



# Camunda BPM for Developers

Getting started with the Camunda BPM Platform

1



2

## Agenda

### Day 1

1. Process Modelling with BPMN 2.0
2. Camunda BPM Platform
3. Client-API and Delegation
4. Data Objects, Gateways & Expressions

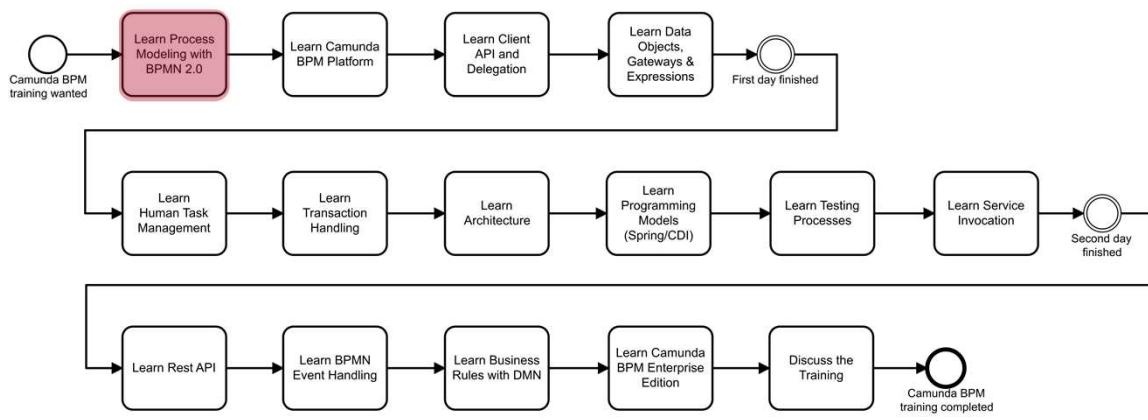
### Day 2

5. Human Task Management
6. Transaction Handling
7. Architecture
8. Programming Models (Spring/CDI)
9. Testing Processes
10. Service Invocation

### Day 3

11. REST API
12. BPMN Event Handling
13. Business Rules with DMN
14. Camunda BPM Enterprise Edition
15. Wrap up: Outlook, Feedback and Open Discussion

2



3

4

## Business Process Model and Notation (BPMN)

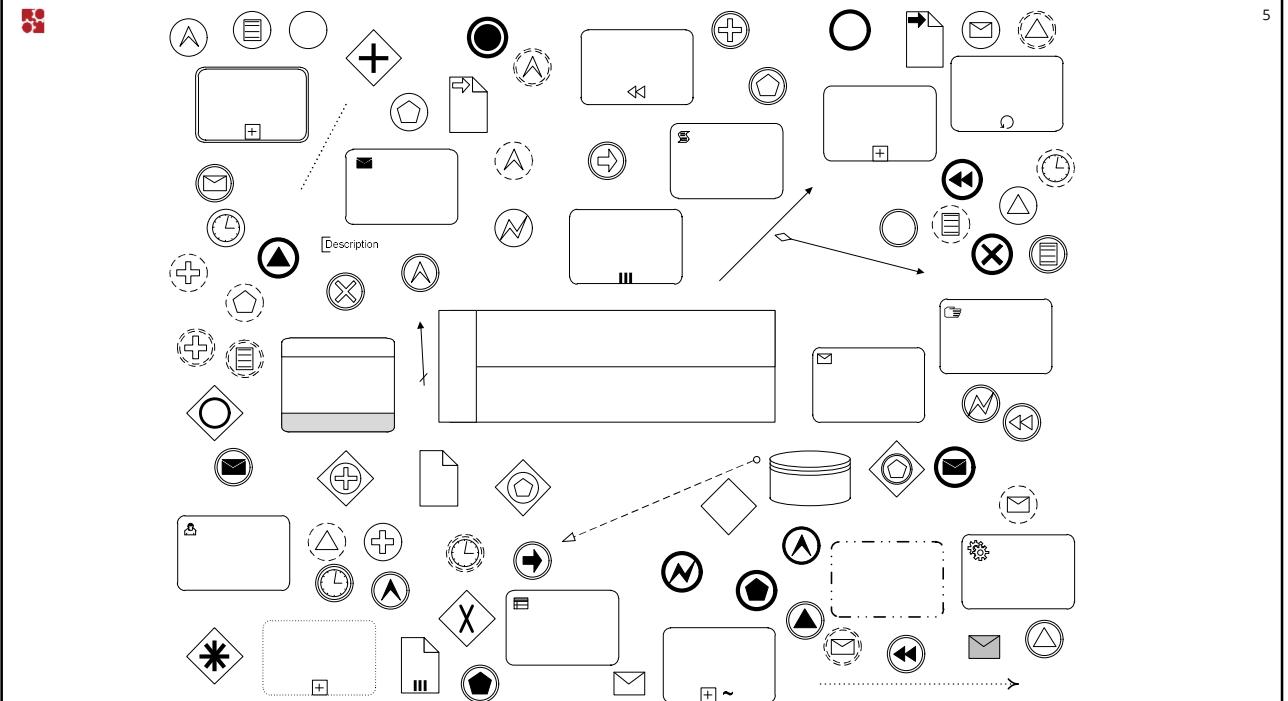
- BPMN is a worldwide OMG standard
- Latest version is BPMN 2.0
- Lots of companies joined the standardization committee
- All major BPMS vendors go for BPMN



