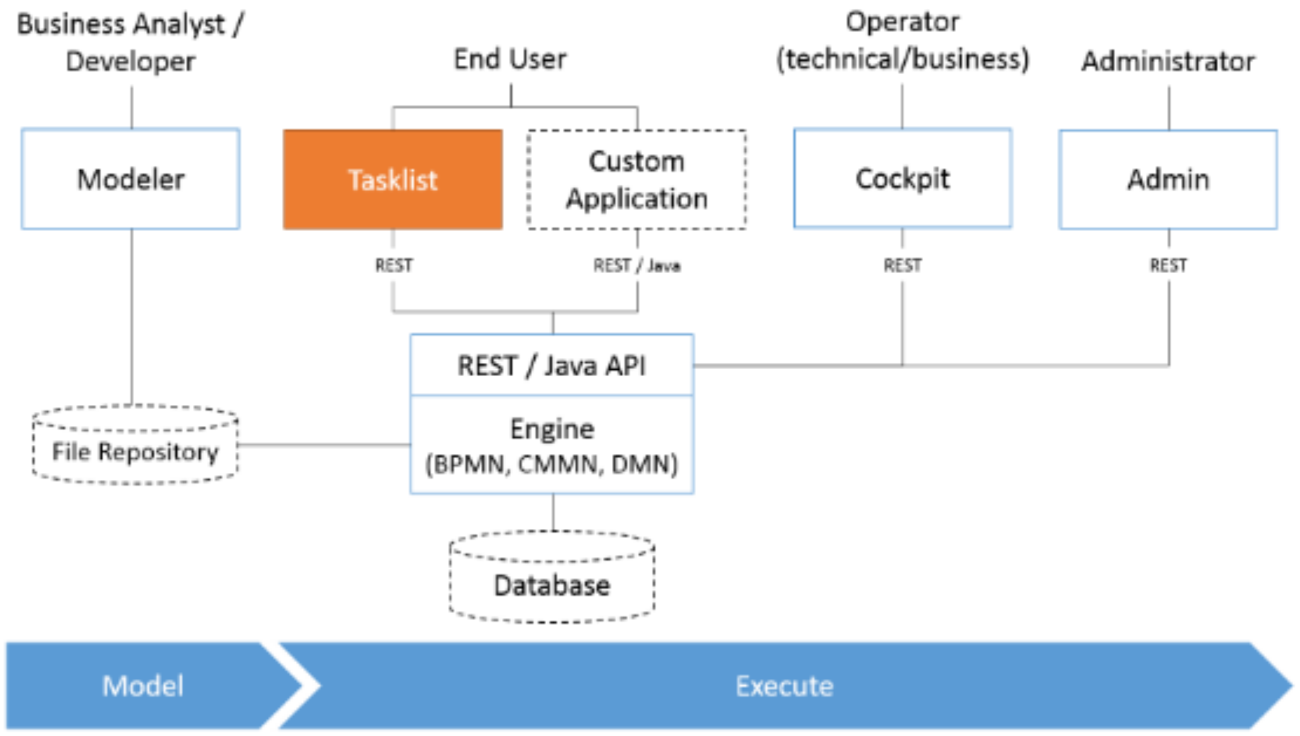
Platform overview

The core of Camunda is the **execution engine**, that can be accessed by web applications through WEB API.



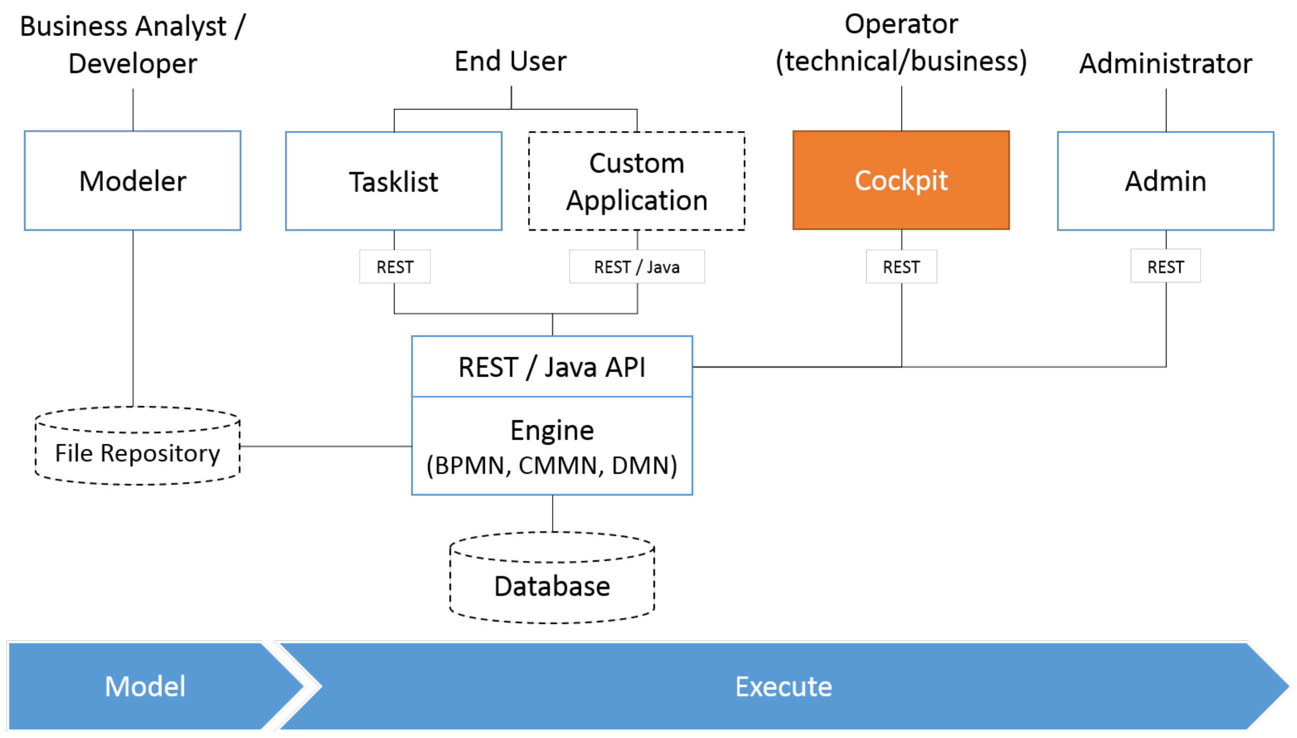
Platform overview

Tasklist allows all end user to check what are the tasks they need to work on, their process and their history.



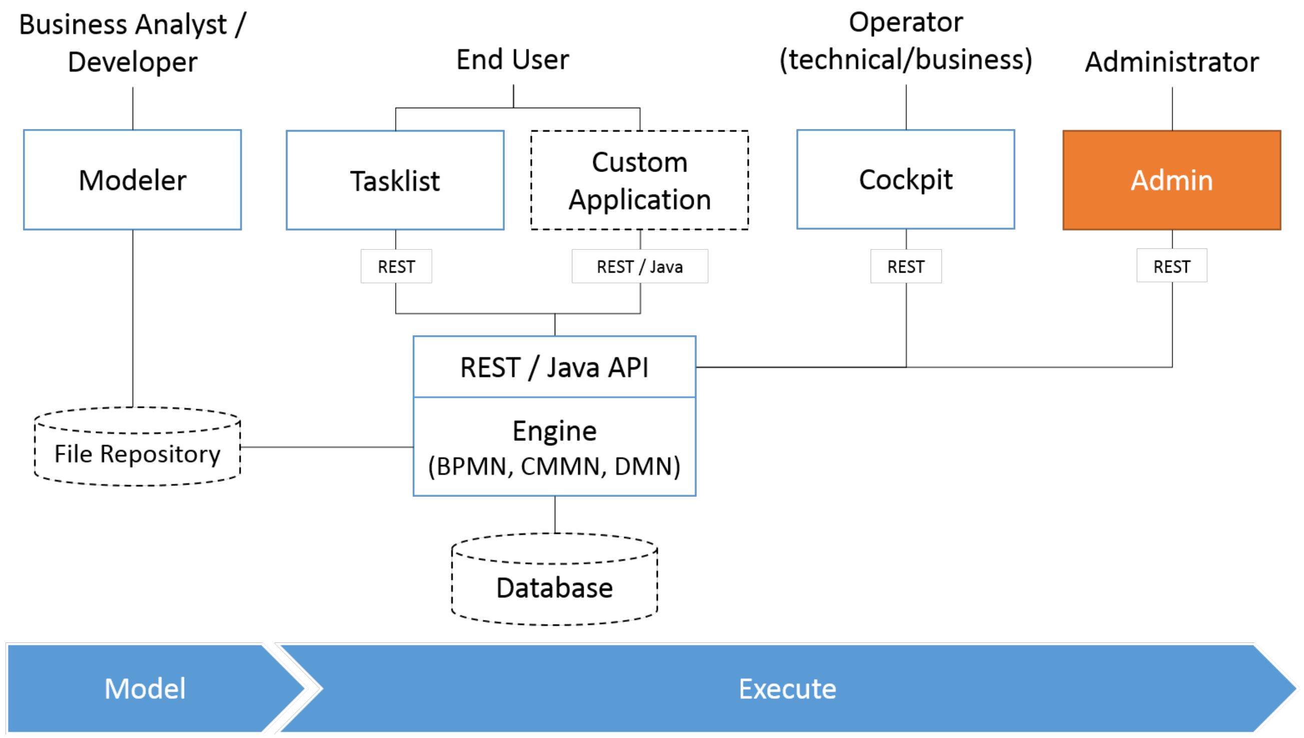
Platform overview

Cockpit allows to inspect all running instances of processes and to take repair actions in case exceptions happen.



Platform overview

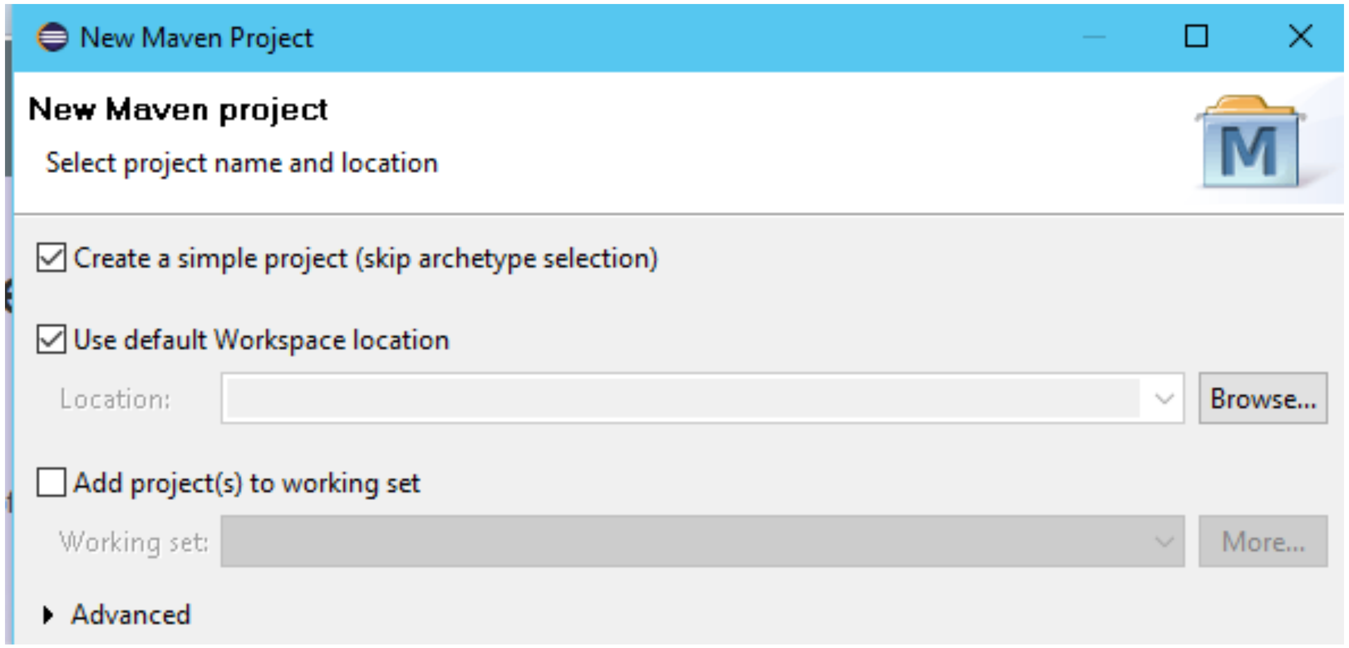
Admin allows the administrators of the system to have control on all the system users and their permissions.



PROJECT SETUP AND DEPLOYMENT

Project Setup

Create a **Java Maven Project** (File / New / Other / Maven / Maven Project) in Eclipse using the following settings:



New Maven Project

New Maven project

Select project name and location

[✓] Create a simple project (skip archetype selection)

[✓] Use default Workspace location

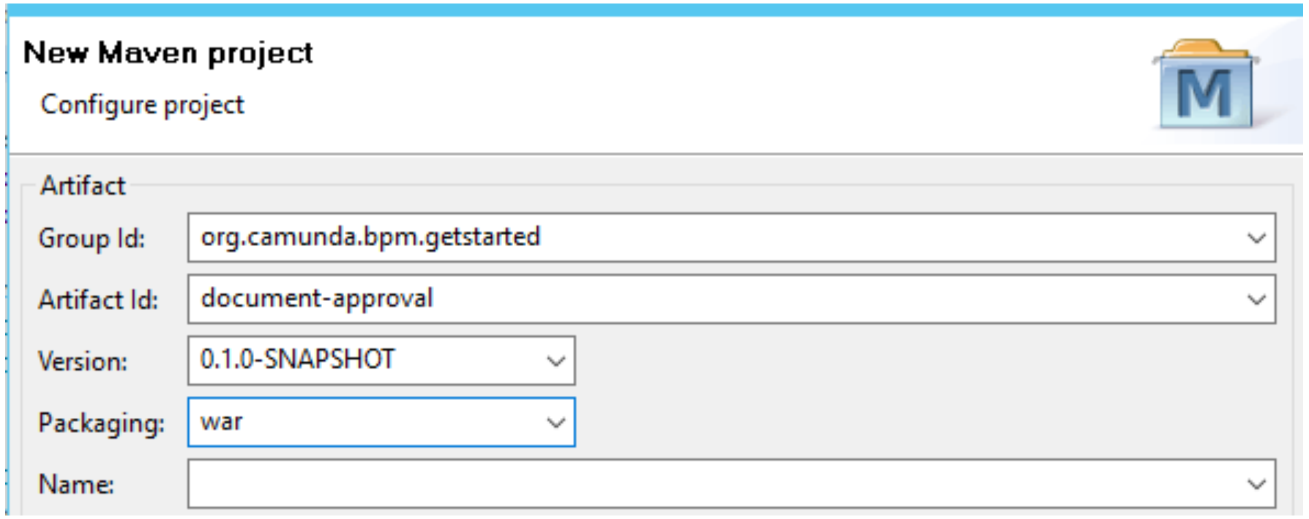
Location:

[ ] Add project(s) to working set

Working set:

Project Setup

Create a **Java Maven Project** (File / New / Other / Maven / Maven Project) in Eclipse using the following settings:


New Maven project

Configure project

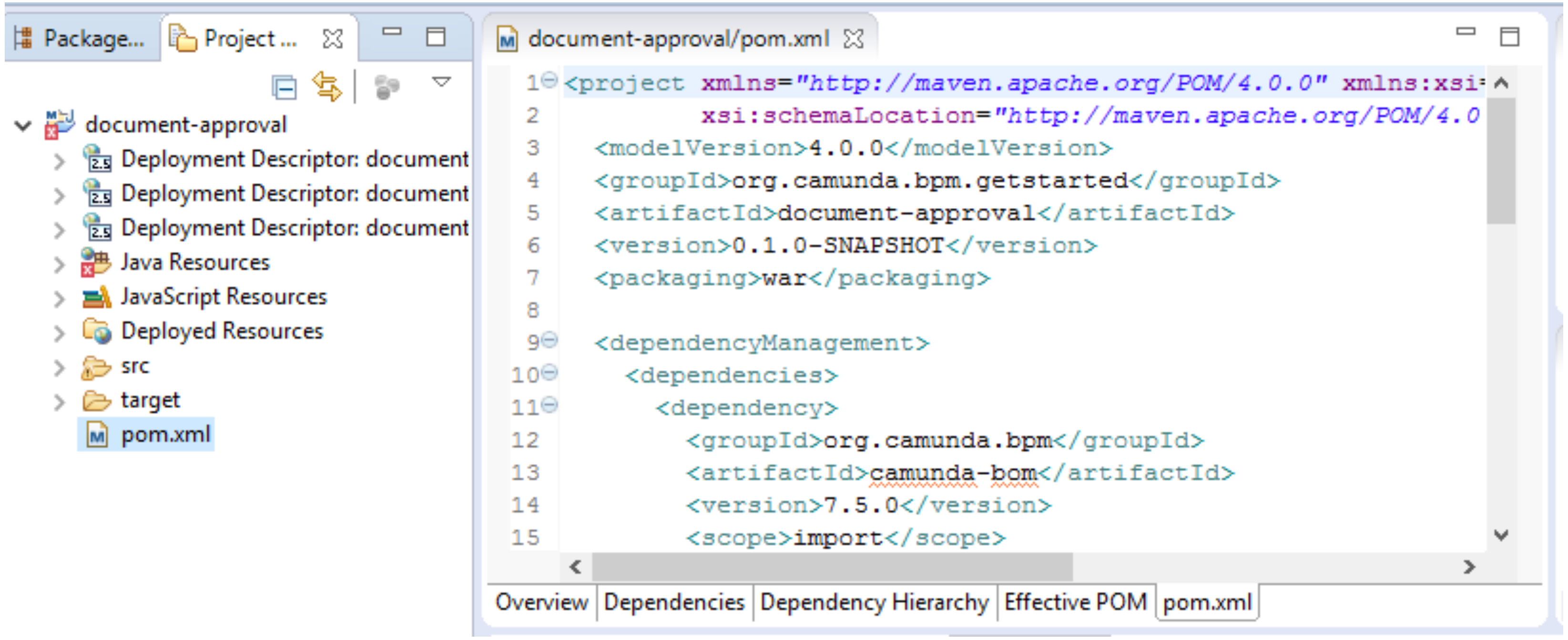
Artifact

Group Id:	org.camunda.bpm.getstarted
Artifact Id:	document-approval
Version:	0.1.0-SNAPSHOT
Packaging:	war
Name:	

Project Setup

Add the Camunda Maven dependencies to the pom.xml file of the project.