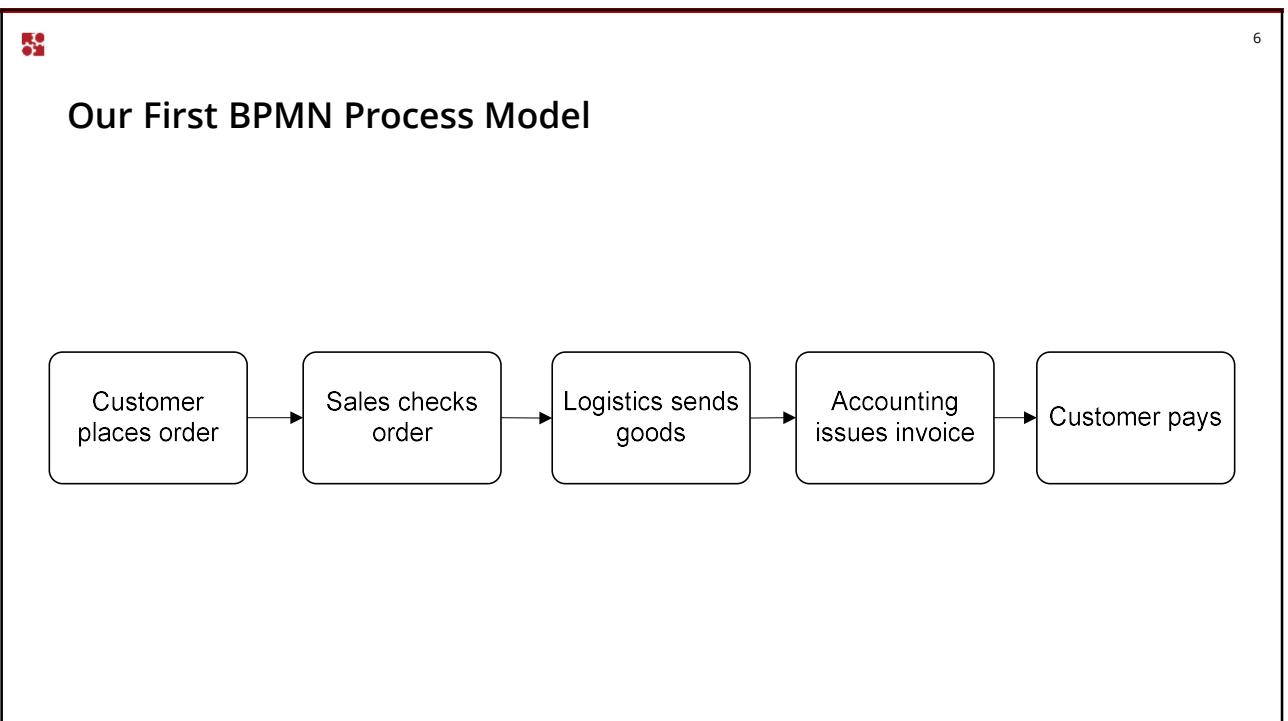
OBJECT MANAGEMENT GROUP



4



5

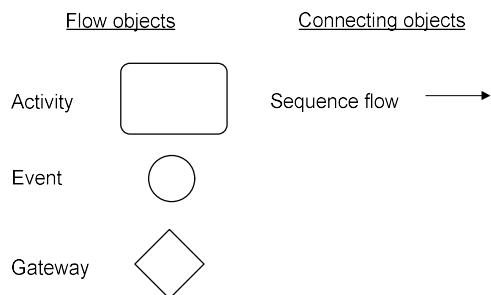


6



7

## Basic Elements

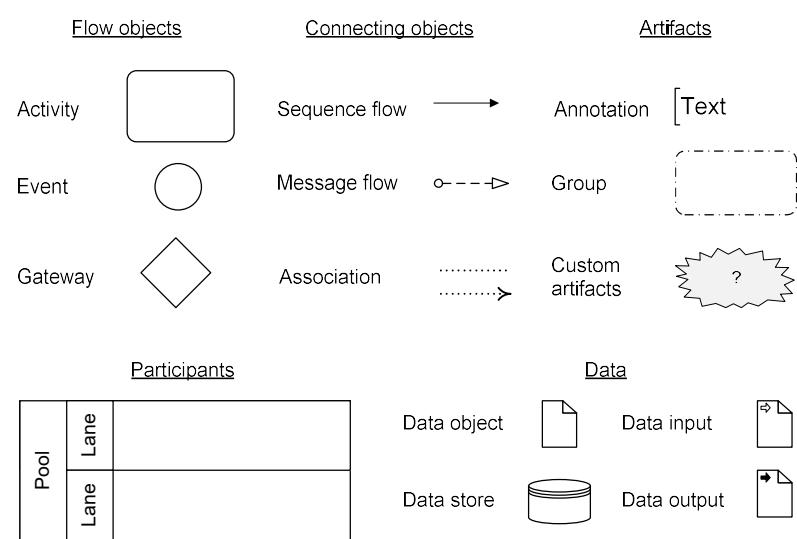


7



8

## Basic Elements - Documentation

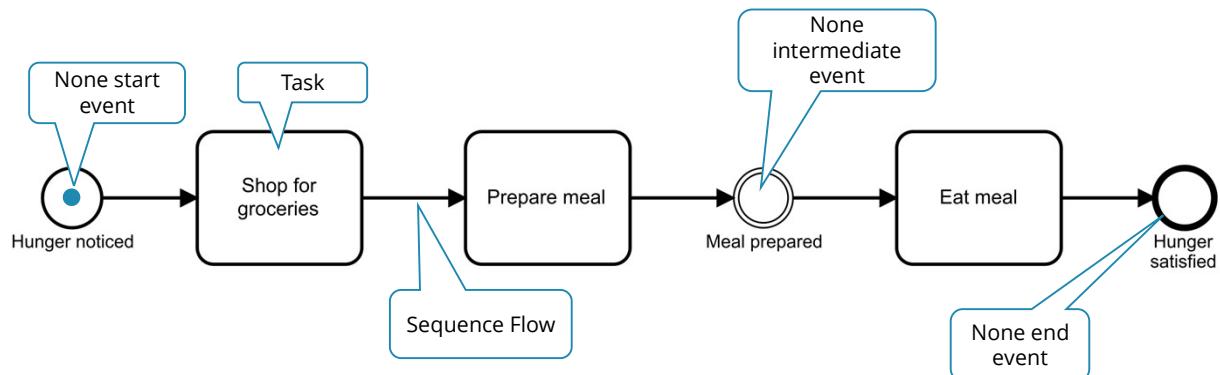


8



9

## Tasks and events

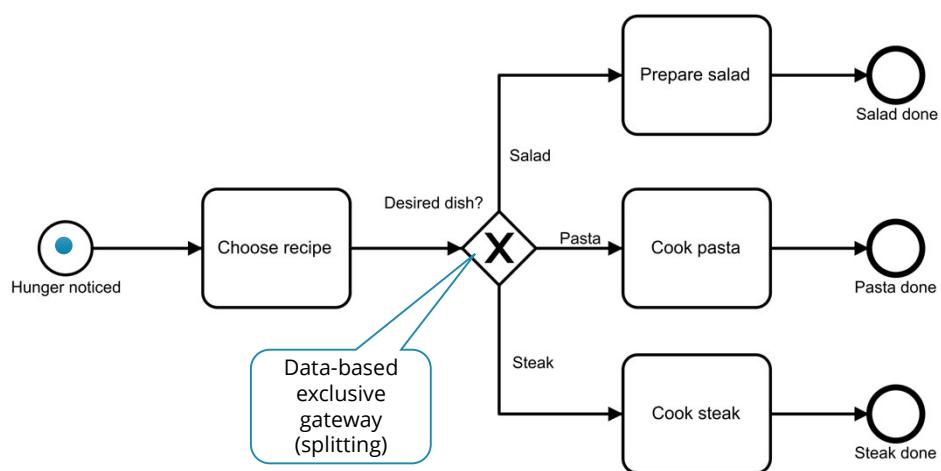


9



10

## The XOR-gateway

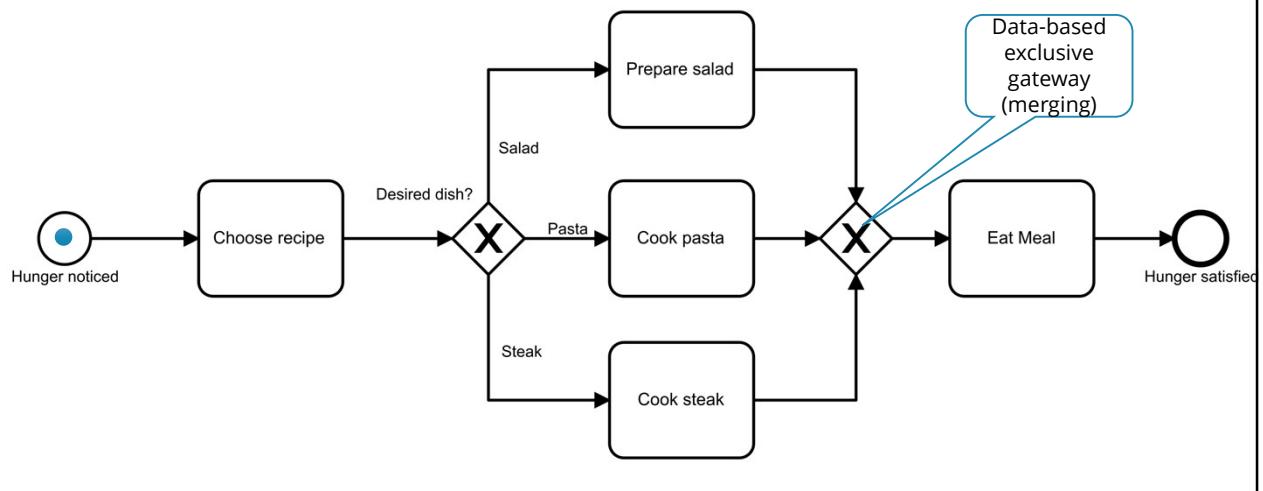


10



11

## XOR-gateways can also merge

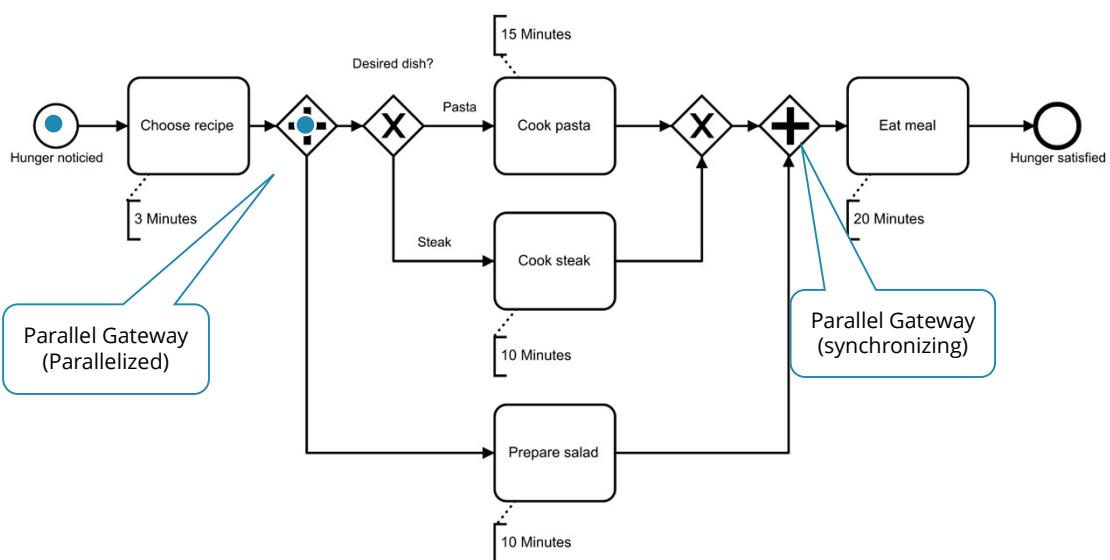


11



12

## Preparing salad and main course at the same time

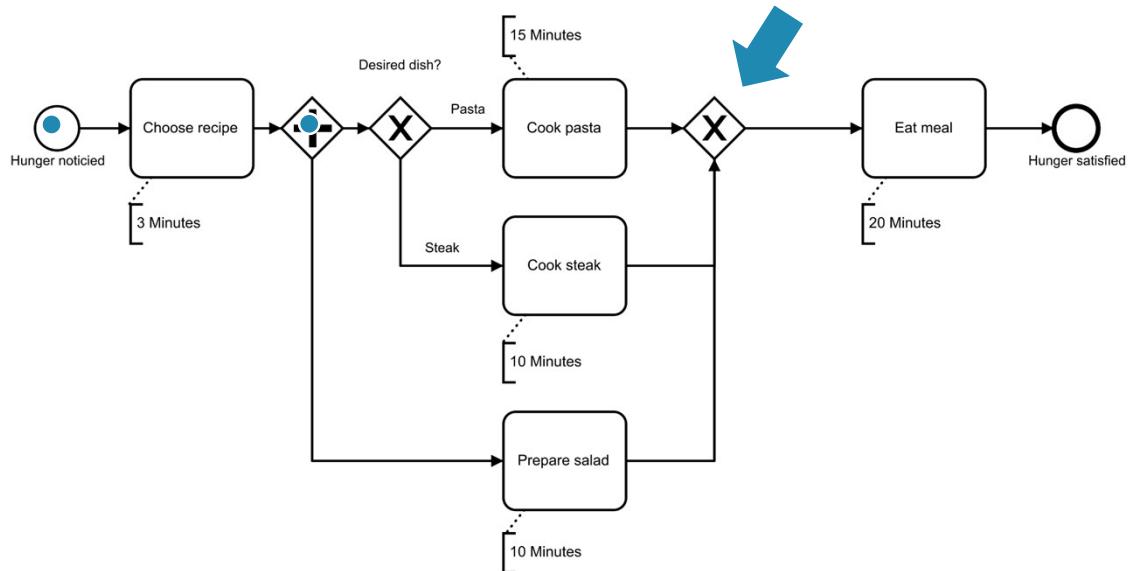


12



13

## What happens in this process?

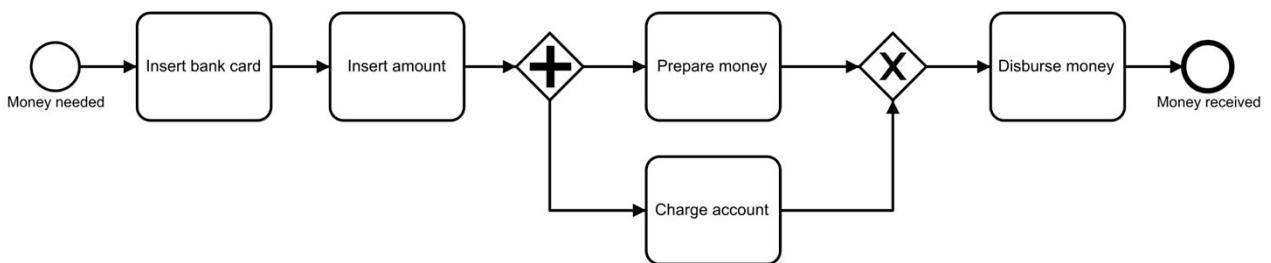


13



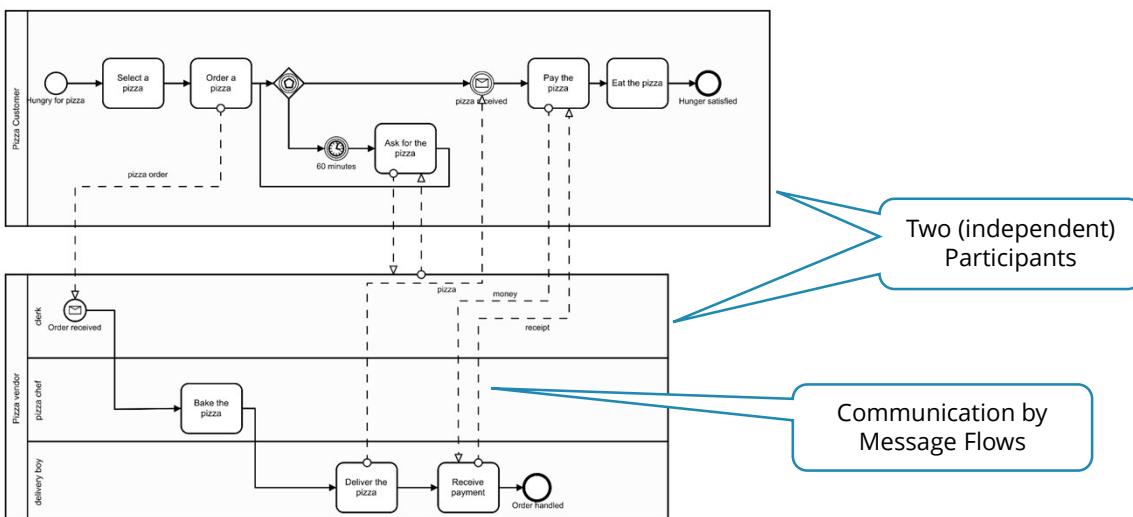
14

## Example: ATM



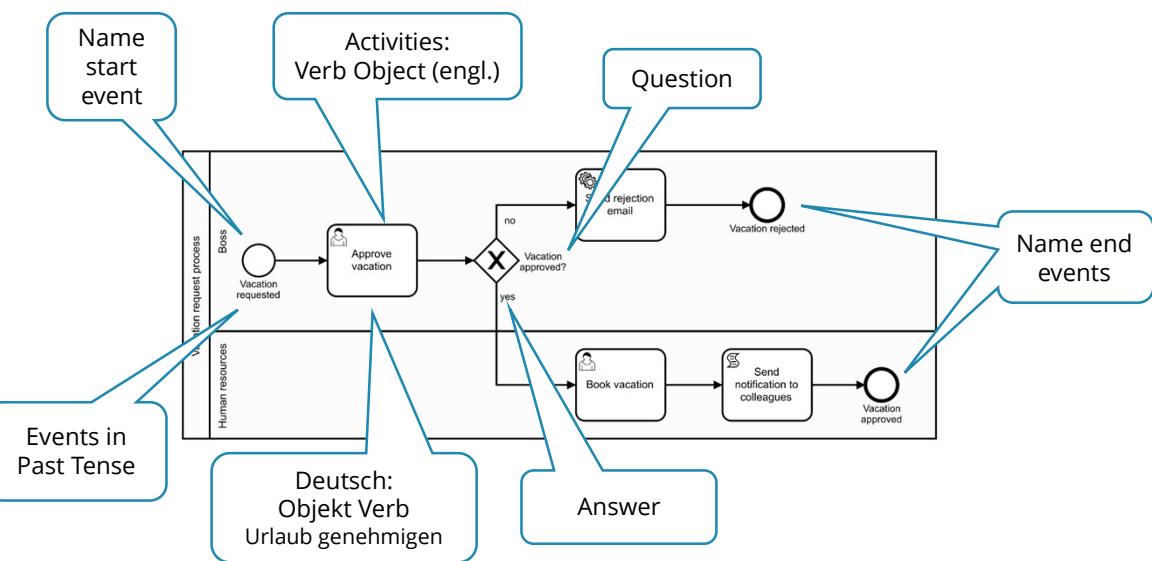
14

## Collaboration Diagram

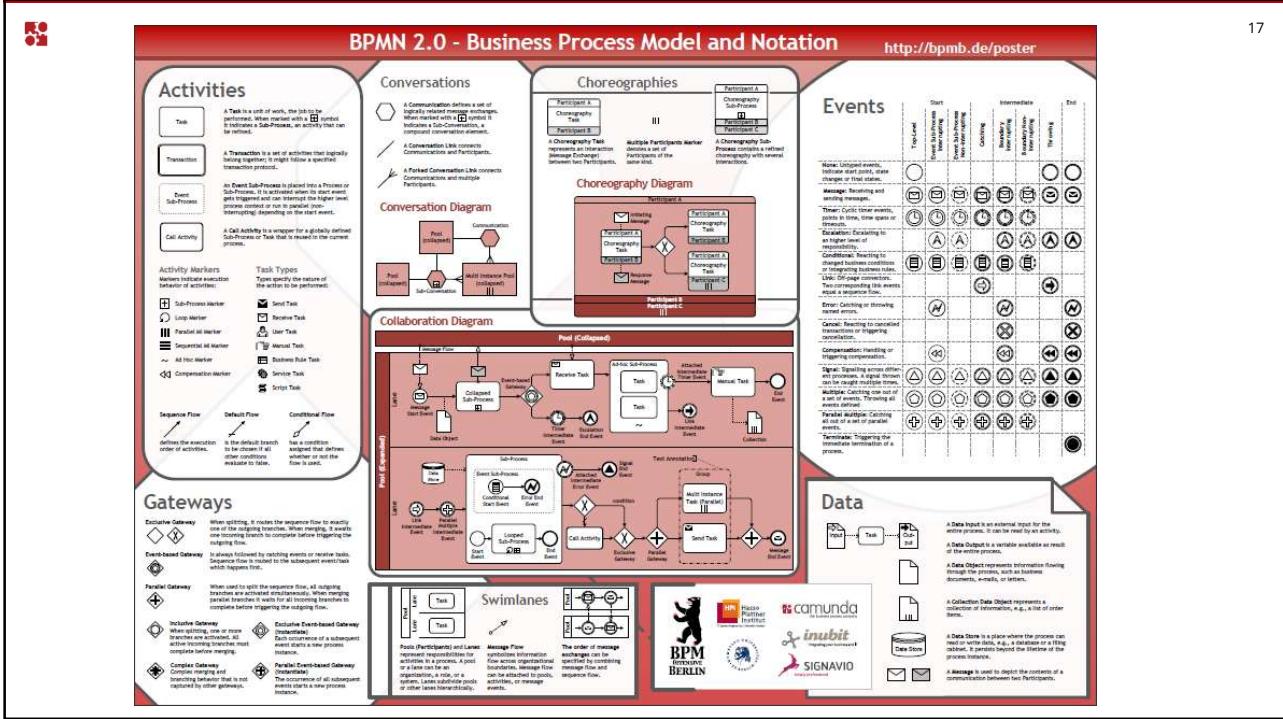


15

## Create Readable Process Models



16



17

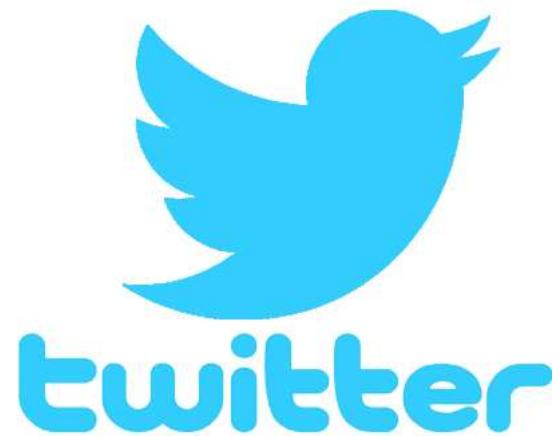
## Scenario for Exercises

18