To do so, you can copy and paste the code in the provided pom.xml file:



```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi=
         xsi:schemaLocation="http://maven.apache.org/POM/4.0
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.camunda.bpm.getstarted</groupId>
  <artifactId>document-approval</artifactId>
  <version>0.1.0-SNAPSHOT</version>
  <packaging>war</packaging>

  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.camunda.bpm</groupId>
        <artifactId>camunda-bom</artifactId>
        <version>7.5.0</version>
        <scope>import</scope>
```

Project Setup

Add a new source code package (org.camunda.bpm.getstarted) and a class child of ProcessApplication named DocumentApprovalApplication.java.

```
package org.camunda.bpm.getstarted;

import org.camunda.bpm.application.ProcessApplication;
import org.camunda.bpm.application.impl.ServletProcessApplication;

@ProcessApplication(“Document Approval“)
public class DocumentApprovalApplication extends
ServletProcessApplication {
  // empty implementation
}
```

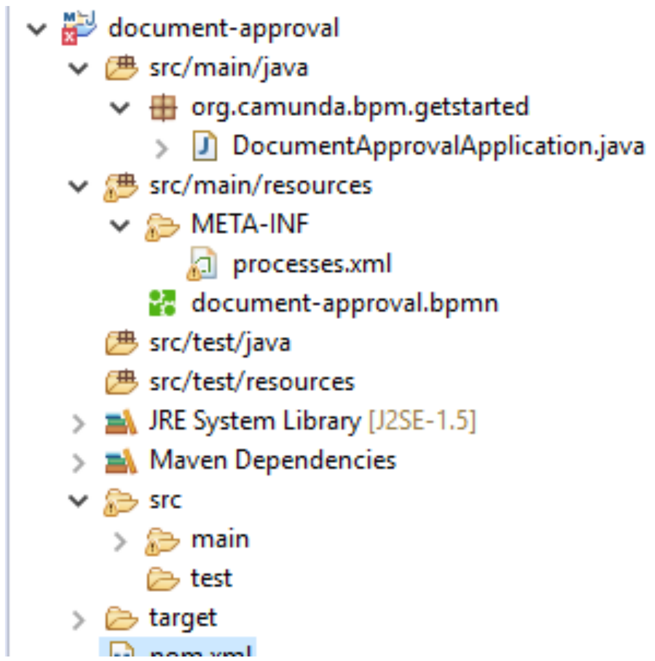
Project Setup

Create the file /src/main/resources/META-INF/processes.xml with the following code.

```
<?xml version="1.0" encoding="UTF-8" ?>

<process-application xmlns="http://www.camunda.org/schema/
1.0/ProcessApplication"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <process-archive name="document-approval">
    <process-engine>default</process-engine>
    <properties>
      <property name="isDeleteUponUndeploy">false</property>
      <property name="isScanForProcessDefinitions">true</
property>
    </properties>
  </process-archive>

</process-application>
```

Project Setup

Move the document-approval.bpmn file you created before in the /src/main/resources folder of the project.

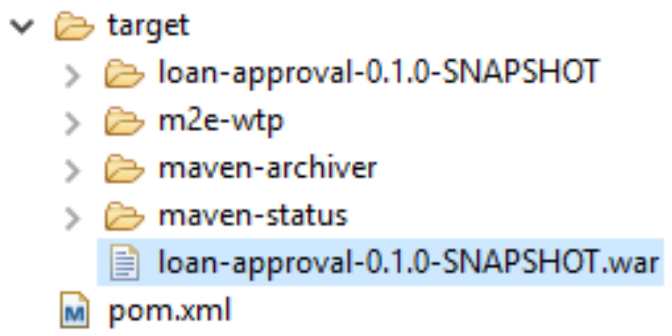


```
document-approval
  src/main/java
    org.camunda.bpm.getstarted
      DocumentApprovalApplication.java
  src/main/resources
    META-INF
      processes.xml
    document-approval.bpmn
  src/test/java
  src/test/resources
  JRE System Library [J2SE-1.5]
  Maven Dependencies
  src
    main
    test
  target
  pom.xml
```

Project Deployment

Perform a Maven Install: right click on the pom.xml file and select Run As / Maven Install.

Locate the .war file in the target folder of the project, and copy it in the $CAMUNDA_HOME/server/apache-tomcat.../webapps folder



```
target
  loan-approval-0.1.0-SNAPSHOT
  m2e-wtp
  maven-archiver
  maven-status
  loan-approval-0.1.0-SNAPSHOT.war
pom.xml
```

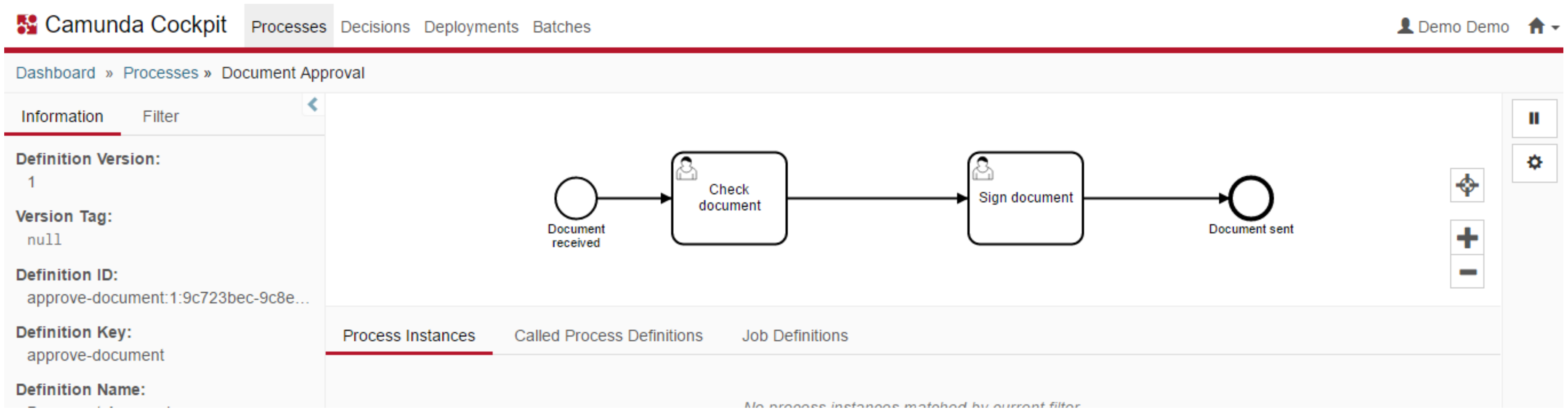
If the Tomcat server was already running, at this point the process should be deployed on the Camunda Platform.

TEST ON BROWSER

Process Test – See Processes

Go to **Camunda Cockpit** (http://localhost:8080/camunda/app/cockpit/). Login using the default admin username and password provided by Camunda: demo / demo.

From **Process Definitions** the BPMN diagram that was uploaded on Camunda Core can be seen.



Definition Version:
1

Version Tag:
null

Definition ID:
approve-document:1:9c723bec-9c8e...

Definition Key:
approve-document

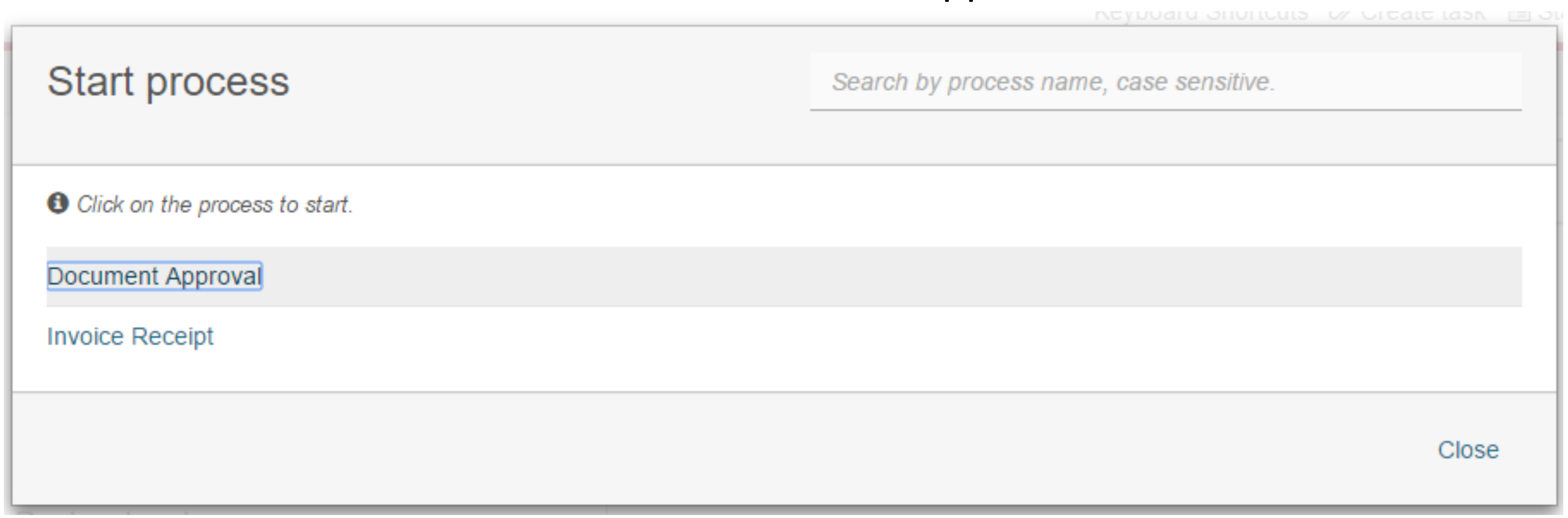
Definition Name:

No process instances matched by current filter

Process Test – Start a process

Go to **Camunda Tasklist** (http://localhost:8080/camunda/app/tasklist/). From this page we can see the tasks currently executing, and we can start a new process.

Click on **Start Process** and select Document Approval.



Start process

Search by process name, case sensitive.

Click on the process to start.

Document Approval

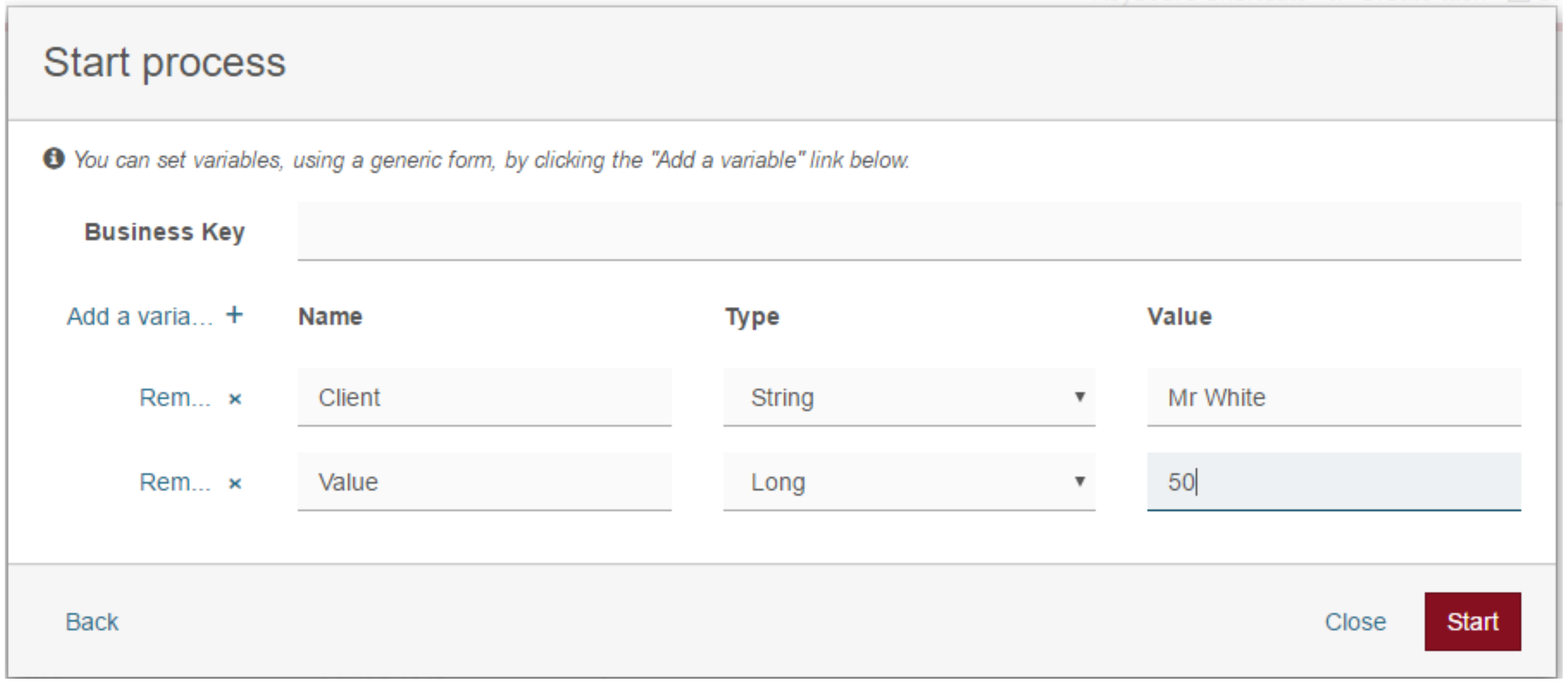
Invoice Receipt

Close

Process Test – Start a process

At this point we can set some variables for the process instance that we are launching.

Set some example variables like in the screen below and then click **start**.



Start process

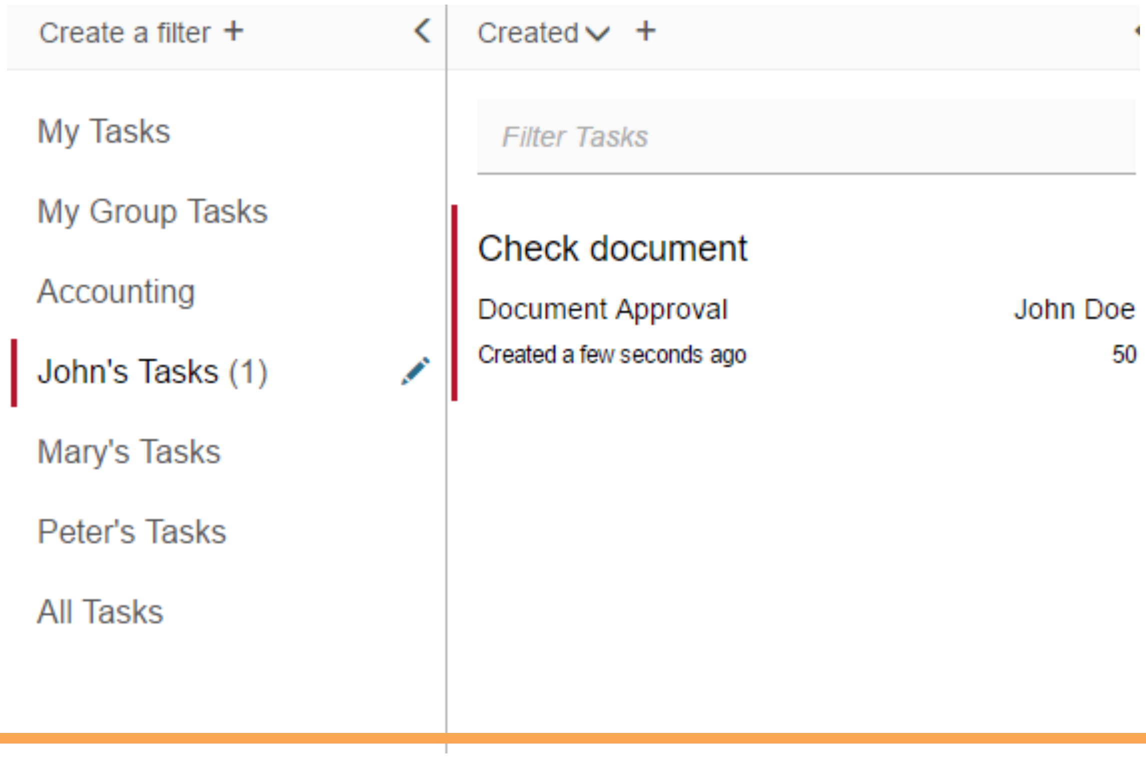
You can set variables, using a generic form, by clicking the "Add a variable" link below.

Business Key			
Add a varia... +	**Name**	**Type**	**Value**
Rem... ×	Client	String	Mr White
Rem... ×	Value	Long	50

Process Test – Check execution

The first human task that must be performed is John’s task.

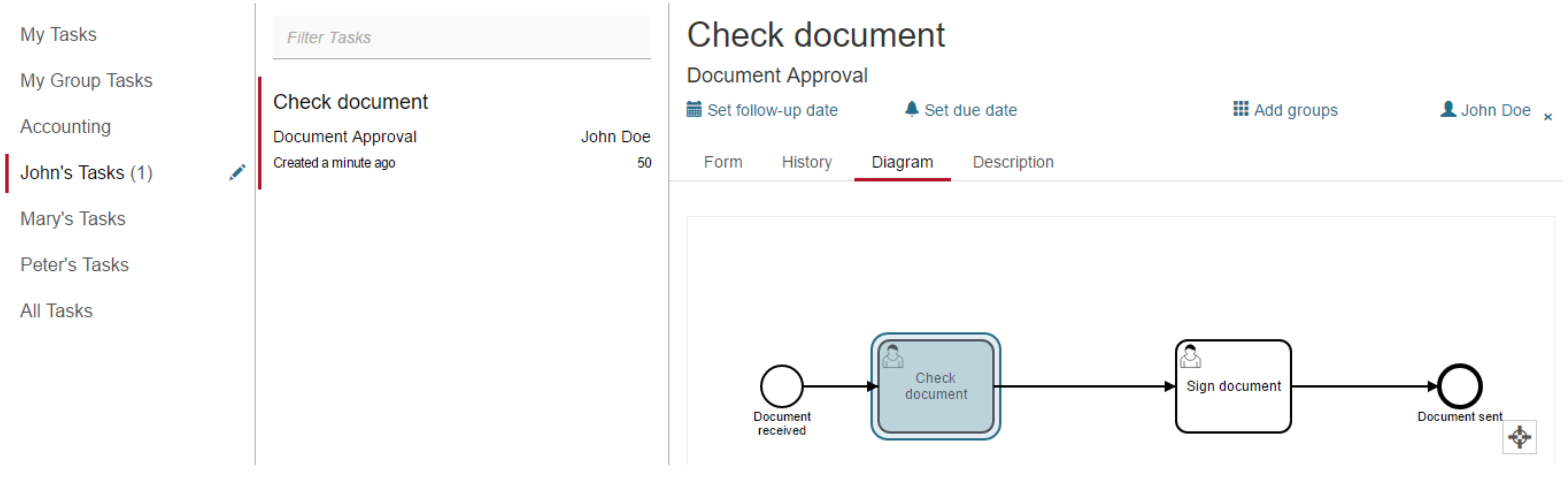
Click on **John’s Task** on the left (since we are logged as Admin, we can see all the tasks of different users).



Check document
Document Approval John Doe
Created a few seconds ago 50

Process Test – Check execution

We can see that John has to perform a task belonging to the process Check module. Clicking on Diagram on the right, we can see the progress on the whole task, currently stopped at “Check Document” and waiting for John’s operations.

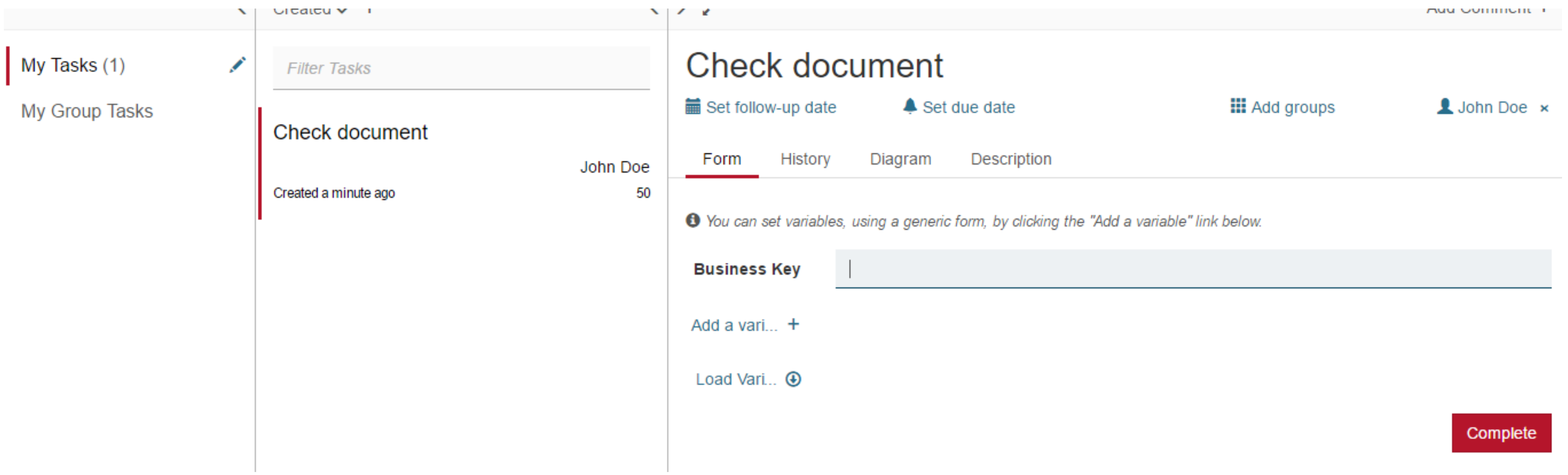


Process Test – Perform tasks

Logout from Tasklist, and **login as John** (default username and password: john / john).

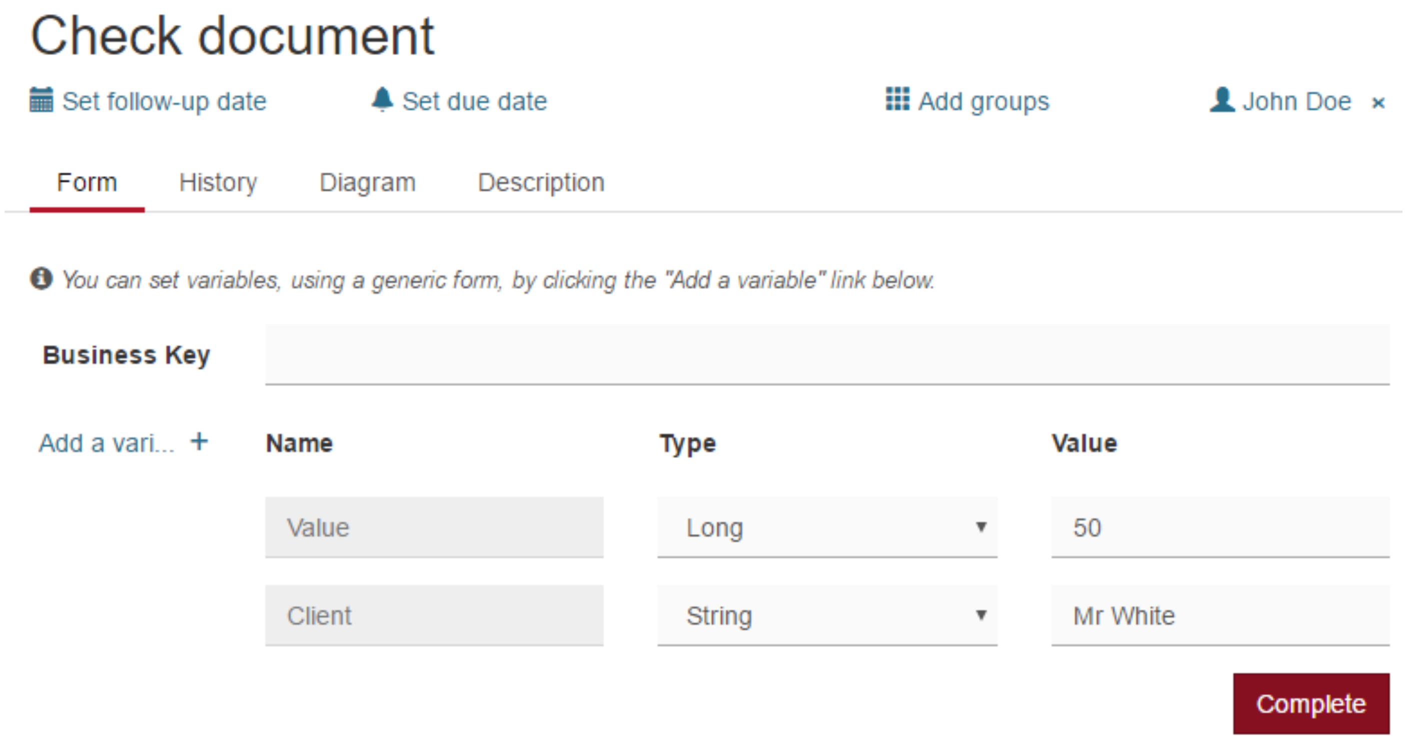
Logged as a user, we can find the task waiting for our execution, the Check Document of the Document Approval process we created before.

By clicking Load Variables, we can see the values that have been associated with this instance when it was created.



Process Test – Perform tasks

By clicking on Complete, we can perform our task and make the process go on.



Check document

Set follow-up date Set due date Add groups John Doe ×

Form History Diagram Description

You can set variables, using a generic form, by clicking the "Add a variable" link below.

Business Key

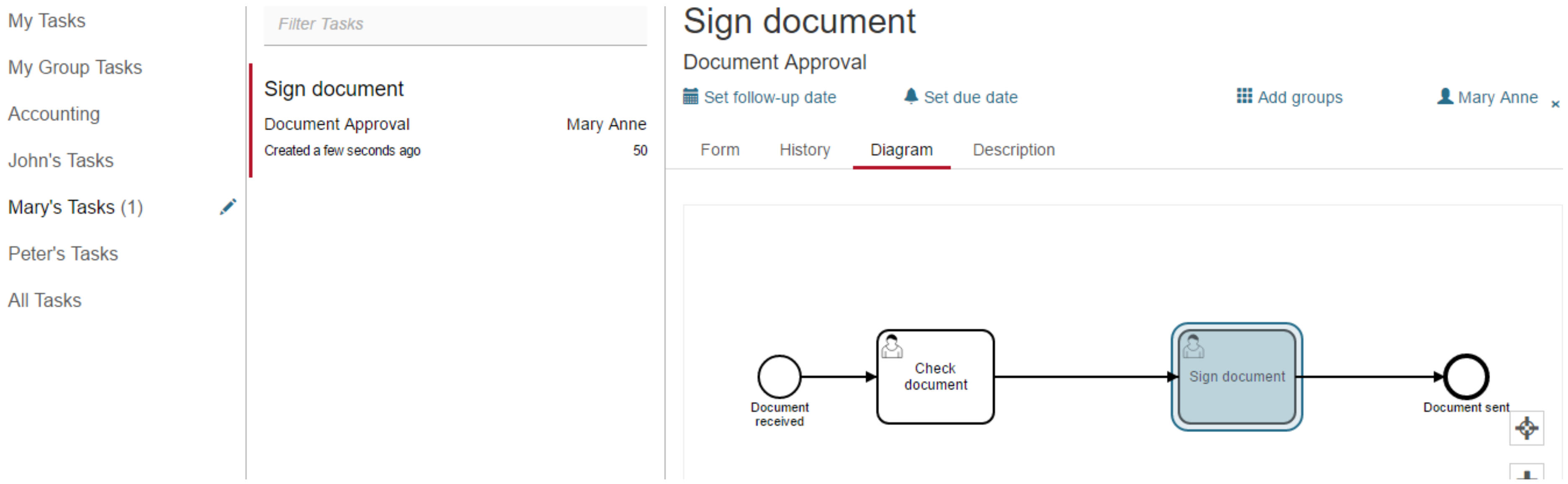
Add a vari... +

Name	Type	Value
Value	Long	50
Client	String	Mr White

Complete

Process Test – Check Execution

Now we can logout from John’s profile, and login again as the admin (demo / demo): we can see that the process has progressed, and now is awaiting for Mary to perform the task **Sign document**.



My Tasks
My Group Tasks
Accounting
John's Tasks
Mary's Tasks (1)
Peter's Tasks
All Tasks

Filter Tasks

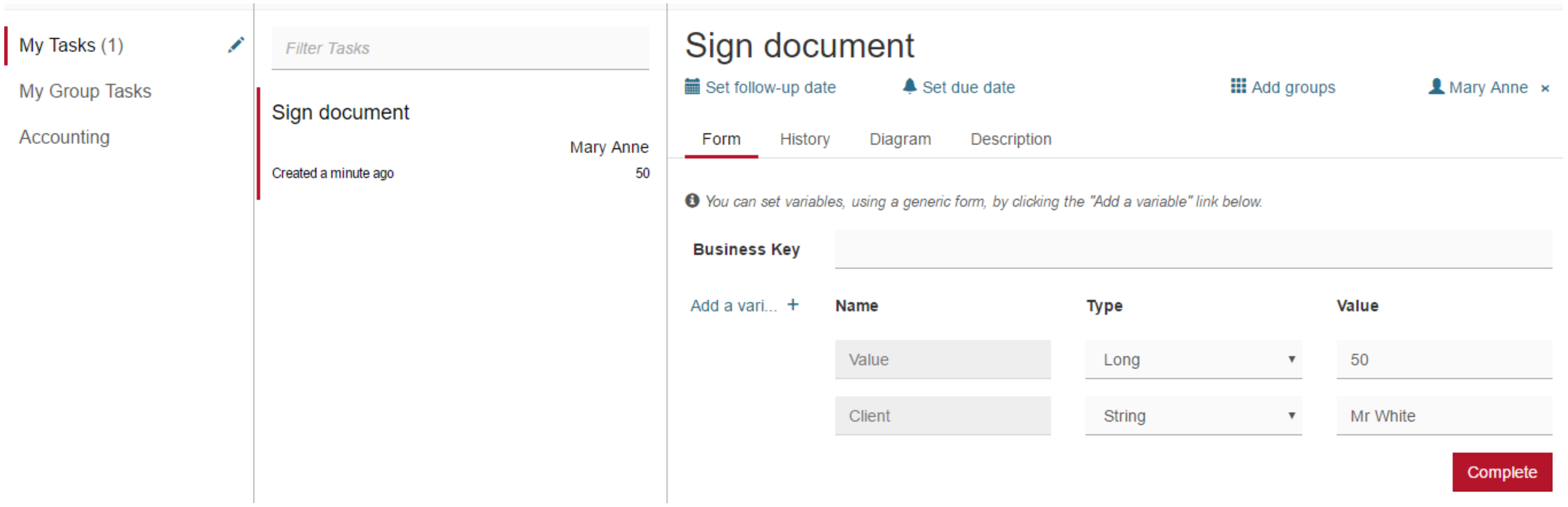
Sign document
Document Approval — Mary Anne
Created a few seconds ago — 50

Sign document
Document Approval
Set follow-up date · Set due date · Add groups · Mary Anne

Form · History · Diagram · Description

Process Test – Perform tasks

We log in with Mary's profile (mary / mary), and we see that we have a task awaiting in our task list (Sign document). We can perform the task by clicking Complete on it.



Sign document
Mary Anne
Created a minute ago
50

Sign document

You can set variables, using a generic form, by clicking the "Add a variable" link below.

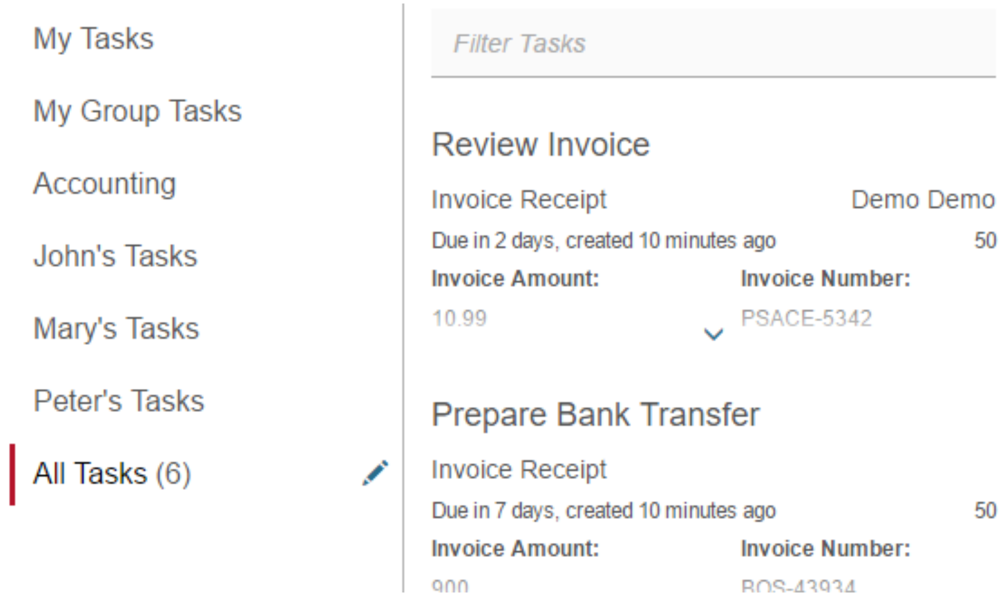
Business Key

Name	Type	Value
Value	Long	50
Client	String	Mr White

Process Test – Check Execution

Finally, we can log back with the administrator (demo / demo). We can see that at this point the process has ended.

Going back to “My Group’s Process” and checking “All Tasks”, we can see that there are no more tasks pertaining to our Document Approval process.



Review Invoice

Invoice Receipt — Demo Demo

Due in 2 days, created 10 minutes ago — 50

Invoice Amount: 10.99

Invoice Number: PSACE-5342

Prepare Bank Transfer

Invoice Receipt

Due in 7 days, created 10 minutes ago — 50

Invoice Amount: 900

Invoice Number: BOS-43934

USER TASK FORMS

SOftEng
http://softeng.polito.it

Process Test – User Task Forms

Camunda Tasklist supports four different kinds of task forms:

- Embedded Task Forms: HTML-based task forms, displayed embedded within Tasklist.
- Generated Task Forms: Like embedded task forms, but generated from XML metadata within the BPMN 2.0 XML.
- External Task Forms: The user is directed to another application to complete the task.
- Generic Task Forms: If no task form exists, a generic form is displayed to edit the process variables.

For instance, we can create three **Generated Task Forms**, respectively for the Start Event, for the task Check Module, and the task Sign Module.

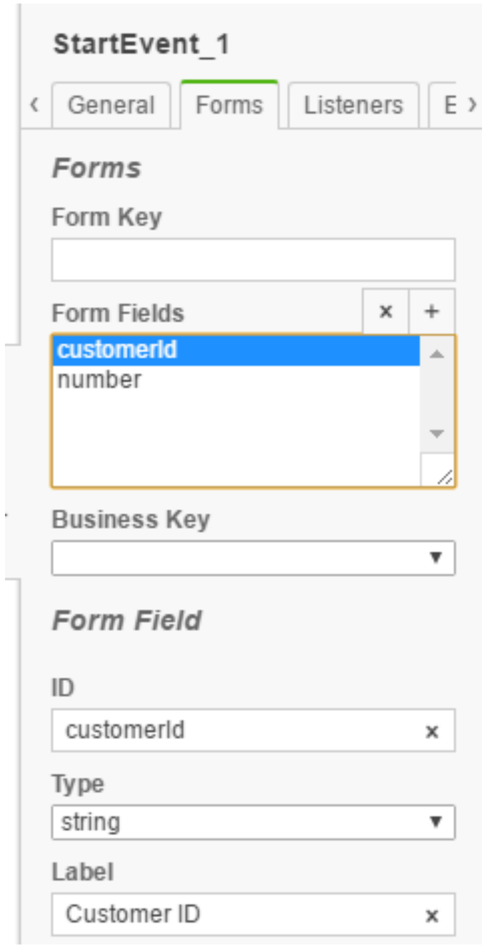
Process Test – User Task Forms

Add a form to the Start Event with two fields:

- customerId
- number

For each field, it is possible to specify:

- ID
- Type
- Label
- Default Value


StartEvent_1

Forms

Form Key:

Form Fields: customerId, number

Business Key:

Form Field

ID: customerId

Type: string

Label: Customer ID

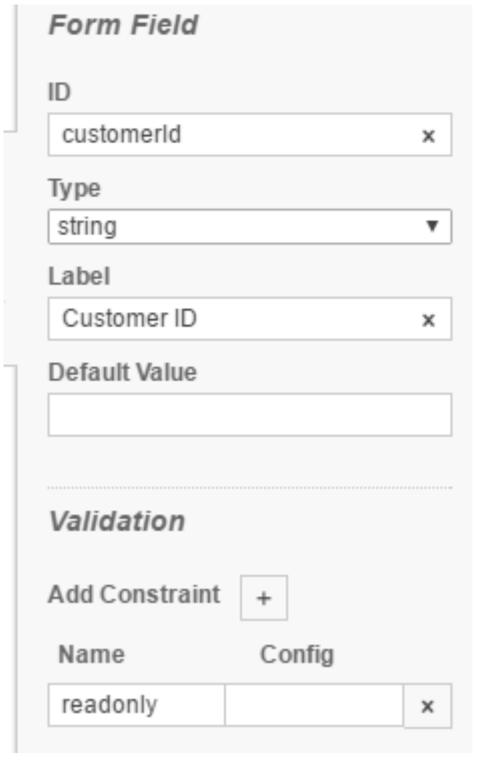
SOftEng http://softeng.polito.it

Process Test – User Task Forms

Create the same form for "Check document" and "Sign document".

For these two tasks, we will set the customerId fields in read-only mode.

In this way, John and Mary will not be able to change the value of customerId.