19

You Know Twitter?



19



20

Imagine the Anti Agile Inc.

**Why do we need Twitter?**

- We do not trust our employees.
- We want to control everything.
- We do not like Twitter,
- But we want to be part of it.



20

# The Anti Agile Inc. Needs Twitter QA!

21

## Group Exercise 0

Use a whiteboard or flipchart  
to model a Twitter QA process



22



23

## QA Process for Twitter

23



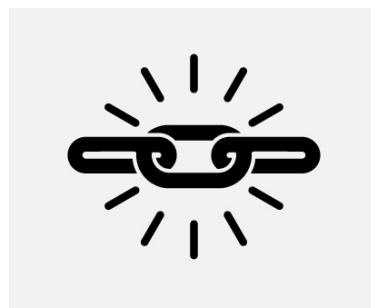
24

## Process Automation Use Cases

Human Task Management

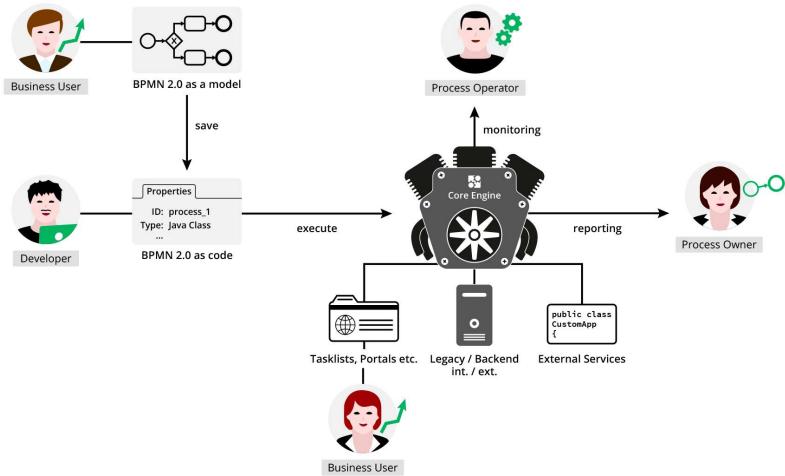


System Integration



24

## What's So Cool About BPMN 2.0



25

## Why Using a Process Engine?

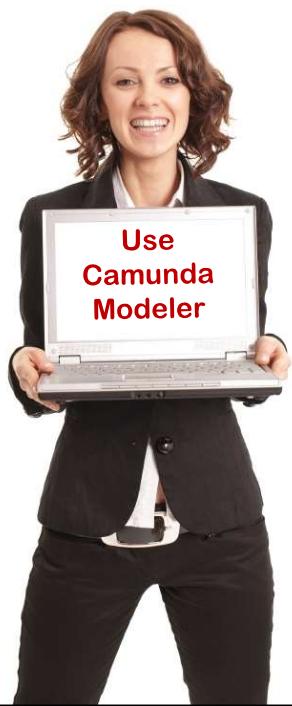
- Features
  - Task management
  - Timers & escalation
  - Error management and retry mechanisms
  - Versioning
  - The power of BPMN 2.0
  - Tools (modeler, cockpit, ...)
  - ...
- Transparency and Agility
  - Process is not hidden in the code
  - Process model can be used for monitoring and operations
- Quality