Form Field

ID: customerId

Type: string

Label: Customer ID

Default Value:

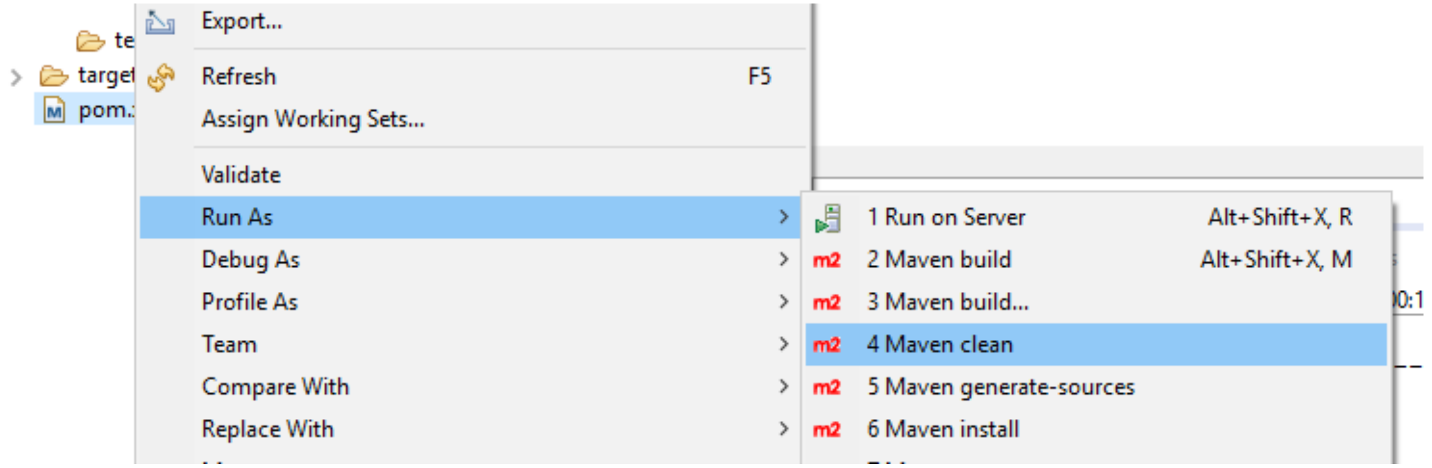
Validation

Add Constraint

Name	Config
readonly	

Process Test – User Task Forms

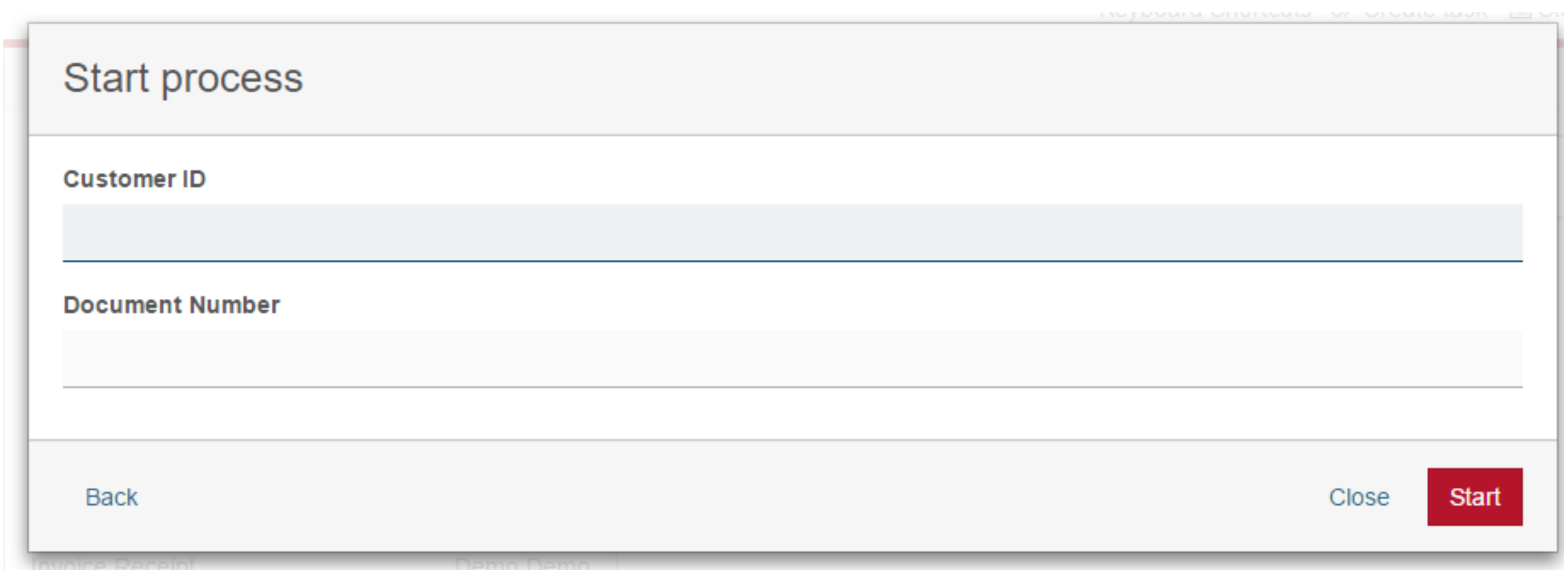
In Eclipse, right-click on pom.xml and perform a **Maven clean** followed by **Maven install.** Copy the new generated **.war** file in the folder $CAMUNDA_HOME/server/apache-tomcat.../webapps .



Possible building errors may be solved by right-clicking the project name, and selecting **Maven -> Update Project** (or by pressing Alt + F5).

Process Test – User Task Forms

Now, when launching from the administrator's Tasklist the Document Approval process, the form we defined will be prompted.



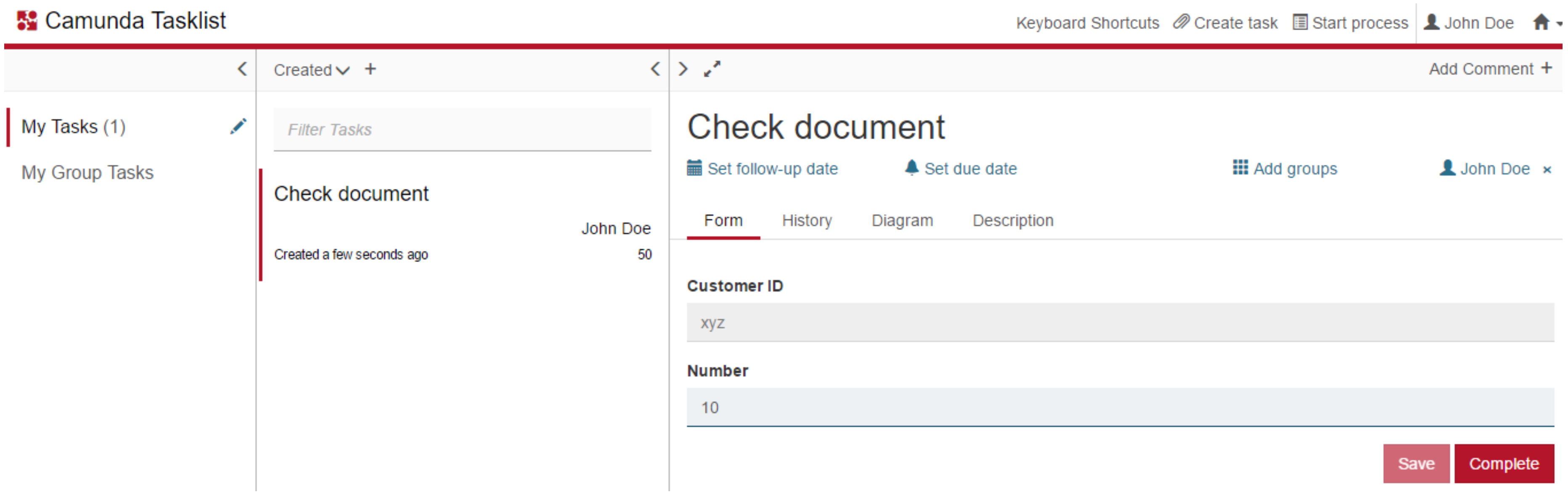
Start process

Customer ID

Document Number

Process Test – User Task Forms

In John’s and Mary’s tasklist we will see a different form, where the customerId is read-only and it cannot be edited.



Check document

John Doe

Created a few seconds ago 50

Check document

Customer ID

xyz

Number

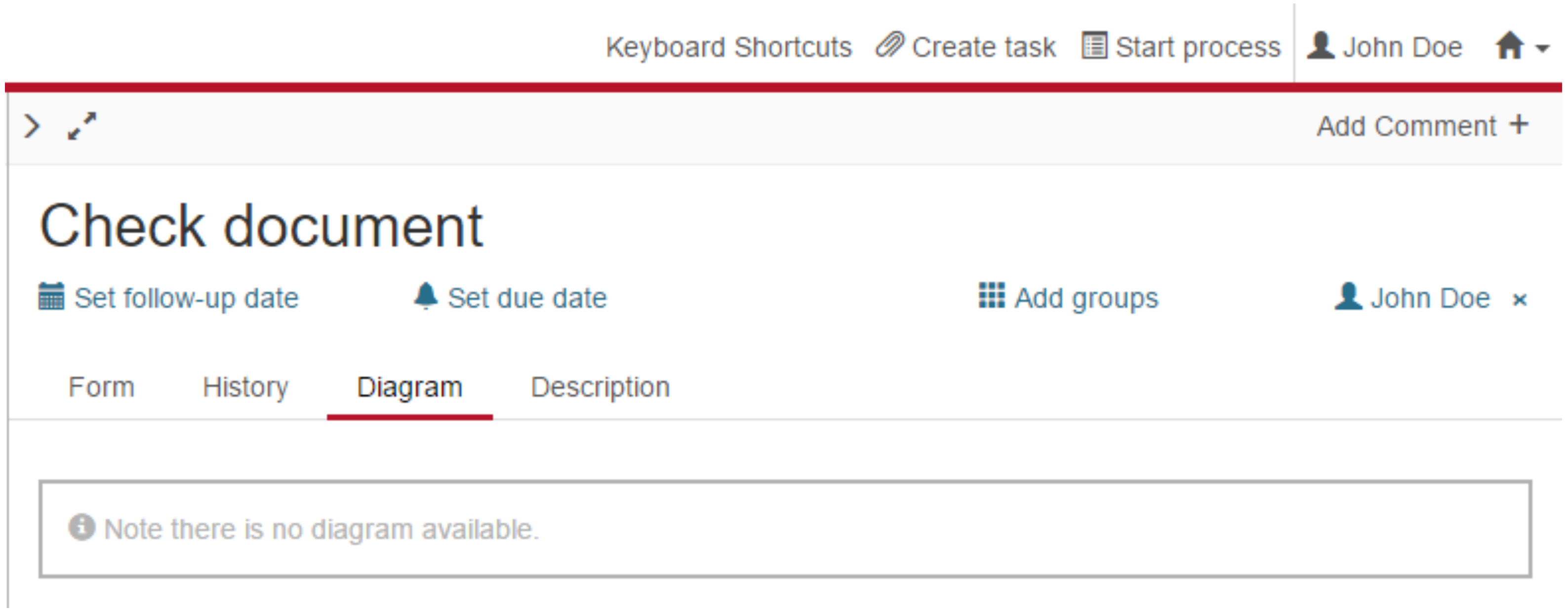
10

AUTHORIZATION MANAGEMENT

Authorization management

Let’s consider the process Document Approval we have already defined.

When John has to work on his task, we can notice that he has no possibility to look at the process diagram from his tasklist.

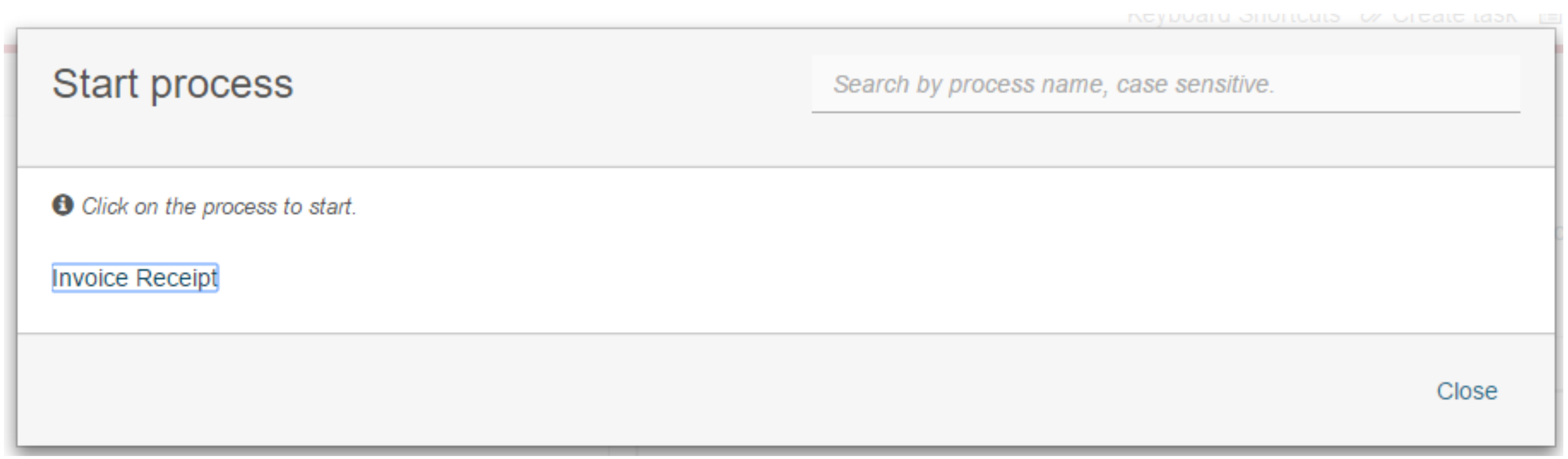


Check document

Note there is no diagram available.

Authorization management

Clicking on “Start Process” in John’s tasklist, we can see that he does not have the possibility to start an instance of the Document Approval process.


Start process

Search by process name, case sensitive.

Click on the process to start.

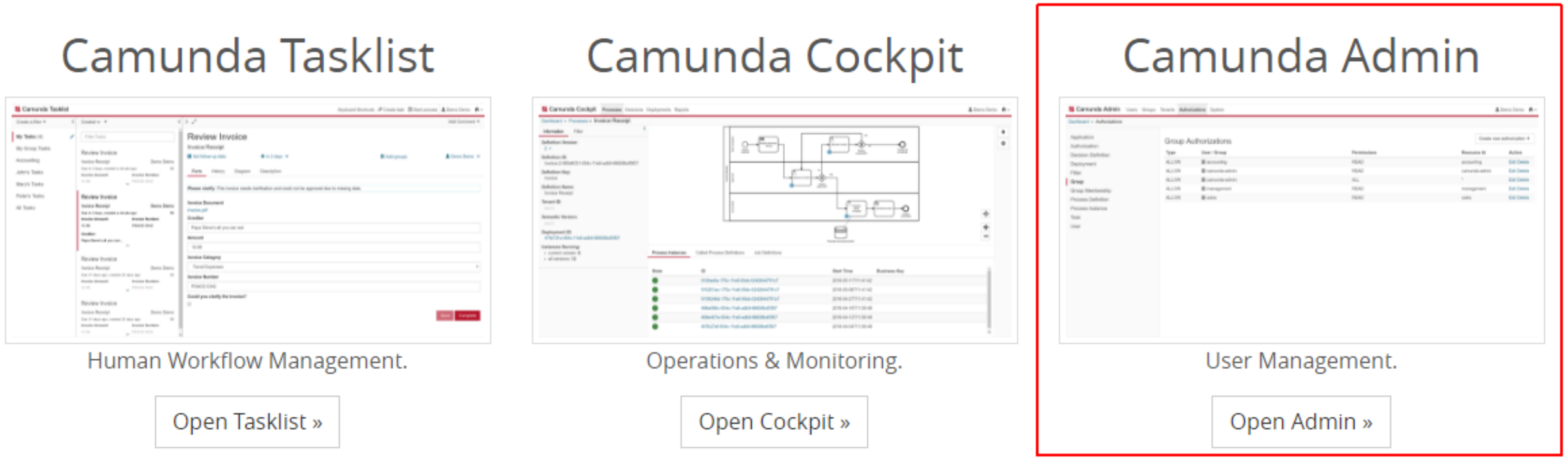
Invoice Receipt

Close

Authorization management

To make operations on processes possible for specific users of the team, we have to modify the permissions in our system.

Permissions are managed in the **Authorization** panel inside **Camunda Admin**.



Camunda Tasklist

Human Workflow Management.

Open Tasklist »

Camunda Cockpit

Operations & Monitoring.

Open Cockpit »

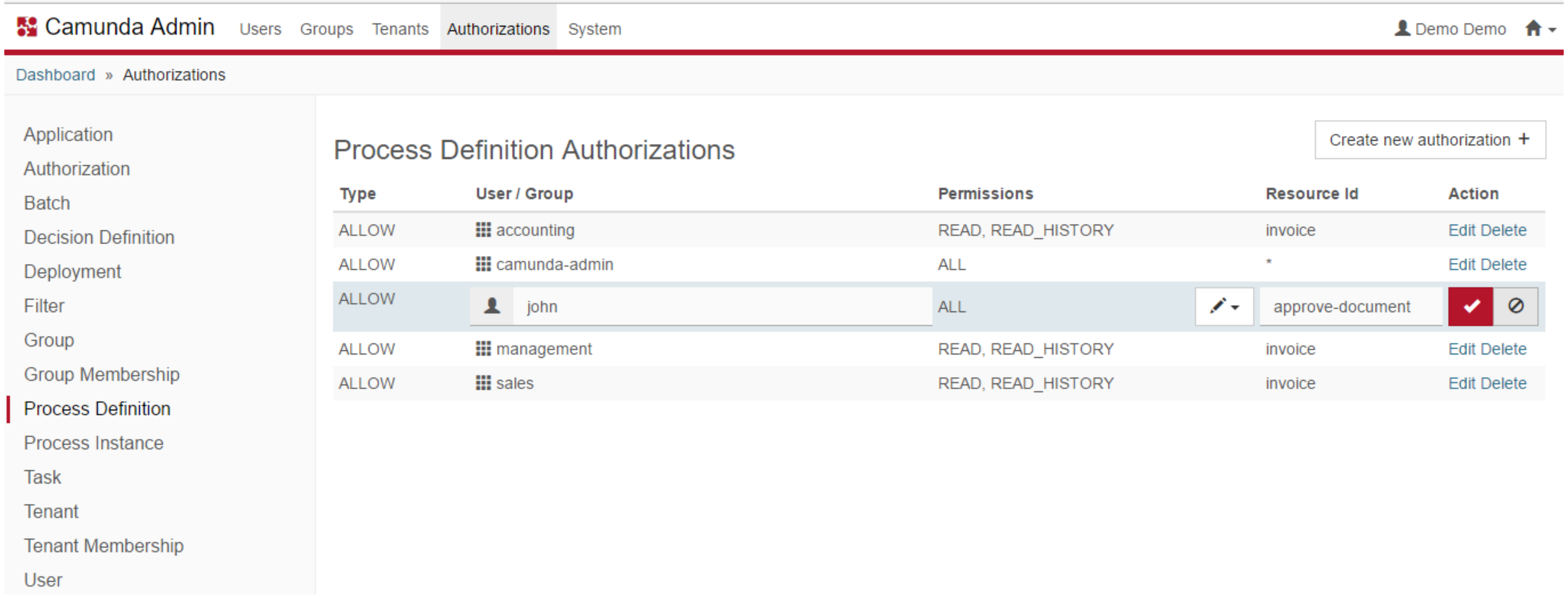
Camunda Admin

User Management.

Open Admin »

Authorization management

For instance, we may want to give John the permissions to track the execution of a Document Approval process instance. We can set this permission in the **Process Definition** tab.

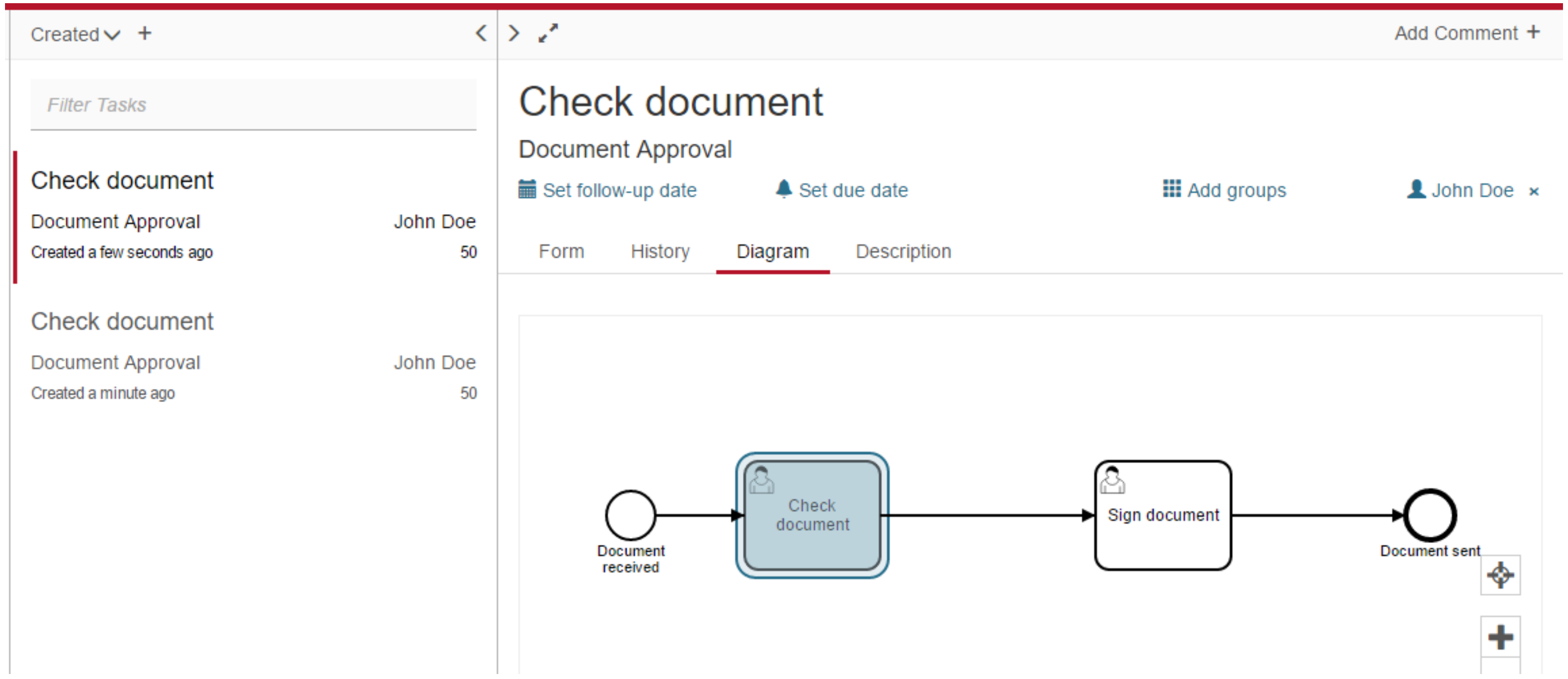


Process Definition Authorizations

Type	User / Group	Permissions	Resource Id	Action
ALLOW	accounting	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	camunda-admin	ALL	*	Edit Delete
ALLOW	john	ALL	approve-document	✓ ⊘
ALLOW	management	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	sales	READ, READ_HISTORY	invoice	Edit Delete

Authorization management

Now, when a new instance of the process is started, it is possible for John to see its current evolution.



Created +

Filter Tasks

Check document
Document Approval — John Doe
Created a few seconds ago — 50

Check document
Document Approval — John Doe
Created a minute ago — 50

Add Comment +

Check document

Document Approval

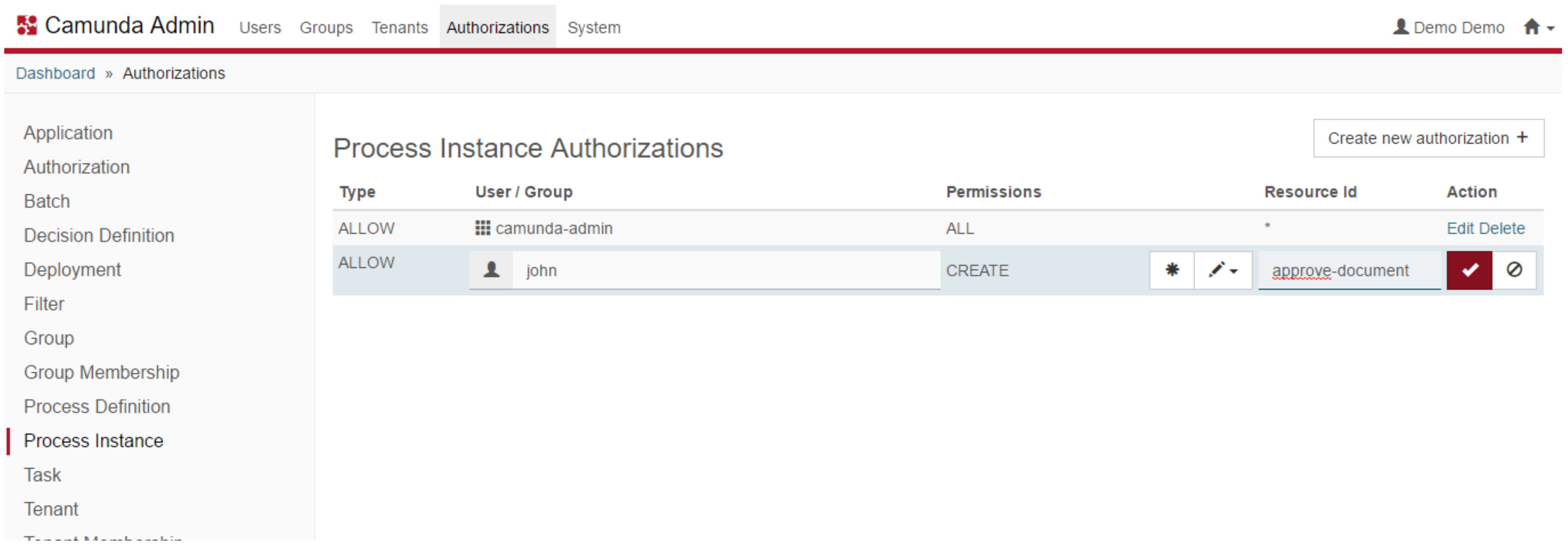
Set follow-up date · Set due date · Add groups · John Doe ×

Form · History · Diagram · Description

Authorization management

As another example, to allow John to start a new instance of the Document Approval process, we can set the following permission in the **Process Instance** tab.

Permissions can be also set for groups instead of individual users.



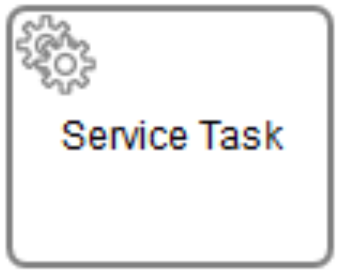
Process Instance Authorizations

Type	User / Group	Permissions	Resource Id	Action
ALLOW	camunda-admin	ALL	*	Edit Delete
ALLOW	john	CREATE	approve-document	

SERVICE TASKS

Service tasks

A **Service Task** is a task that can be used to invoke services.

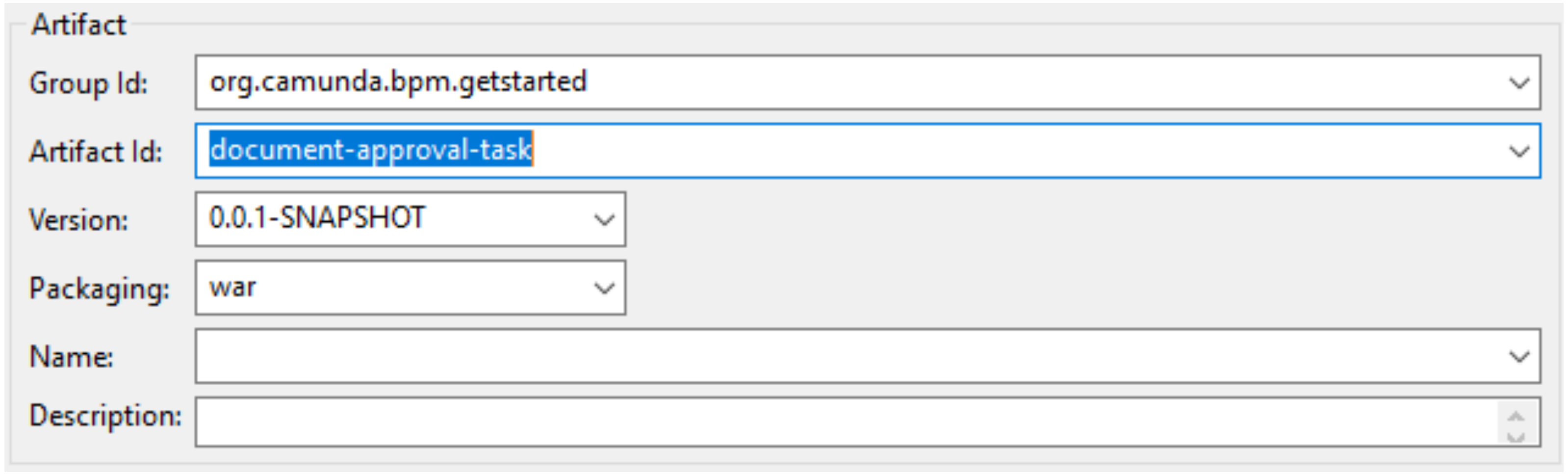


There are several ways to invoke external services using the Camunda platform.

In the following examples the possibility to invoke a **Delegate Java Class** will be used.

Service tasks

Start by creating a new Maven Project in Eclipse as in the Document Approval example. Call it document-approval-task.

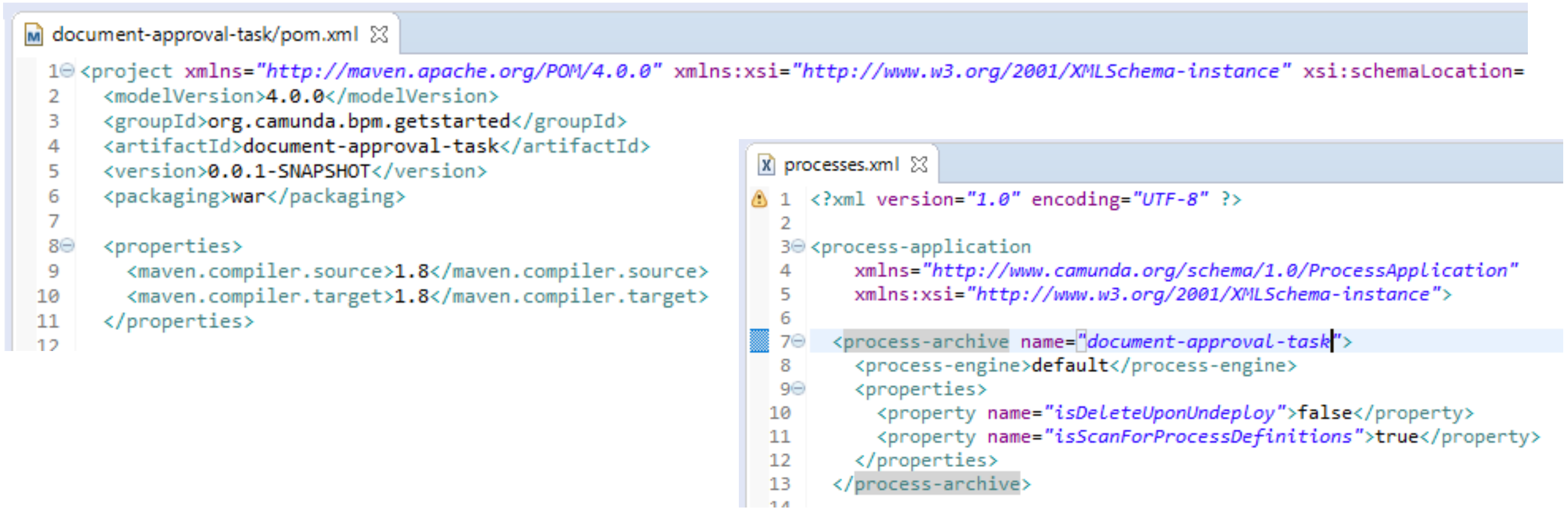


Artifact

Group Id:	org.camunda.bpm.getstarted
Artifact Id:	document-approval-task
Version:	0.0.1-SNAPSHOT
Packaging:	war
Name:	
Description:	

Service tasks

Configure the project by adding the pom.xml and processes.xml file with the Camunda dependencies and setup instructions. Modify the artifact ID in the pom.xml file, and the process-archive name in the processes.xml, so that they are compliant with the name of the project.


document-approval-task/pom.xml

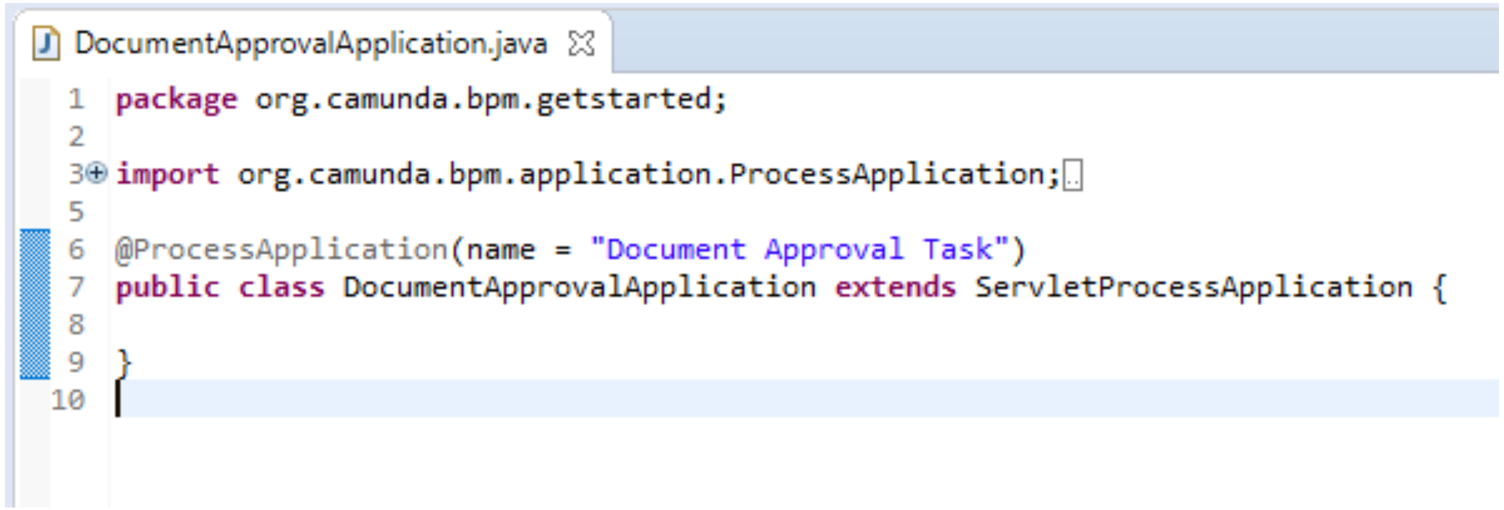
```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.camunda.bpm.getstarted</groupId>
  <artifactId>document-approval-task</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
```

processes.xml

```xml
<?xml version="1.0" encoding="UTF-8" ?>

<process-application
    xmlns="http://www.camunda.org/schema/1.0/ProcessApplication"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <process-archive name="document-approval-task">
    <process-engine>default</process-engine>
    <properties>
      <property name="isDeleteUponUndeploy">false</property>
      <property name="isScanForProcessDefinitions">true</property>
    </properties>
  </process-archive>
```

Service tasks

Create the package org.camunda.bpm.getstarted and the class DocumentApprovalApplication inside it. Make it a child of ServletProcessApplication and leave it with an empty implementation.



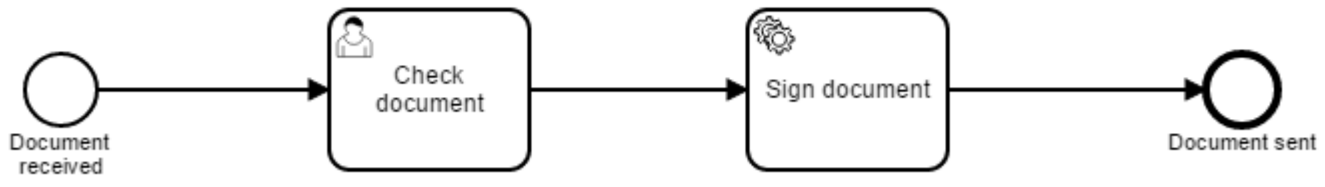
DocumentApprovalApplication.java

```java
package org.camunda.bpm.getstarted;

import org.camunda.bpm.application.ProcessApplication;

@ProcessApplication(name = "Document Approval Task")
public class DocumentApprovalApplication extends ServletProcessApplication {

}
```

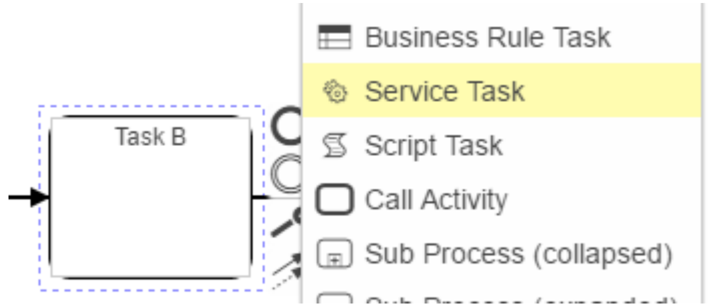
Now the project is configured for the deployment on the Camunda engine.

Service tasks

In **Camunda Modeler**, model a sample process model with a service task. Assign the User Task to john.



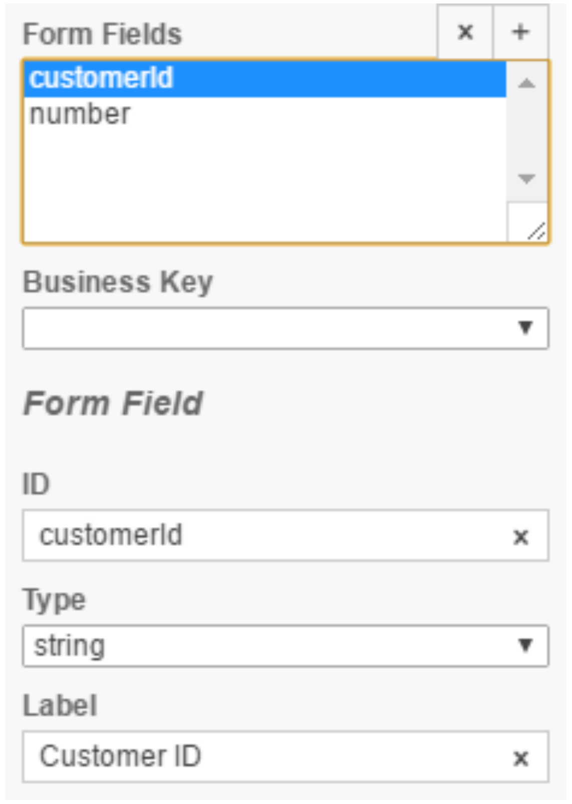
The type of the task can be assigned by clicking on the wrench icon after selecting it.