26



27

## Demo

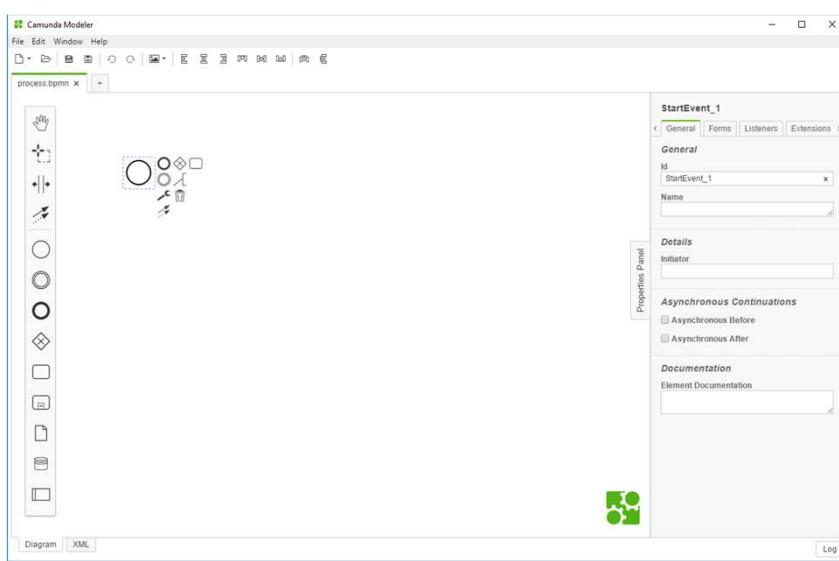


27



28

## Start Modeling!



28

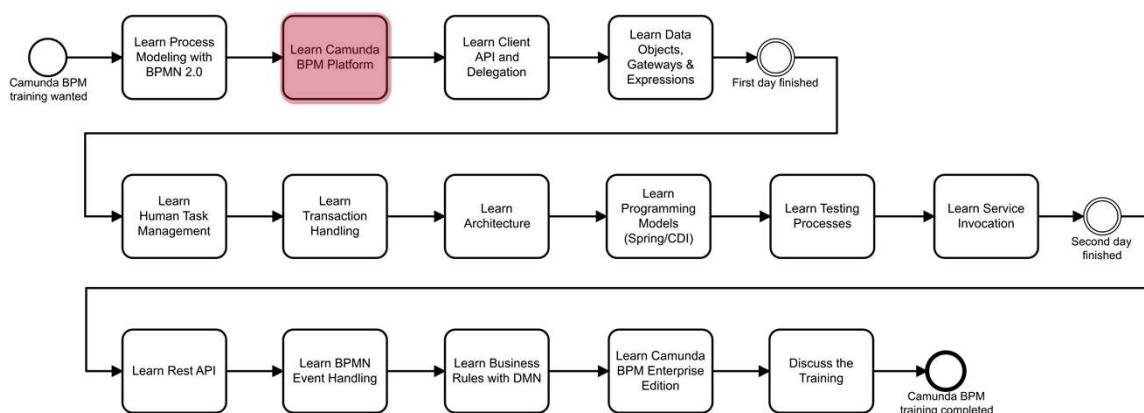


## Exercise 1



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

29

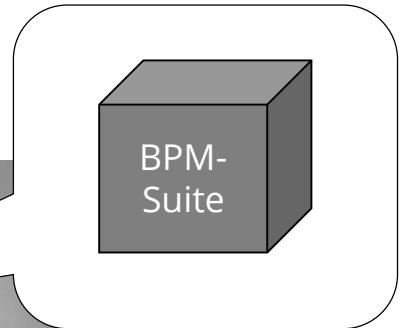


30



31

## Shiny BPM Suites?

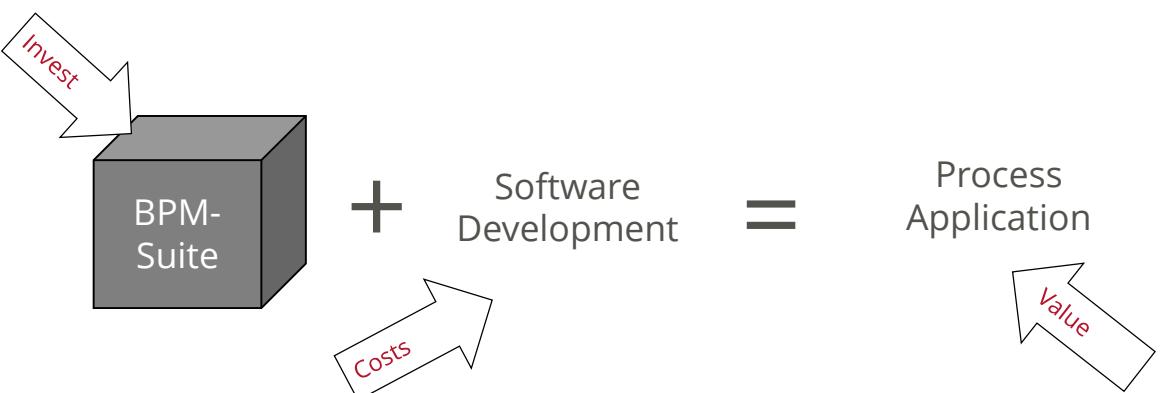


31



32

## You Always Need Software Development



32



33

## Tempting, but Deceptive

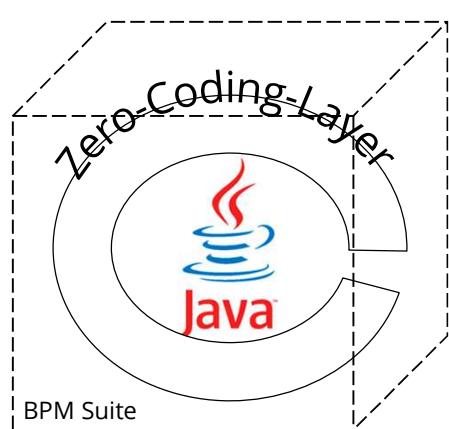


33



34

## A Fundamentally Wrong Approach\*



complicated  
restrictive



Business User

restrictive  
proprietary



Software  
Developer

\*for automating core business processes

34



## Our Approach: BPM + Java

35



**@Inject  
ProcessEngine engine;**

35



## Developer-Friendliness

- A fundamentally different approach compared to Zero-Code Suites!
- It's not about „empowering Business Analysts“. Even with low code approaches coding is still required
- Camunda helps developers be more productive („Less code“)
- Camunda is microservice orchestration ready

<https://camunda.com/learn/whitepapers/developer-friendly-bpm/>

**Developer-Friendly  
Business Process Management**  
May 2014  
Kemsley Design Ltd.

[www.kemsleydesign.com](http://www.kemsleydesign.com)  
[www.column2.com](http://www.column2.com)



36



## In short

Camunda

Developer-Friendly  
System Integration

VS.

Zero-Code BPM Suites

Death by  
Property Panel

Camunda

Developer-Friendly  
System Integration

VS.

Low-Code BPM Suites

You're still coding

37

38

## Best of Breed – Based on Standards



You name it!

Any component from the Java cosmos.

Camunda BPM

Process Automation,  
Monitoring, Operations, ...

Java

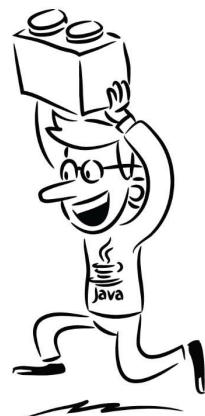
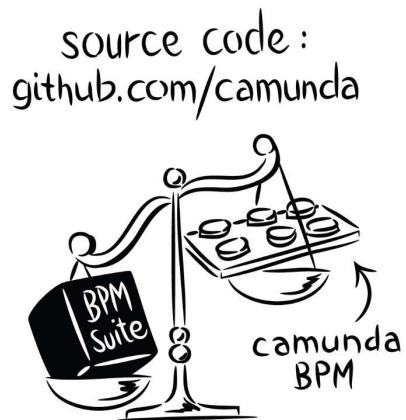
Persistence (JPA), Transactions  
(JTA, EJB3), Clustering, Connectivity  
(JAX-WS, JAX-RS, JAX-B), UI (JSF), ...

BPMN 2.0

Process Modeling,  
Requirements  
Engineering,  
Roundtrip, ...

38

## Lightweight and Open Source BPM Framework



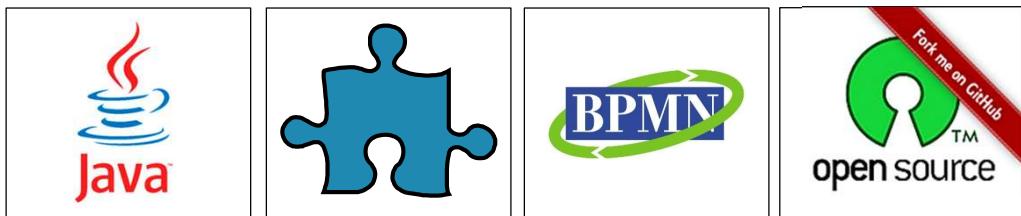
## ...Plus BPMN Based Business-IT-Alignment





41

## Summary: Core Concepts of Camunda BPM



41

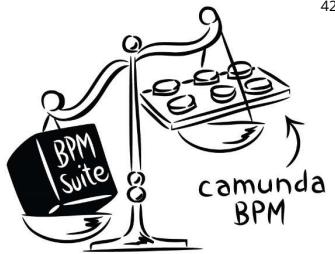


42

Decide Yourself!



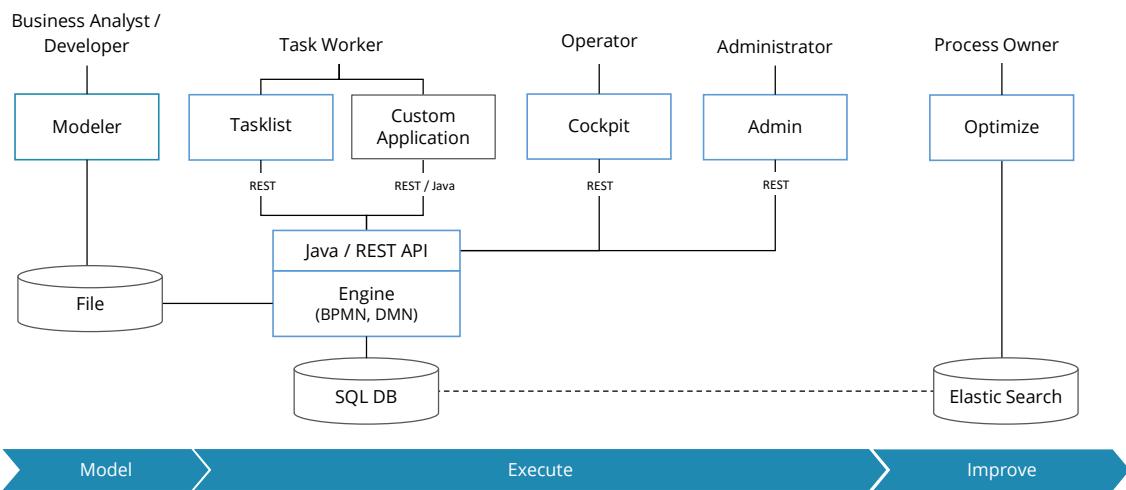
BPM in a can



BPM cook-it-yourself

42

## Camunda BPM Components



43

## Exercise 2a



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

44



45

# Demo



45



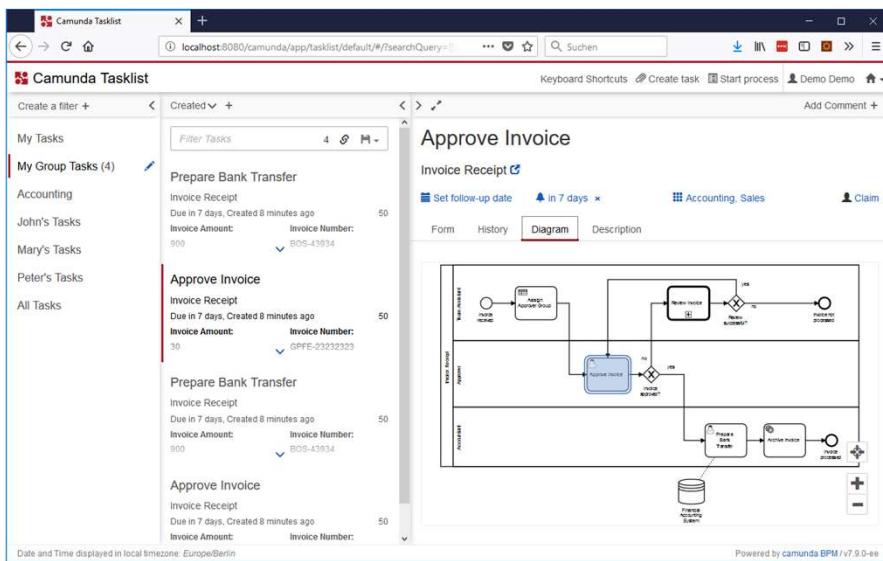
46

## Camunda Tasklist

The screenshot shows the Camunda Tasklist application interface. On the left, there's a sidebar with navigation links like 'My Tasks', 'My Group Tasks (4)', 'Accounting', 'John's Tasks', 'Mary's Tasks', 'Peter's Tasks', and 'All Tasks'. The main area displays a list of tasks under 'My Group Tasks (4)'. One task is selected: 'Approve Invoice' for 'Invoice Receipt' (Due in 7 days, Created 7 minutes ago). The task details show an amount of 30 and an invoice number of GPFE-23232323. To the right, a modal window titled 'Approve Invoice' is open, showing the same task details and a form for approving the invoice. The form includes fields for 'Creditor' (Great Pizza for Everyone Inc.), 'Amount' (30), 'Invoice Category' (Travel Expenses), and 'Invoice Number' (GPFE-23232323). There's also a checkbox labeled 'Do you approve?' and buttons for 'Save' and 'Complete'.

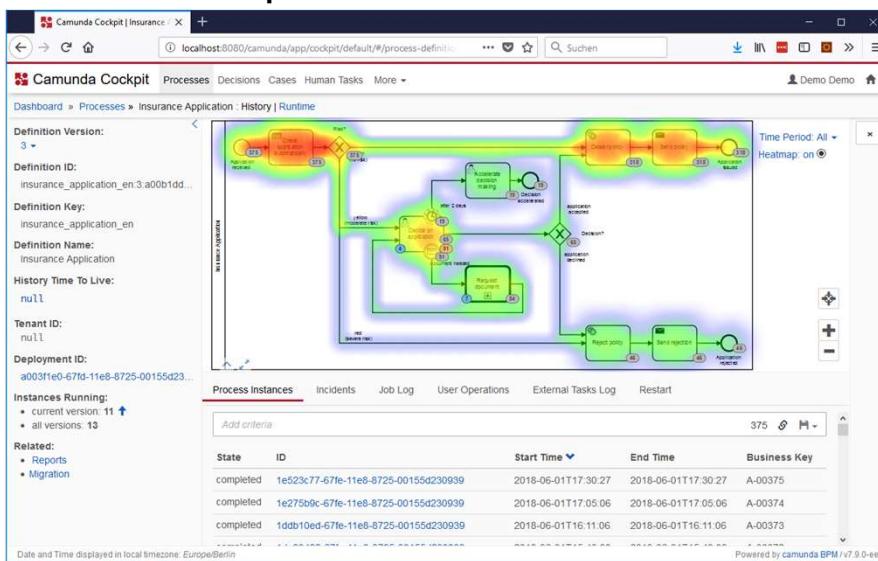
46

## Process Overview



47

## Camunda Cockpit



48



49

## Cockpit Features

- Dashboard
- Monitor BPMN Processes
- Monitor DMN Decisions
- Browse deployments and resources in the process engine repository
- Graphical Process Instance Modification (EE only)
- Graphical Process Instance Migration (EE only)
- Auditing of Cockpit operations (EE only)
- History Cleanup View (EE only)
- Batch view
- Reports (EE only)

49



50

## Camunda Admin

The screenshot shows the Camunda Admin interface on a Windows desktop. The title bar reads "Camunda Admin | Dashboard". The address bar shows the URL "localhost:8080/camunda/app/admin/default/#/". The top navigation bar includes links for "Users", "Groups", "Tenants", "Authorizations", and "System". A dropdown menu shows "Demo Demo". The main content area is divided into five sections:

- Users**: Contains links for "Create New User", "List of Users", and "My Profile".
- Groups**: Contains links for "Create New Group" and "List of Groups".
- Tenants**: Contains links for "Create New Tenant" and "List of Tenants".
- Authorizations**: Contains a link for "Manage Authorizations".
- System**: Contains links for "General", "Execution Metrics", and "License Key".

At the bottom of the page, there is a footer note: "Date and Time displayed in local timezone: Europe/Berlin" and "Powered by camunda BPM / v7.9.0-ea".

50

## Authorizations

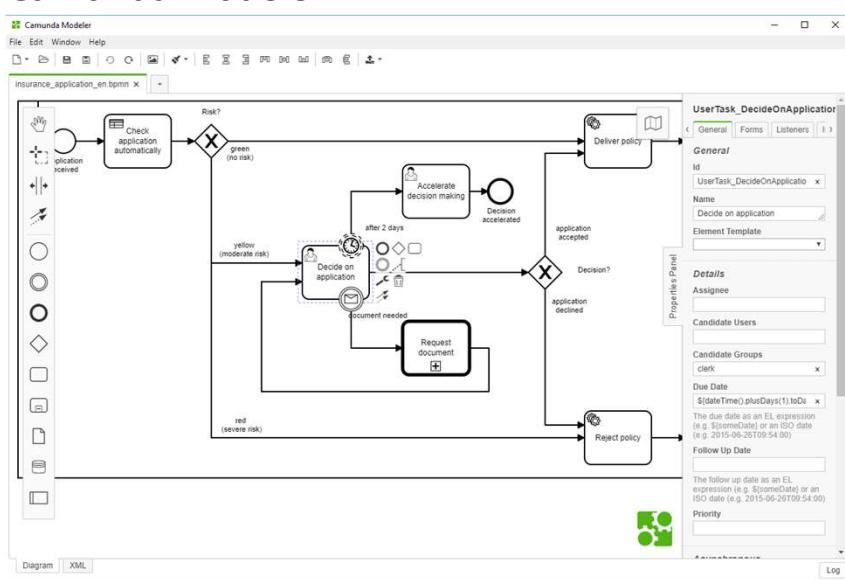
The screenshot shows the Camunda Admin interface with the 'Authorizations' tab selected. On the left, a sidebar lists various administrative categories. The main area displays a table titled 'Process Definition Authorizations' with the following data:

Type	User / Group	Permissions	Resource ID	Action
ALLOW	# accounting	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	# camunda-admin	ALL	*	Edit Delete
ALLOW	# clerk	READ, UPDATE_INSTANCE, READ_HISTORY	insurance-application	Edit Delete
ALLOW	# lisa	READ, READ_INSTANCE, UPDATE_INSTANCE, READ_HISTORY	insurance-application	Edit Delete
ALLOW	# management	READ, READ_HISTORY	invoice	Edit Delete
ALLOW	# sales	READ, READ_HISTORY	invoice	Edit Delete

At the bottom, it says 'Powered by camunda BPM /v7.9.0-ee'.