Service tasks

We assign a form to the Start Event and to the User Task, like in the Document Approval application.

The form field ID is the way we can refer to the instance variable in the Camunda Platform, for the entire execution of the process.


Form Fields

customerId
number

Business Key

Form Field

ID

customerId

Type

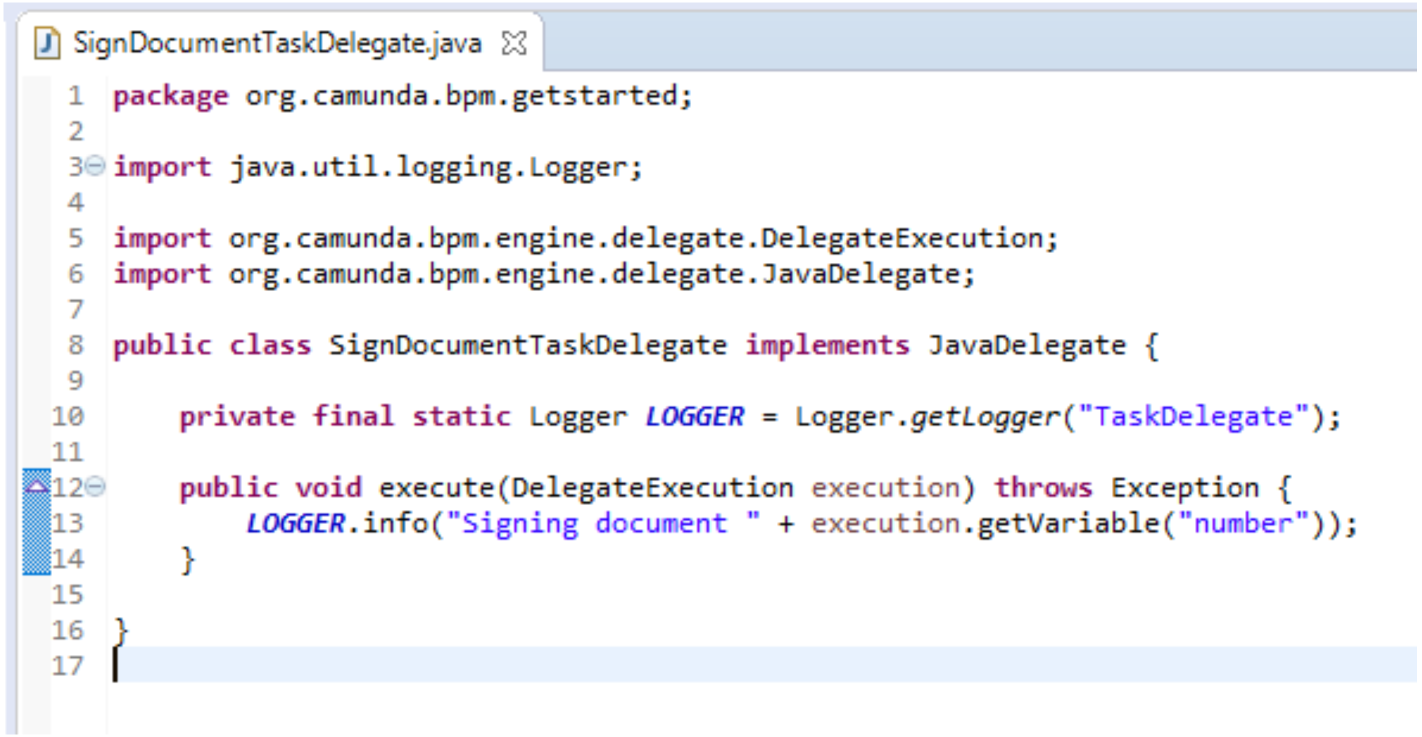
string

Label

Customer ID

Service tasks

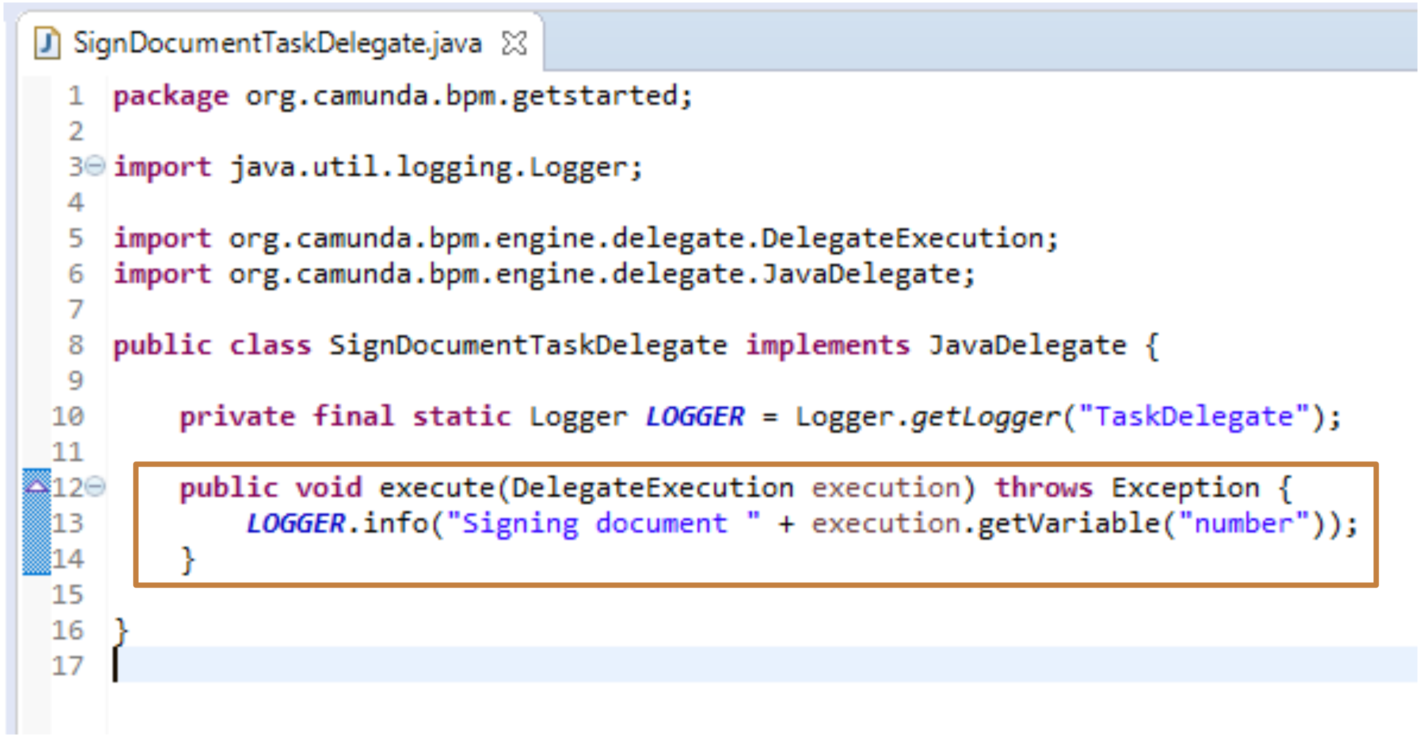
To make the Service Task executable, a class implementing JavaDelegate is needed. Create a new class that implements JavaDelegate in the package com.camunda.bpm.getstarted and name it SignDocumentTaskDelegate.


SignDocumentTaskDelegate.java

```java
package org.camunda.bpm.getstarted;

import java.util.logging.Logger;

import org.camunda.bpm.engine.delegate.DelegateExecution;
import org.camunda.bpm.engine.delegate.JavaDelegate;

public class SignDocumentTaskDelegate implements JavaDelegate {

    private final static Logger LOGGER = Logger.getLogger("TaskDelegate");

    public void execute(DelegateExecution execution) throws Exception {
        LOGGER.info("Signing document " + execution.getVariable("number"));
    }

}
```

Service tasks

The **execution** variable represents the process instance being executed. The variables declared during the execution of the instance can be accessed by means of the execution.getVariable(“variable name”) method.

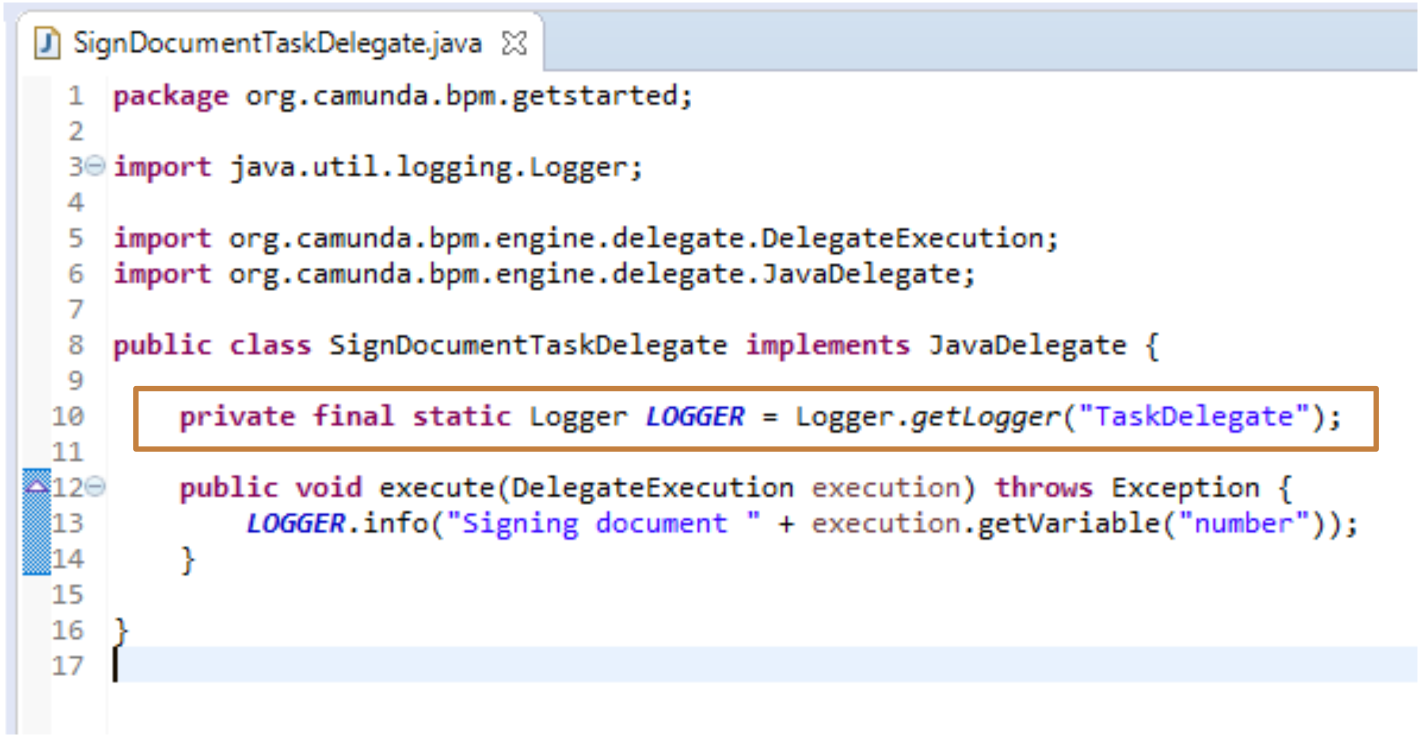


SignDocumentTaskDelegate.java

```java
package org.camunda.bpm.getstarted;

import java.util.logging.Logger;

import org.camunda.bpm.engine.delegate.DelegateExecution;
import org.camunda.bpm.engine.delegate.JavaDelegate;

public class SignDocumentTaskDelegate implements JavaDelegate {

    private final static Logger LOGGER = Logger.getLogger("TaskDelegate");

    public void execute(DelegateExecution execution) throws Exception {
        LOGGER.info("Signing document " + execution.getVariable("number"));
    }

}
```

Service tasks

To give a simple example of a process to be automatically executed, we simply use the Logger of the Tomcat server on which the Camunda engine runs, to show the value we assigned to the variable “number”.

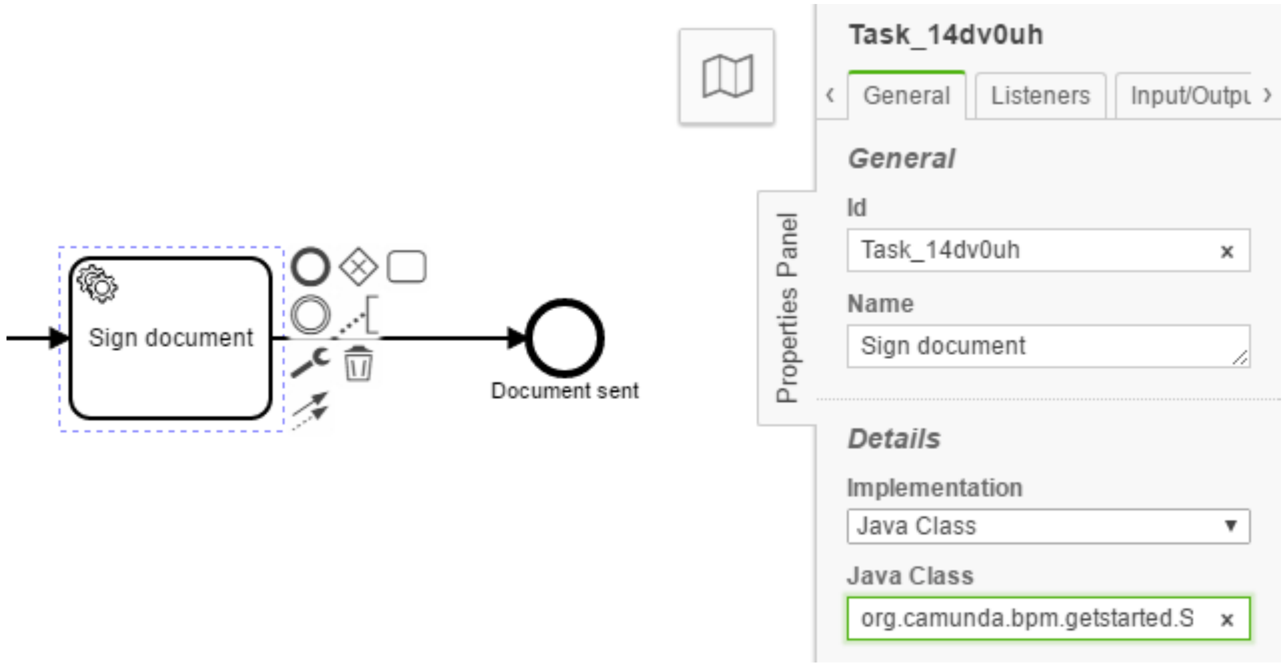


```java
package org.camunda.bpm.getstarted;

import java.util.logging.Logger;

import org.camunda.bpm.engine.delegate.DelegateExecution;
import org.camunda.bpm.engine.delegate.JavaDelegate;

public class SignDocumentTaskDelegate implements JavaDelegate {

    private final static Logger LOGGER = Logger.getLogger("TaskDelegate");

    public void execute(DelegateExecution execution) throws Exception {
        LOGGER.info("Signing document " + execution.getVariable("number"));
    }


}
```

Service tasks

The delegate just created must be assigned to the Service Task from which it will be called. This can be done directly from the modeler.

Select **Java Class** in the **Implementation** field of the Service task, and type the name of the class in the Java Class field.



Task_14dv0uh

General

Id: Task_14dv0uh

Name: Sign document

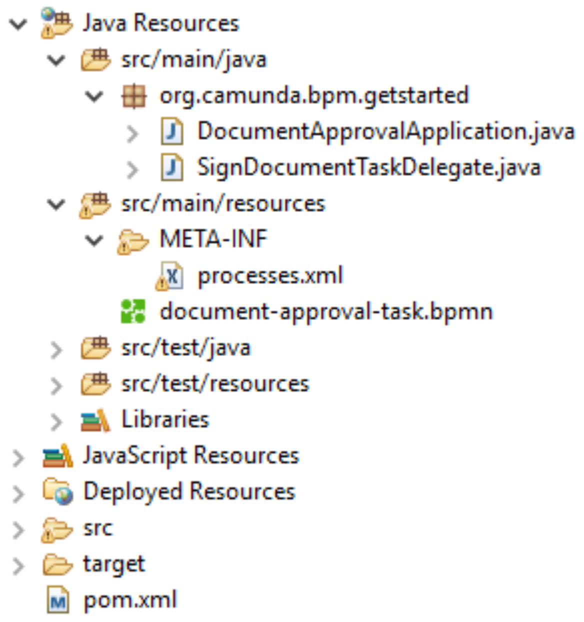
Details

Implementation: Java Class

Java Class: org.camunda.bpm.getstarted.S

Service tasks

Save the bpmn model in the src/main/resources folder.

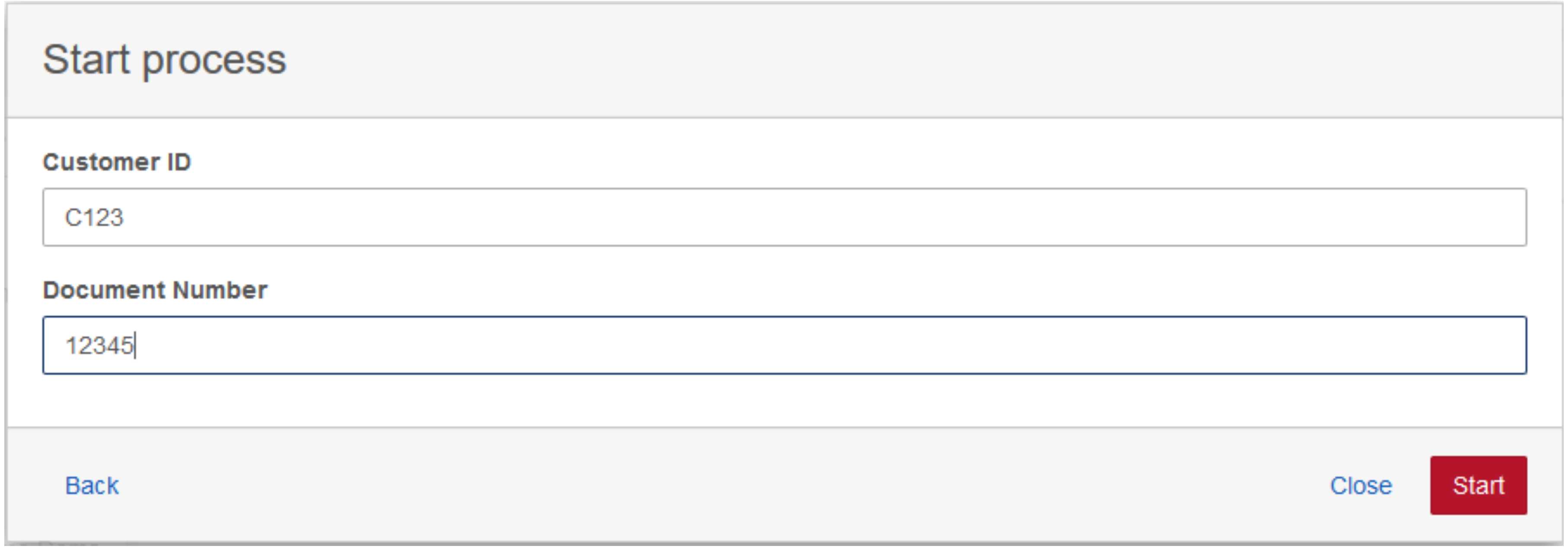


```
Java Resources
  src/main/java
    org.camunda.bpm.getstarted
      DocumentApprovalApplication.java
      SignDocumentTaskDelegate.java
  src/main/resources
    META-INF
      processes.xml
    document-approval-task.bpmn
  src/test/java
  src/test/resources
  Libraries
JavaScript Resources
Deployed Resources
src
target
pom.xml
```

Perform a Maven build, and copy the resulting .war file in the webapps folder of the Camunda Tomcat distribution.