51

## Camunda Modeler



52

## bpmn.io – a Developer Kit and Web Modeler for BPMN 2.0

The screenshot shows the bpmn.io interface in a browser window. At the top, it says "BPMN everywhere, for everyone" and "Create, embed and extend BPMN diagrams in your Browser. Use it standalone or integrate it into your application." Below this, there are three main sections:

- Model**: Shows a simple BPMN diagram with a start event, a task, and an end event.
- Embed and Annotate**: Shows a more complex diagram with a start event, a task, a decision diamond, and a parallel gateway. A callout box says "Customers like our performance".
- Extend**: Shows a diagram with a start event, a task, a decision diamond, and a parallel gateway. A callout box says "Integrate an in-browser process engine, token simulation, styling or enhanced interactivity. It is up to you because bpmn-js is an open toolkit."

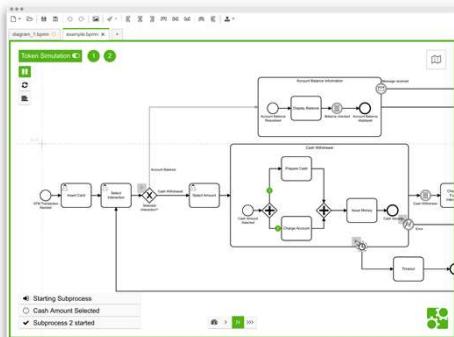
53

## Extend the Camunda Modeler

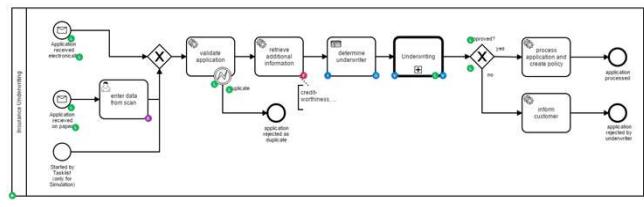
The screenshot shows the Camunda Modeler interface in a browser window. The title bar says "camunda/camunda-modeler-linter-plugin: Check your BPMN diagrams for common issues - Mozilla Firefox". The main workspace displays a BPMN diagram with a start event, a task labeled "Check Account", a decision diamond labeled "Exists?", and a task labeled "Validate Information". A red error icon is visible near the decision diamond. In the bottom status bar, it says "4 Errors, 6 Warnings".

54

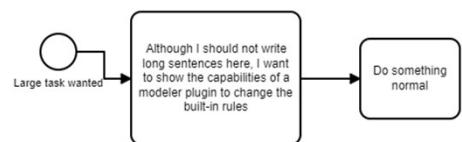
## Camunda Modeler Plugin Examples



Token Simulation Plugin



Property Info Plugin



Resize Tasks Plugin

<https://github.com/camunda/camunda-modeler-plugins>

## cawemo.com

## <https://docs.camunda.org>

The screenshot shows the Camunda BPM Manual homepage. The URL in the address bar is <https://docs.camunda.org/manual/latest/>. The page title is "The Camunda BPM Manual". On the left, there's a sidebar with a navigation menu under "BPM Platform: latest (7.9)" which includes links for Introduction, User Guide, Reference, Installation, Modeler, Web Applications Examples, and Update & Migration. The main content area features a section titled "Getting Started" with a brief introduction and links to various documentation topics like BPMN 2.0, CMMN 1.1, DMN 1.1, and Java EE 7.

57

## Camunda Best Practices

The screenshot shows the Camunda Best Practices homepage. The URL in the address bar is <https://camunda.com/best-practices/>. The page title is "Camunda Best Practices". The main content area explains what Best Practices are and how to find them. It includes sections for "For Stakeholders", "By Alphabet", "By Keywords", and "Customer Success Path". A sidebar on the left lists categories like Architecture, Development, and Integration. At the bottom, there's a footer with copyright information and a link to the homepage.

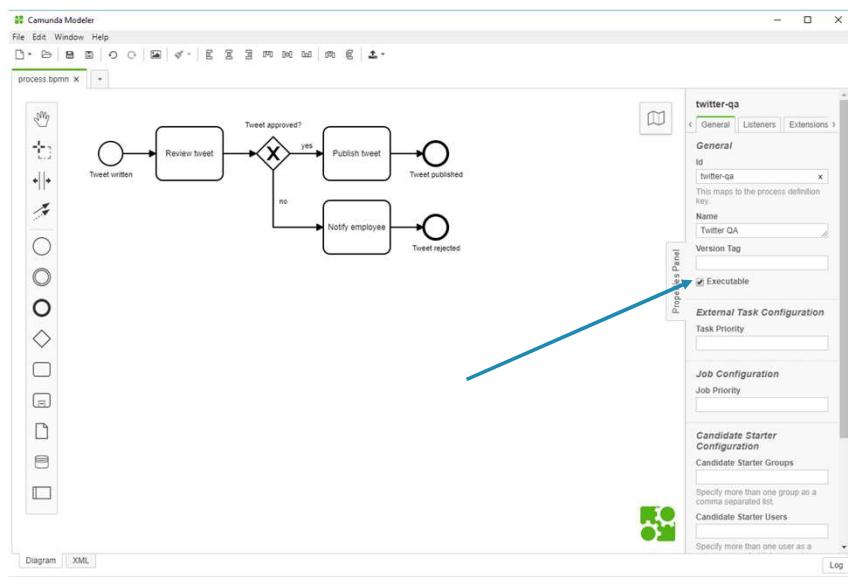
58

# Prepare for First Launch

59

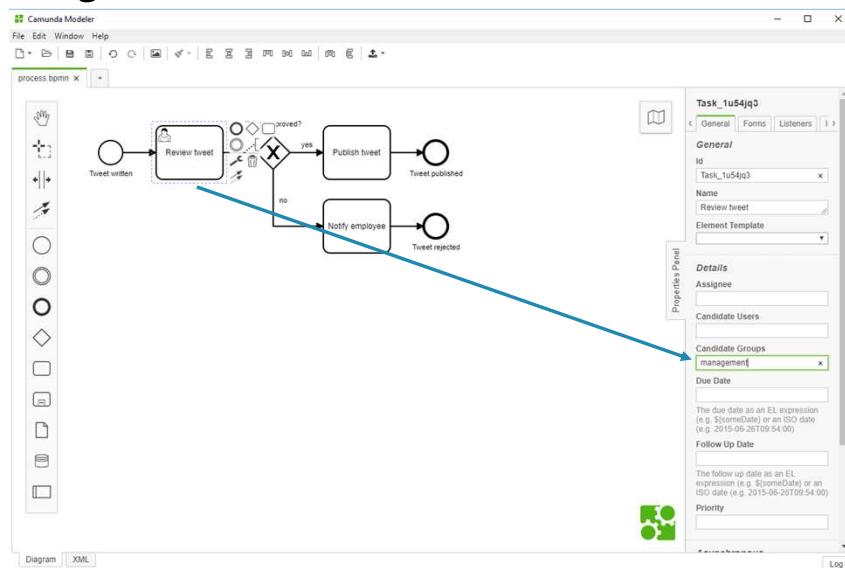
60

## Make Process Executable



60

## Assign User Task



**Details**

**Assignee**

**Candidate Users**

**Candidate Groups** management

**Due Date**

The due date as an EL expression (e.g. \${someDate}) or an ISO date (e.g. 2015-06-26T09:54:00)

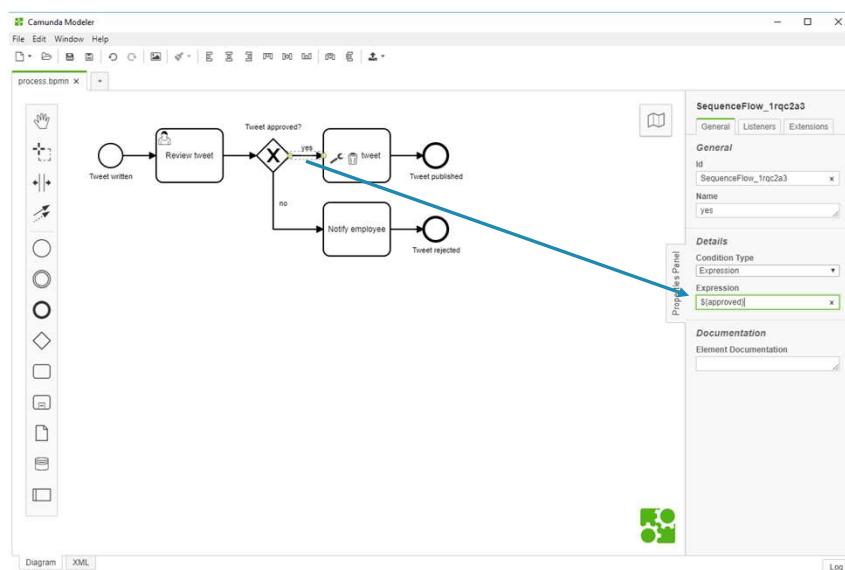
**Follow Up Date**

The follow up date as an EL expression (e.g. \${someDate}) or an ISO date (e.g. 2015-06-26T09:54:00)

**Priority**

61

## Conditions



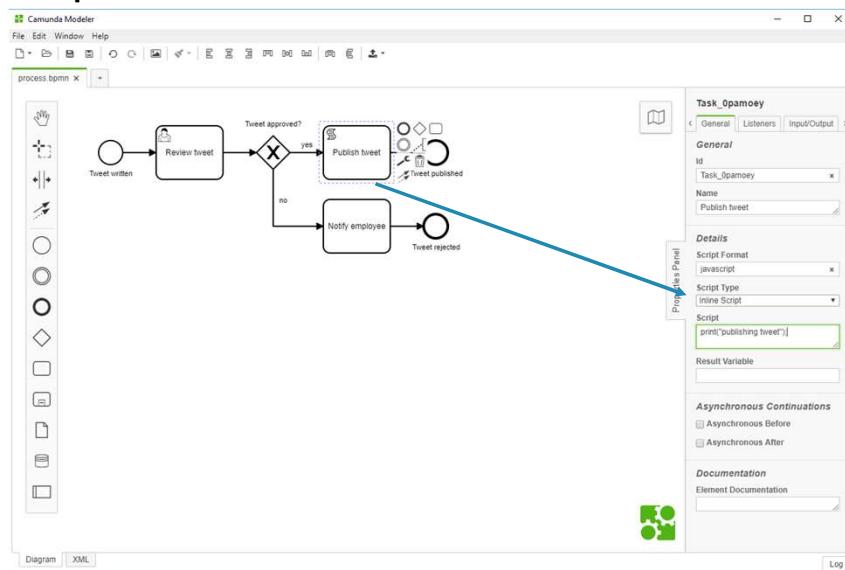
**Details**

**Condition Type** Expression

**Expression** \${approved}

62

## Scripts



### Details

**Script Format**

javascript

**Script Type**

Inline Script

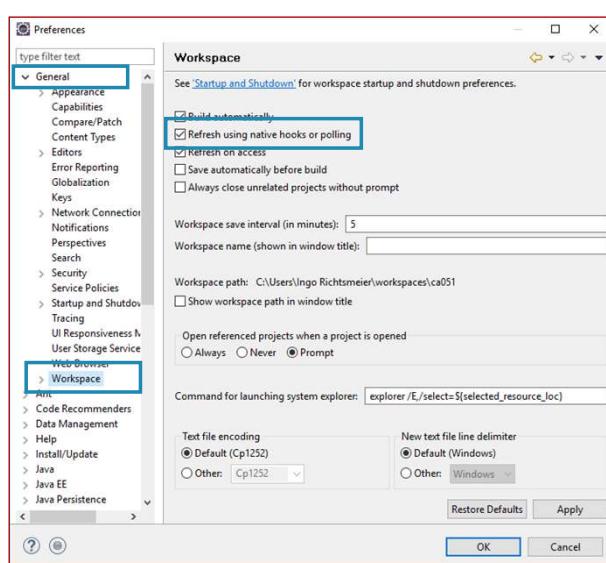
### Script

```
print("publishing tweet");
```

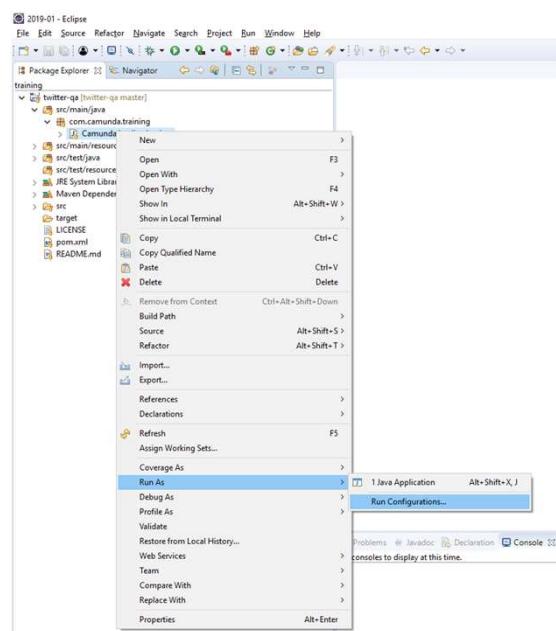
### Result Variable

[empty]

## Setup Eclipse for Automatic Synchronization



## Start the application



65

## Use Generic Task Forms

The screenshot shows the Camunda Tasklist web application. A generic task form is open, prompting the user to set variables. It includes a table with columns 'Name', 'Type', and 'Value'. A sample entry 'content' of type String is shown with the value 'My first tweet from Twitter QA'. Below the table, there are buttons for 'Back', 'Close', and 'Start'.

66

## Complete Generic Task Forms

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with navigation links like 'My Tasks', 'My Group Tasks (4)', 'Accounting', 'John's Tasks', 'Mary's Tasks', 'Peter's Tasks', and 'All Tasks'. The main area displays several generic task forms:

- Review tweet**: A form for managing tweets. It includes fields for 'content' (String type) with value 'My first tweet from Ti' and 'approved' (Boolean type) with checked status.
- Twitter QA**: A form with tabs for 'Form', 'History', 'Diagram', and 'Description'. It has buttons for 'Set follow-up date' and 'Set due date'.
- Approve Invoice**: A form for approving invoices. It shows two entries: one for 'Invoice Receipt' with amount 50 and another for 'Invoice Number: 900' with value '0FFE-2323233'; and another for 'Invoice Receipt' with amount 50 and another for 'Invoice Number: 900' with value 'BOB-43934'.
- Prepare Bank Transfer**: A form for preparing bank transfers. It shows two entries: one for 'Invoice Receipt' with amount 50 and another for 'Invoice Number: 900' with value 'BOB-43934'; and another for 'Invoice Receipt' with amount 50 and another for 'Invoice Number: 900' with value 'BOB-43934'.

At the bottom, there's a note: 'Date and Time displayed in local timezone: Europe/Berlin' and 'Powered by camunda BPM / v7.9.0-ee'.

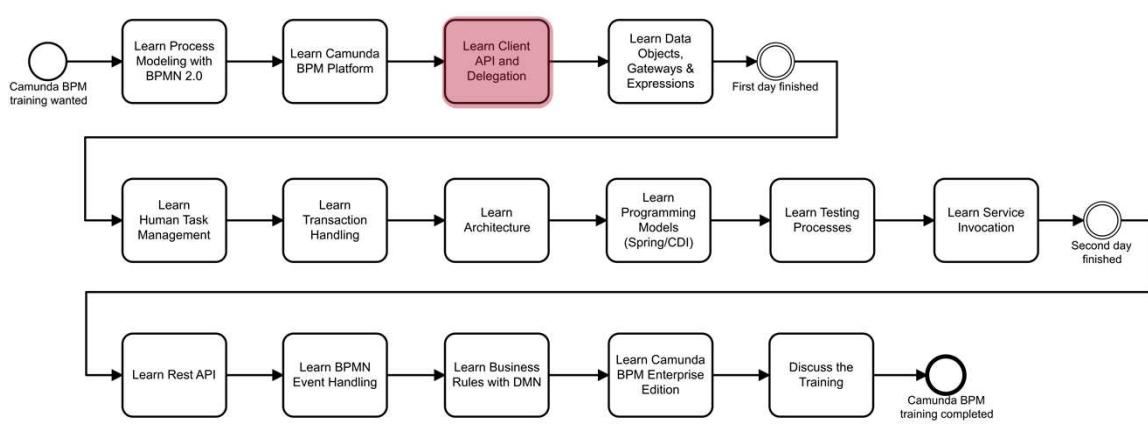
67

## Exercise 2b

<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks



68



69

## BPM + Java

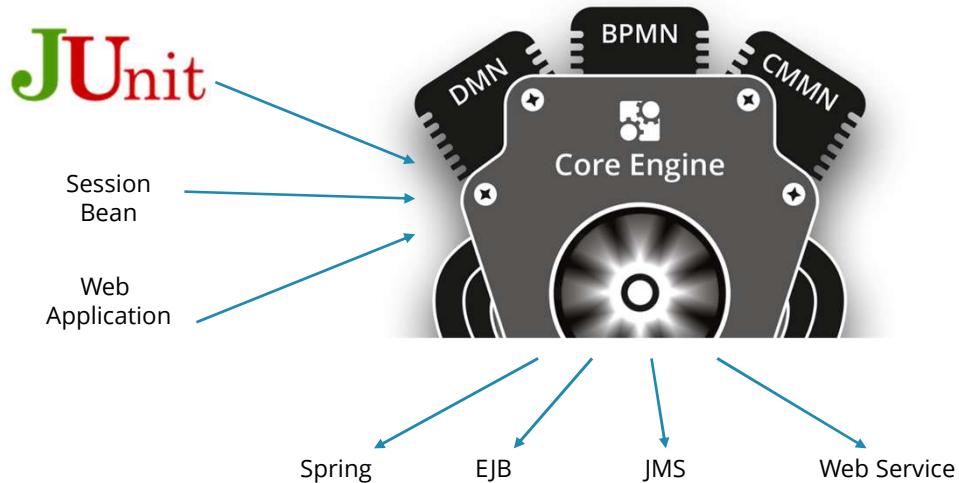


70



71

## We Need a Way INTO and OUT of the Engine

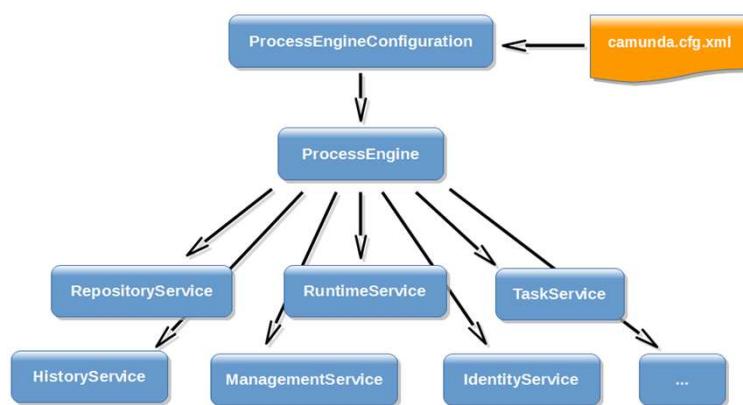


71



72

## Services, the Way IN



72



## Overview Engine Services

Service	Description
Repository	Deployment, Process definitions
Runtime	Start of process instances, Variables, Event correlation
Task	Task lists, Task life cycle
ExternalTask	External task lifecycle
Decision	Evaluate decisions
History	Logs, Statistics, KPIs
Identity	Users, Groups
Form	Form key, Form property, Form
Management	Jobs, Statistics, Maintenance
Filter	Tasklist filters
Authorization	Permissions
Case	Start of cases, Case execution lifecycle



## Java Client API

```
ProcessEngine processEngine = ProcessEngineConfiguration
    .createStandaloneProcessEngineConfiguration()
    .setJdbcUrl("jdbc:h2:./camunda-h2-dbs/process-engine;DB_CLOSE_DELAY=1000")
    .setDatabaseSchemaUpdate("true")
    .setJobExecutorActivate(true)
    .buildProcessEngine();

processEngine.getRepositoryService()
    .createDeployment()
    .addClasspathResource("vacation-request.bpmn")
    .deploy();

ProcessInstance processInstance = processEngine.getRuntimeService()
    .startProcessInstanceByKey("VacationRequestProcess");
```



75

## Process Versioning

### Process Definitions

ID_	KEY_	VERSION_	...
invoice:1:12ce6c9a-5b09-11e6-bfac-185e0f710ea8	invoice	1	
invoice:2:131a1ba0-5b09-11e6-bfac-185e0f710ea8	invoice	2	