Service tasks

Now that the process is deployed, it can be launched from the Admin Tasklist. Assign any values to the fields.



Start process

Customer ID

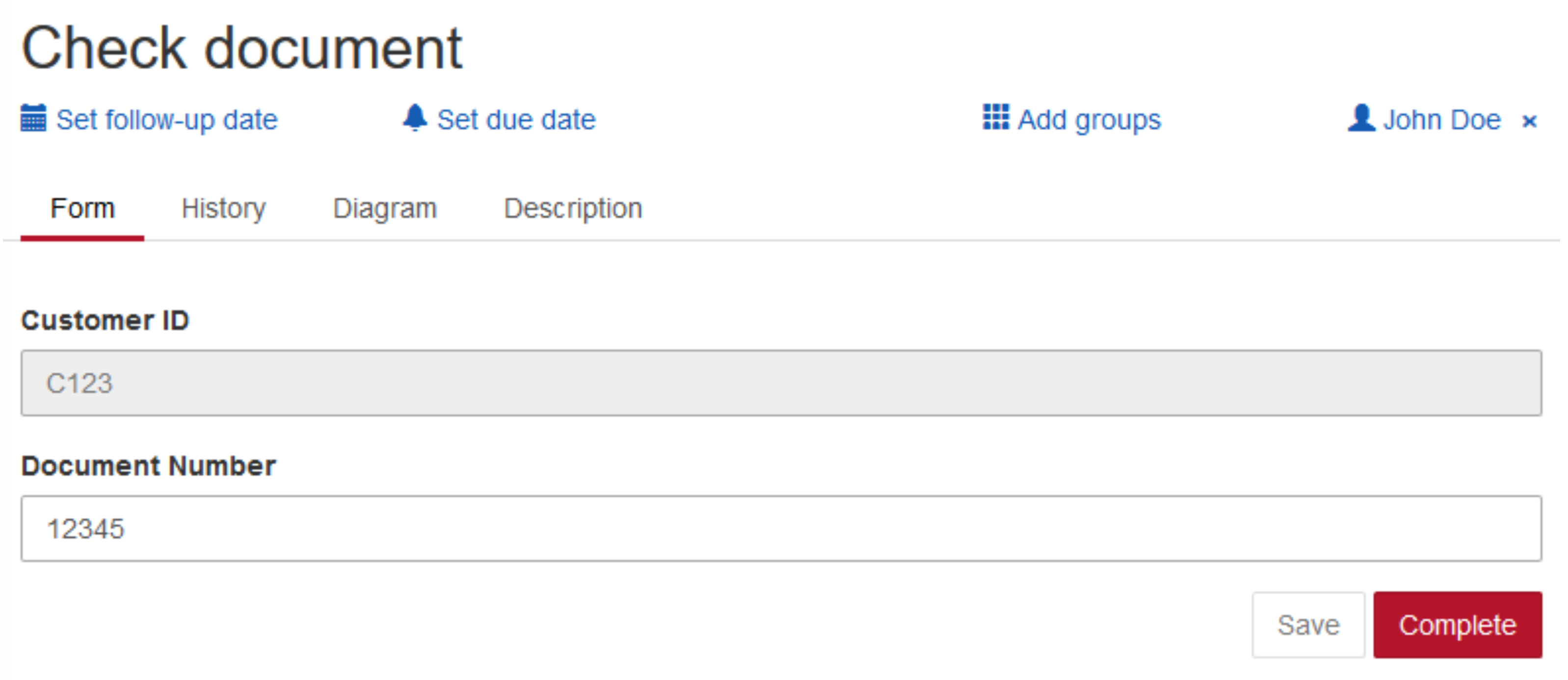
C123

Document Number

12345

Service tasks

Go to John's tasklist (usr/pwd: john/john), select Check document from the tasklist and complete the task.

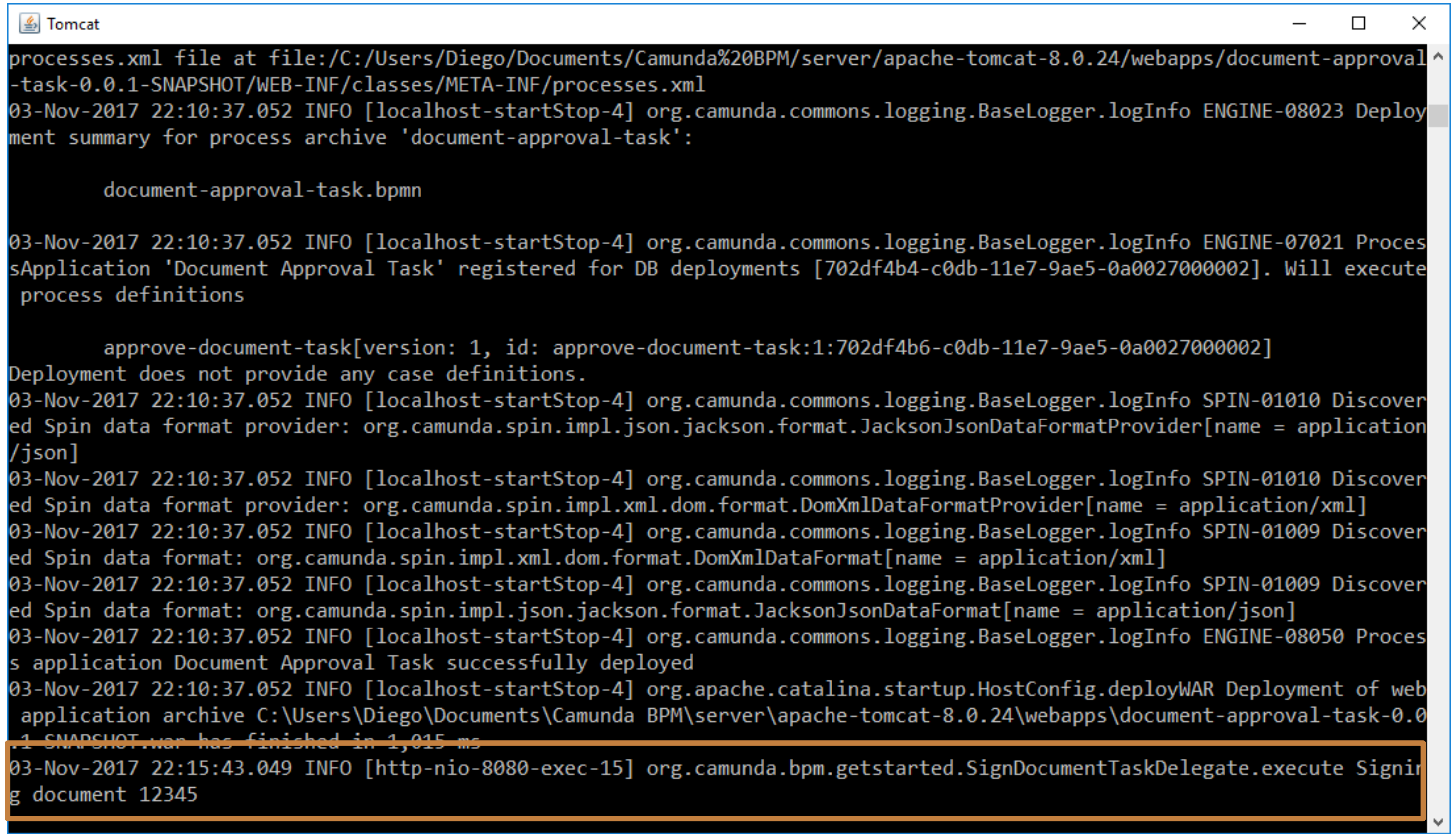

Check document

Customer ID: C123

Document Number: 12345

Service tasks

After the execution of John’s task, a line is added to the Tomcat logger. It means that the routine assigned to the Service Task has been performed.



```
processes.xml file at file:/C:/Users/Diego/Documents/Camunda%20BPM/server/apache-tomcat-8.0.24/webapps/document-approval
-task-0.0.1-SNAPSHOT/WEB-INF/classes/META-INF/processes.xml
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo ENGINE-08023 Deploy
ment summary for process archive 'document-approval-task':

        document-approval-task.bpmn

03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo ENGINE-07021 Proces
sApplication 'Document Approval Task' registered for DB deployments [702df4b4-c0db-11e7-9ae5-0a0027000002]. Will execute
 process definitions

        approve-document-task[version: 1, id: approve-document-task:1:702df4b6-c0db-11e7-9ae5-0a0027000002]
Deployment does not provide any case definitions.
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01010 Discover
ed Spin data format provider: org.camunda.spin.impl.json.jackson.format.JacksonJsonDataFormatProvider[name = application
/json]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01010 Discover
ed Spin data format provider: org.camunda.spin.impl.xml.dom.format.DomXmlDataFormatProvider[name = application/xml]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01009 Discover
ed Spin data format: org.camunda.spin.impl.xml.dom.format.DomXmlDataFormat[name = application/xml]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01009 Discover
ed Spin data format: org.camunda.spin.impl.json.jackson.format.JacksonJsonDataFormat[name = application/json]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo ENGINE-08050 Proces
s application Document Approval Task successfully deployed
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.apache.catalina.startup.HostConfig.deployWAR Deployment of web
 application archive C:\Users\Diego\Documents\Camunda BPM\server\apache-tomcat-8.0.24\webapps\document-approval-task-0.0
.1-SNAPSHOT.war has finished in 1,015 ms
03-Nov-2017 22:15:43.049 INFO [http-nio-8080-exec-15] org.camunda.bpm.getstarted.SignDocumentTaskDelegate.execute Signin
g document 12345
```

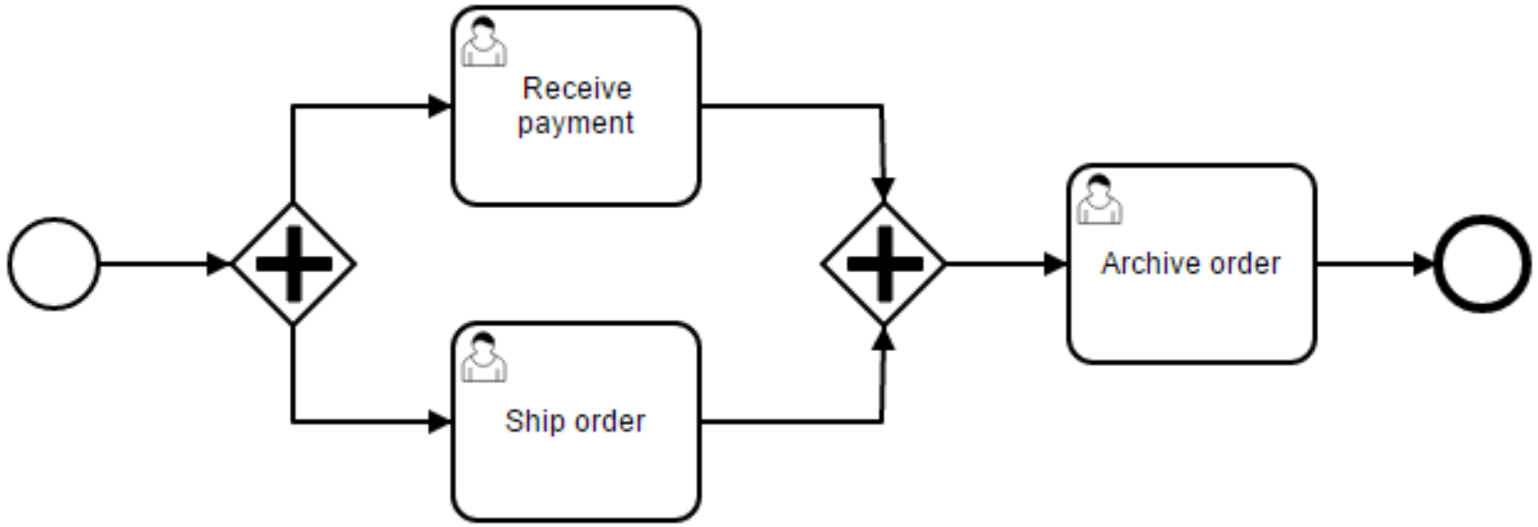
SoftEng http://softeng.polito.it

GATEWAYS

SOftEng
http://softeng.polito.it

Parallel Gateways

Parallel Gateways allow to fork and join multiple path of execution, thus permitting the execution of different tasks concurrently.

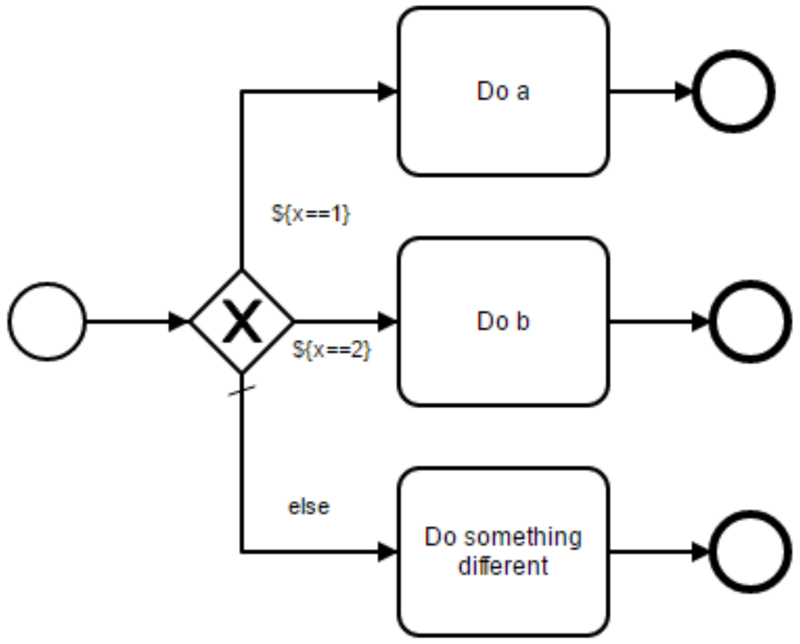


Exclusive Gateways

When the flow arrives to an **Exclusive Gateways**, only one of the subsequent sequence flows is taken.

Each sequence flow has to be paired with a **condition**. Sequent flow are evaluated in the order in which they are defined (the order they are written in the .bpmn file, if read as a regular .xml file).

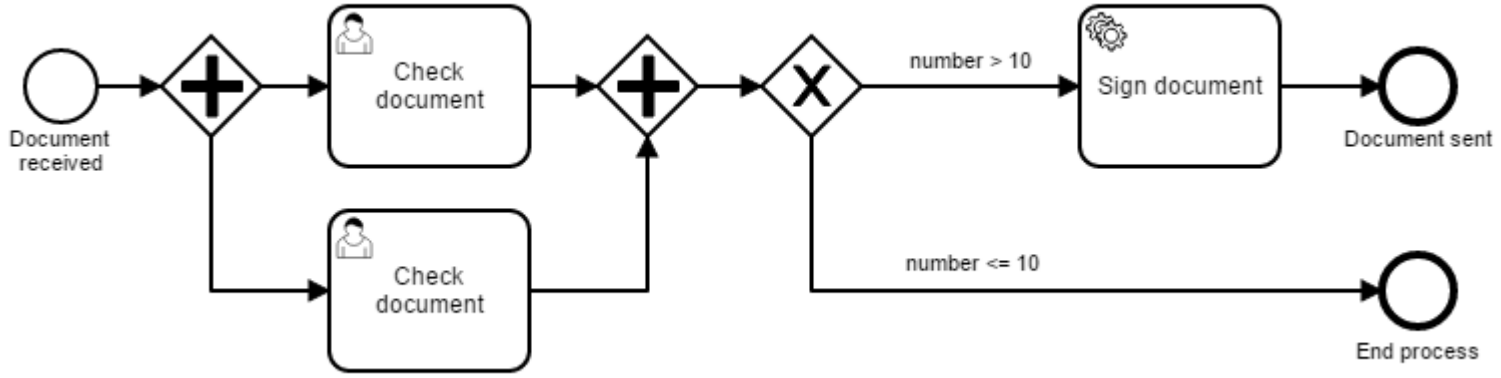
The first sequence flow with a "true" condition (or the "default" one with no condition) is selected.



Sample process with Gateways

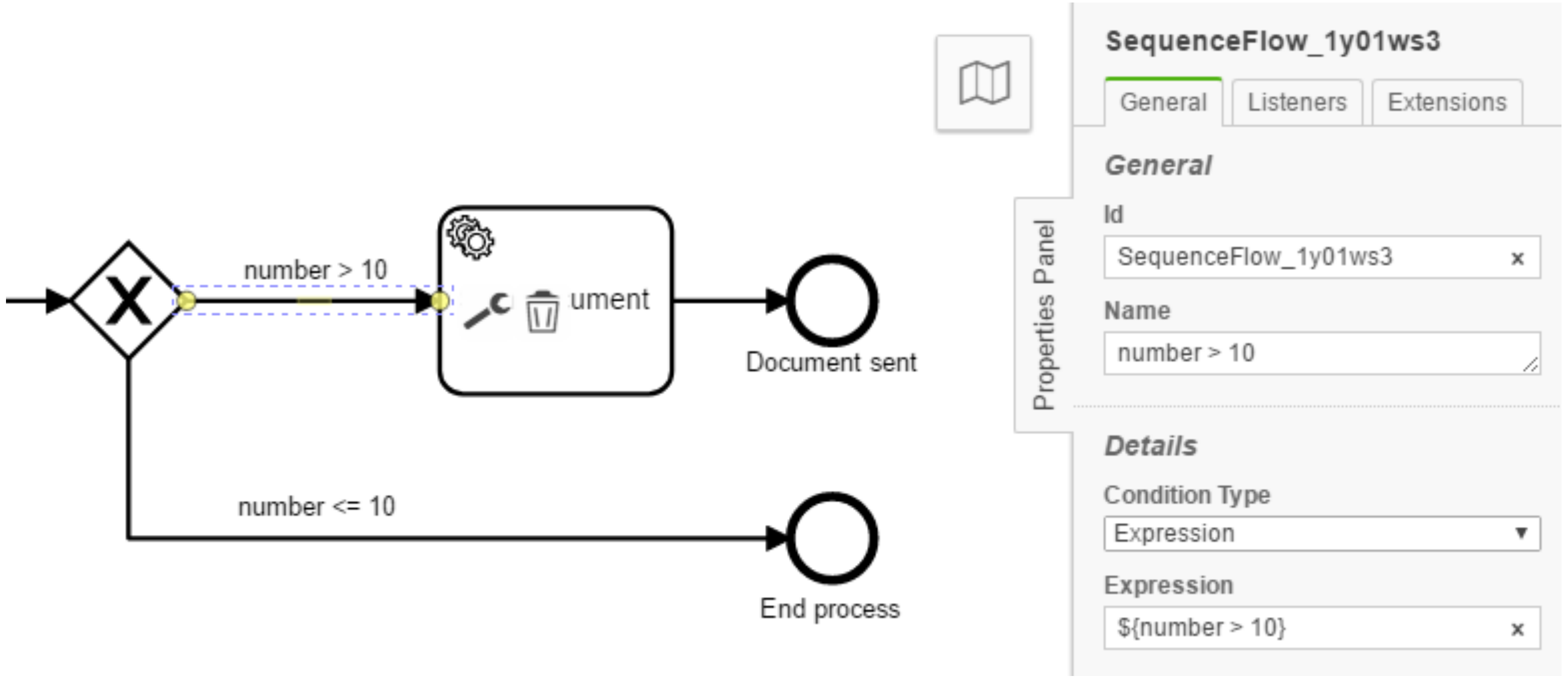
In our previous project, we want to add a second Check document task to be executed in parallel with the existing task. We will assign it to Mary.

We also want to have some output on the logger only if the number is bigger than 10.



Sample process with Gateways

A condition must be associated with the sequence flows using the Properties panel. We assign a condition on the variable “number”, defined in the form previously assigned to the initial task.



SequenceFlow_1y01ws3

General | Listeners | Extensions

General

Id: SequenceFlow_1y01ws3

Name: number > 10

Details

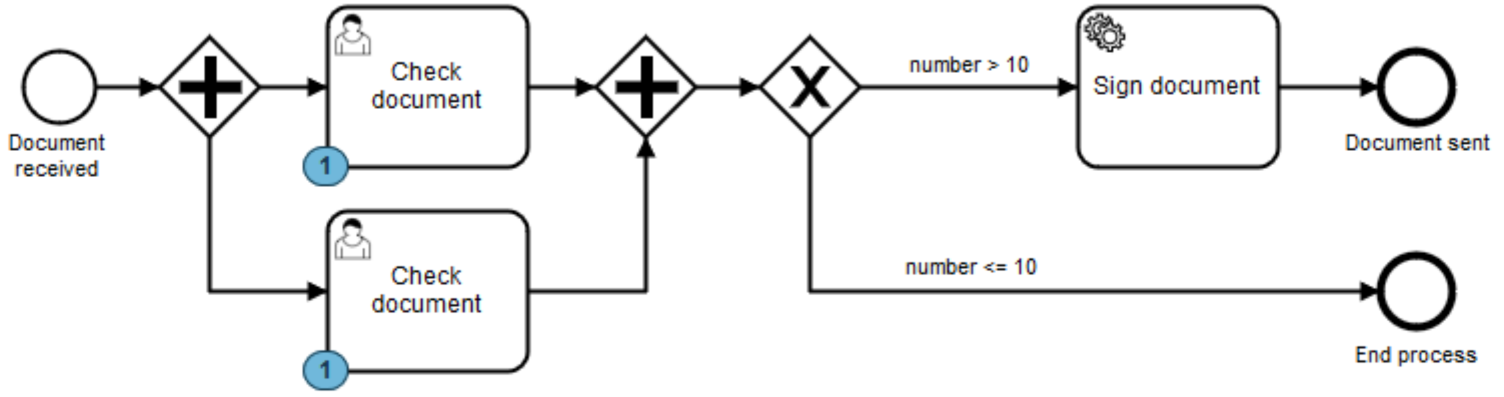
Condition Type: Expression

Expression: ${number > 10}

Sample process with Gateways

We launch an execution with number = 50.

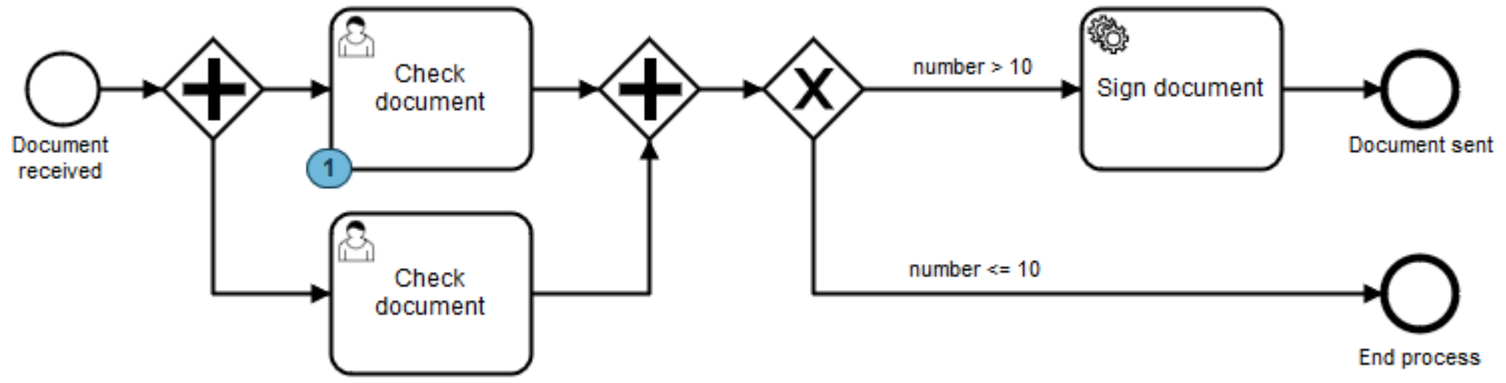
From the Cockpit, we can see that we have two different tasks awaiting for completion.



Sample process with Gateways

We can execute a single task (for instance, Mary’s one).

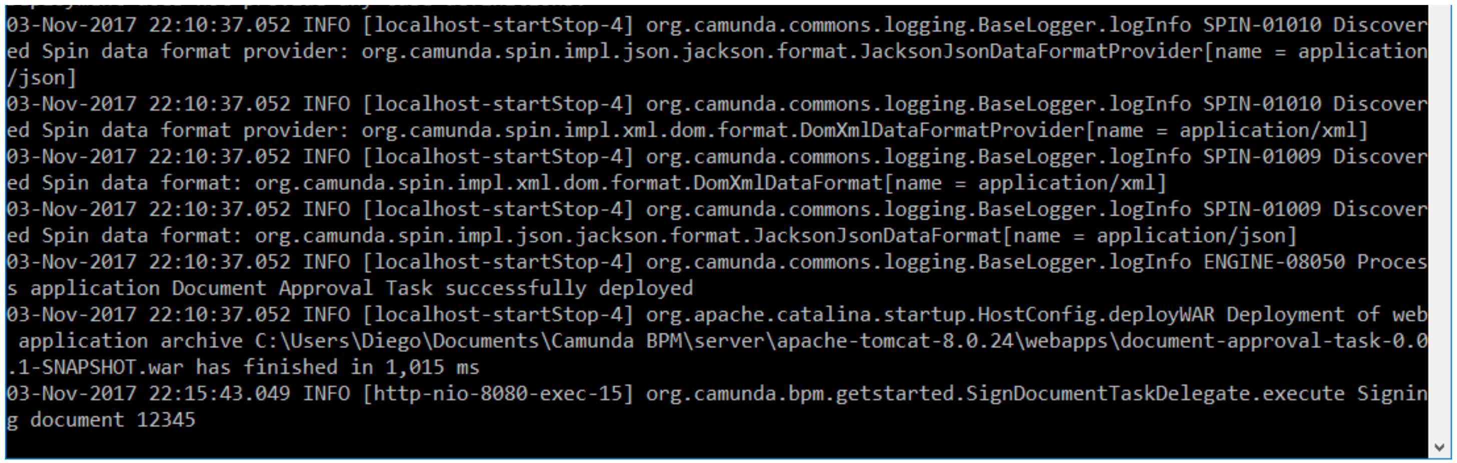
From the Cockpit, we notice that the completion of the second Check document does not make the process advance: the parallel join needs both tasks to be completed before allowing the process to go on.



Sample process with Gateways

If we complete the first task from John’s tasklist, we can see that, as in the previous example, a new line is printed in the Tomcat Logger.

On the other hand, if we launch a new process instance with number = 5, we can see that no lines are printed in the logger. This means that the process has taken the other sequence flow that goes straight to the process end, without performing any Service Task.



```
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01010 Discover
ed Spin data format provider: org.camunda.spin.impl.json.jackson.format.JacksonJsonDataFormatProvider[name = application
/json]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01010 Discover
ed Spin data format provider: org.camunda.spin.impl.xml.dom.format.DomXmlDataFormatProvider[name = application/xml]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01009 Discover
ed Spin data format: org.camunda.spin.impl.xml.dom.format.DomXmlDataFormat[name = application/xml]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo SPIN-01009 Discover
ed Spin data format: org.camunda.spin.impl.json.jackson.format.JacksonJsonDataFormat[name = application/json]
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.camunda.commons.logging.BaseLogger.logInfo ENGINE-08050 Proces
s application Document Approval Task successfully deployed
03-Nov-2017 22:10:37.052 INFO [localhost-startStop-4] org.apache.catalina.startup.HostConfig.deployWAR Deployment of web
 application archive C:\Users\Diego\Documents\Camunda BPM\server\apache-tomcat-8.0.24\webapps\document-approval-task-0.0
.1-SNAPSHOT.war has finished in 1,015 ms
03-Nov-2017 22:15:43.049 INFO [http-nio-8080-exec-15] org.camunda.bpm.getstarted.SignDocumentTaskDelegate.execute Signin
g document 12345
```

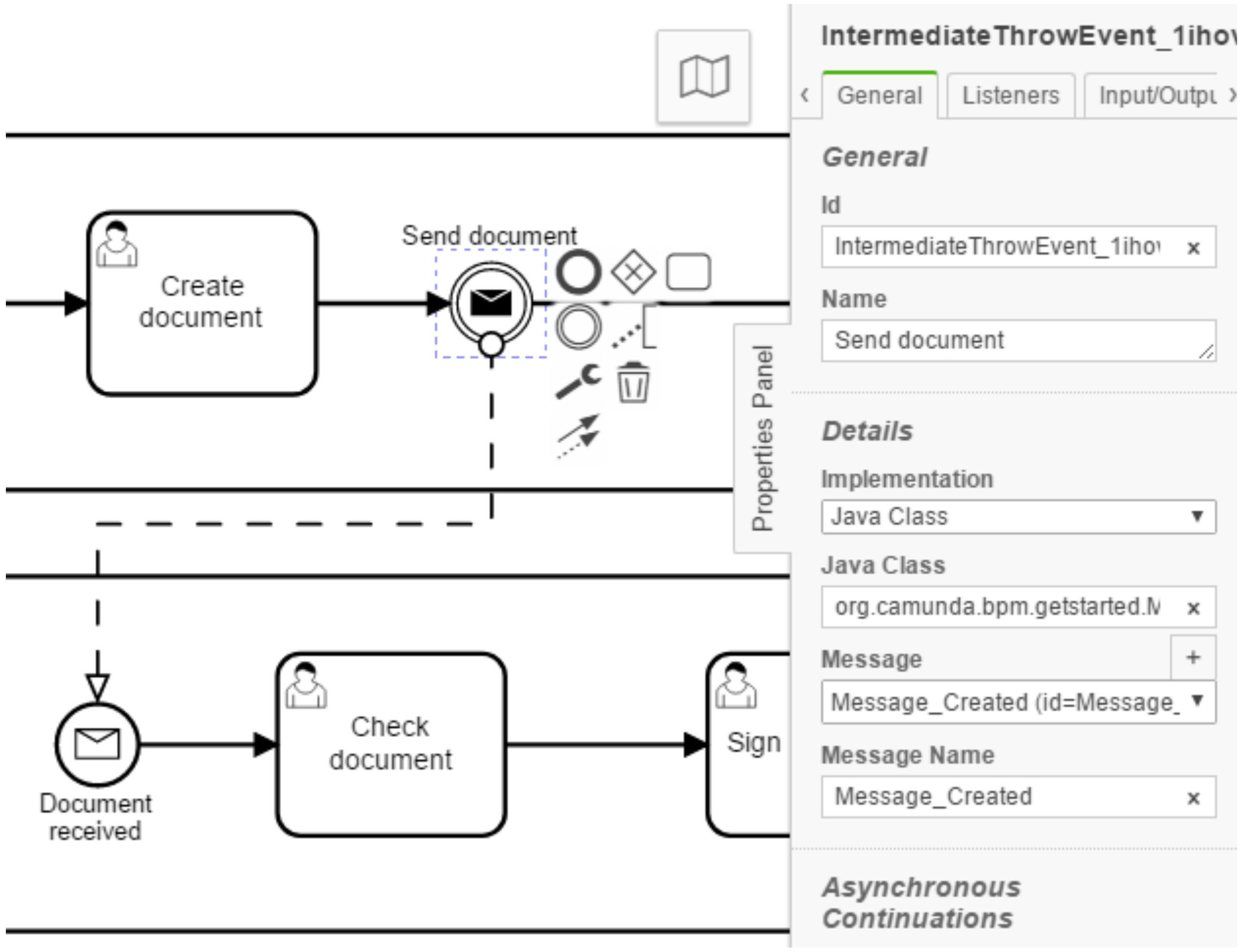
MESSAGE EVENTS

Message Events

Message events are events which reference a named message. A message has a name and a payload. Unlike a signal, a message event is always directed at a single recipient.

1. Create a Message Throw and Message Catch event in two different pools;
2. Assign to them the same Message Name;
3. Create a Java class that implements JavaDelegate;
4. Insert the fully qualified class name in the Message Throw event Java Class property.

Message Events



IntermediateThrowEvent_1ihov

General | Listeners | Input/Outpu

General

Id

IntermediateThrowEvent_1ihov

Name

Send document

Details

Implementation

Java Class

Java Class

org.camunda.bpm.getstarted.M

Message

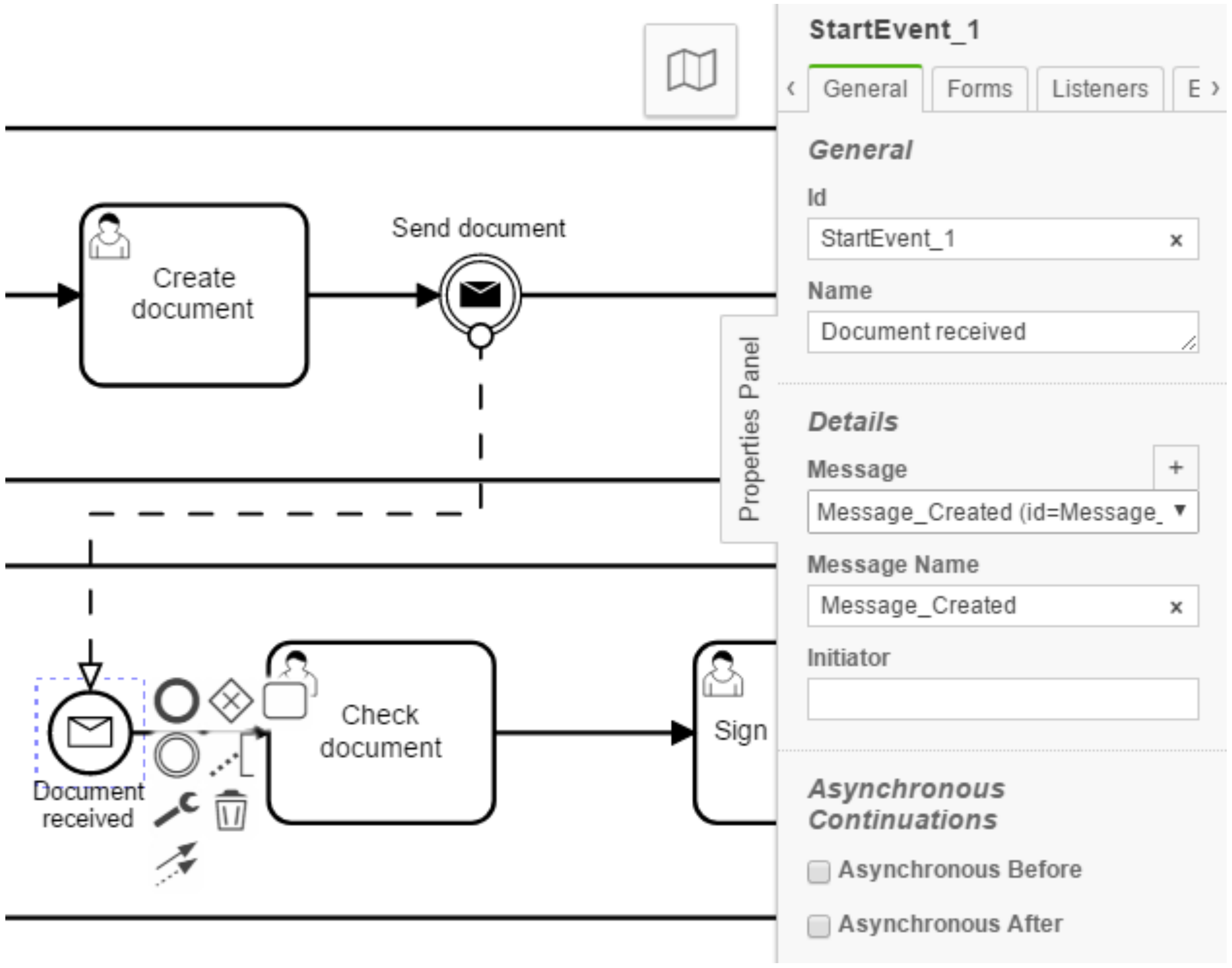
Message_Created (id=Message_

Message Name

Message_Created

Asynchronous Continuations

Message Events



StartEvent_1

General | Forms | Listeners | E

General

Id: StartEvent_1

Name: Document received

Details

Message: Message_Created (id=Message_

Message Name: Message_Created

Initiator:

Asynchronous Continuations

Asynchronous Before

Asynchronous After

Message Events

```
public class MessageCreated implements JavaDelegate {
   public void execute(DelegateExecution execution) throws
   Exception {
       RuntimeService runtimeService =
   execution.getProcessEngineServices().getRuntimeService(
   );

   runtimeService.createMessageCorrelation("Message_Create
   d")
                    .setVariable("topic",
   execution.getVariable("topic"))
                    .correlate();
   }
}
```