### Process Instances

ID_	PROC_DEF_ID_	ACT_ID_	...
1378dcae-5b09-11e6-bfac-185e0f710ea8	invoice:1:12ce6c9a-5b09-11e6-bfac-185e0f710ea8	reviewInvoice	
14b81440-5b09-11e6-bfac-185e0f710ea8	invoice:2:131a1ba0-5b09-11e6-bfac-185e0f710ea8	approveInvoice	

75



76

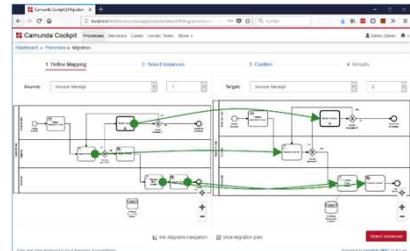
## Start New Process Instances

Default Behavior:

1. New version upon each changed BPMN XML deployment
2. New process instances start with the latest version
3. Running process instances continue using the version they started with

```
ProcessInstance processInstance = processEngine.getRuntimeService()
    .startProcessInstanceByKey("VacationRequestProcess");
```

If needed, process instance migration is possible



76

## Key vs. ID vs. Name

```
ProcessInstance processInstance = processEngine.getRuntimeService()
    .startProcessInstanceByKey("VacationRequestProcess");
```

- ProcessDefinitionKey: BPMN Process ID
- ProcessDefinitionName: BPMN Process Name
- ProcessDefinitionId: Generated Unique ID

```
<bpmn2:process id="VacationRequestProcess"
    name="Vacation request"
    isExecutable="true">
```

Participant\_041airq

General	Listeners	Extensions
<b>General</b>		
Id	Participant_041airq	
Name	Vacation request process	
Version Tag		
Process Id	VacationRequestProcess	
Process Name	Vacation request	
<input checked="" type="checkbox"/> Executable		

## Process Variables

Are persisted with the process instance

### Client Code:

```
Map<String, Object> variables = new HashMap<String, Object>();
variables.put("employee", "Jim");
runtimeService().startProcessInstanceByKey("VacationRequestProcess", variables);
```

**More details later**



## Business Key

- Domain specific identifier of a process instance
- Optional
- Set at the start
- Set or change in a Java Delegate or Execution Listener
- Displayed prominently in Cockpit and Tasklist
- Query for process instances

Candidates for the Business Key  
in the Twitter QA process?

Start process

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

Business Key	<input type="text"/>		
Add a varia... +	Name	Type	Value
Rem... x	<input type="text"/> content	<input type="button" value="String"/>	My first tweet from Twitter QA

Back Close Start

```
runtimeService.startProcessInstanceByKey(  
    "VacationRequestProcess", employee);
```



## Camunda BPM Assert Library

- For comprehensive JUnit tests
- Many helper methods
- E.g. withVariables:

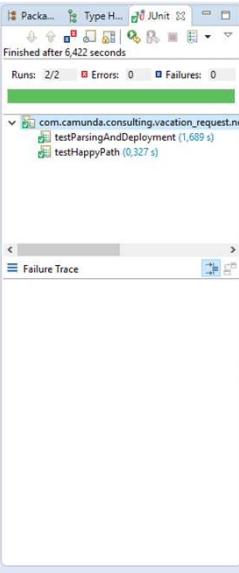
```
public static Map<String, Object> withVariables(final String key, final Object value,  
    final Object... furtherKeyValuePairs)
```

- E.g. access process engine services:

```
public static RepositoryService repositoryService()  
public static RuntimeService runtimeService()  
public static TaskService taskService()  
public static ExternalTaskService externalTaskService()
```

<https://github.com/camunda/camunda-bpm-assert>

## Unit Testing



```

import static org.camunda.bpm.engine.test.assertions.ProcessEngineTests.*;
public class VacationRequestProcessTest {

    @Rule
    public ProcessEngineRule rule = new ProcessEngineRule();

    @Test
    @Deployment(resources = "vacation-request.bpmn")
    public void testHappyPath() {
        ProcessInstance processInstance = runtimeService()
            .startProcessInstanceByKey(PROCESS_DEFINITION_KEY);
        assertThat(processInstance).isWaitingAt("approveVacationUserTask");
        complete(task(), withVariables("approved", true));
        assertThat(processInstance).isWaitingAt("bookVacationUserTask");
        complete(task());
        assertThat(processInstance).isEnded();
    }
}

```

Shortcuts from camunda-bpm-assert

81

## Exercise 3a



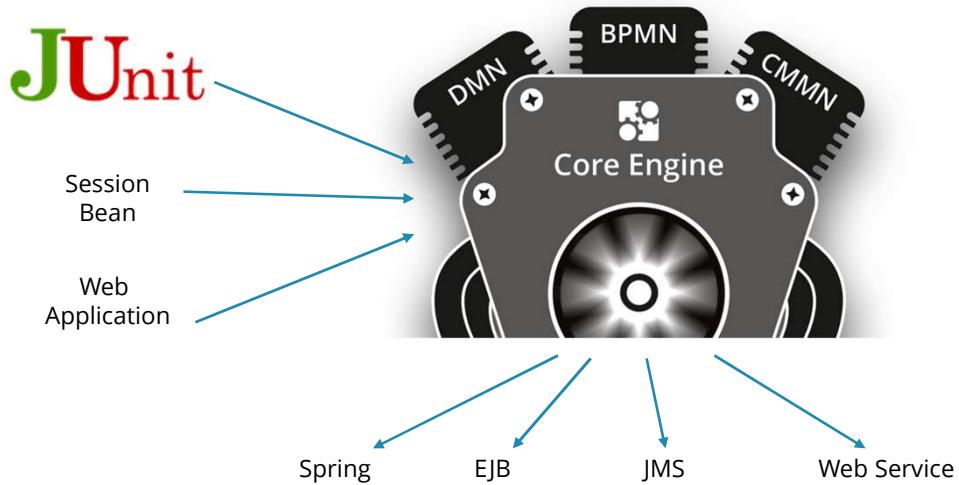
<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

82



83

## The Way OUT

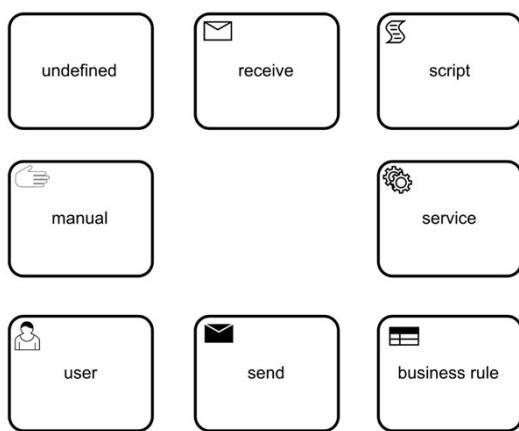


83



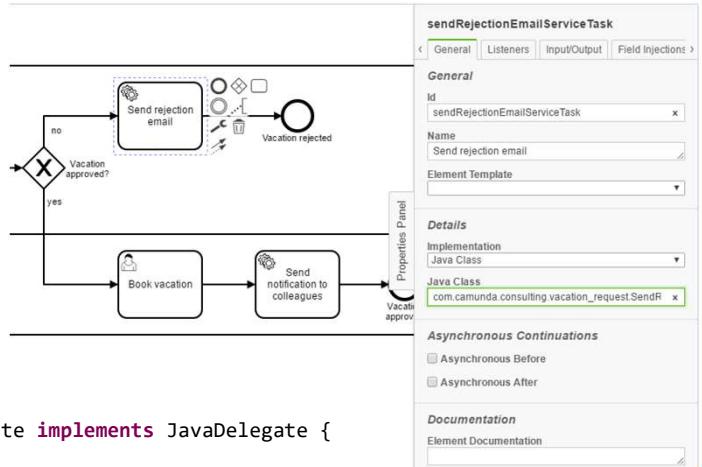
84

## BPMN 2.0 Task Types



84

## Service Task With Java Class



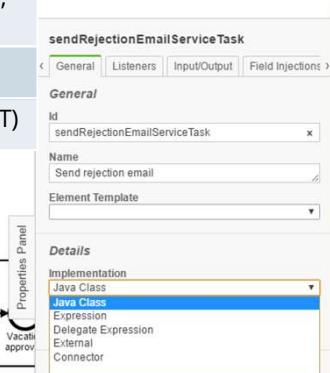
```
public class SendRejectionNotificationDelegate implements JavaDelegate {
    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Send the rejection to the employee...
        String reason = (String) execution.getVariable("rejectionReason");
        execution.setProcessBusinessKey("my new Business Key");
    }
}
```

85

## Other Options for Service Tasks

Implementation	Description
Expression	Execute a method expression, e.g. invoke a bean method
Delegate Expression	Resolve the implementation from an expression, that implements JavaDelegate , e.g. a bean
External	Implement the service outside of the engine
Connector	Configure the call to a web service (SOAP or REST)

More details come later

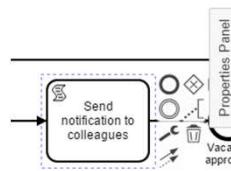
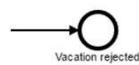


86



## Script Task

- Every JSR-223 compliant scripting engine can be used (we tested JS, Groovy, Jython, JRuby)
- Script sources can be externalized
- Scripts get compiled and cached
- Available variables in all script languages:
  - all process variables
  - execution
  - task



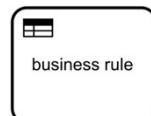
sendNotificationScriptTask

General	Listeners	Input/Output	Extensions
<b>General</b>			
<b>Id</b>	sendNotificationScriptTask		
<b>Name</b>	Send notification to colleagues		
<b>Details</b>			
<b>Script Format</b>	JavaScript		
<b>Script Type</b>	Inline Script		
<b>Script</b>	print("Send notification to colleagues");		
<b>Result Variable</b>			



## Business Rule Task

- Behaves like a Service Task
- Link to a DMN Decision Table
- More details later



## Execution Listeners

```

public class DataTransformationListener implements ExecutionListener {

    @Override
    public void notify(DelegateExecution execution) throws Exception {
        String upperVar = (String) execution.getVariable("someText");
        upperVar = upperVar.toUpperCase();
        execution.setVariable("someText", upperVar);
    }
}

```

89

## Events for Execution Listeners

Level	Event
On process level	Start, end
On activity level	Start, end
On sequence flow	take

### Listener Type

Java Class

Expression

Delegate Expression

Script

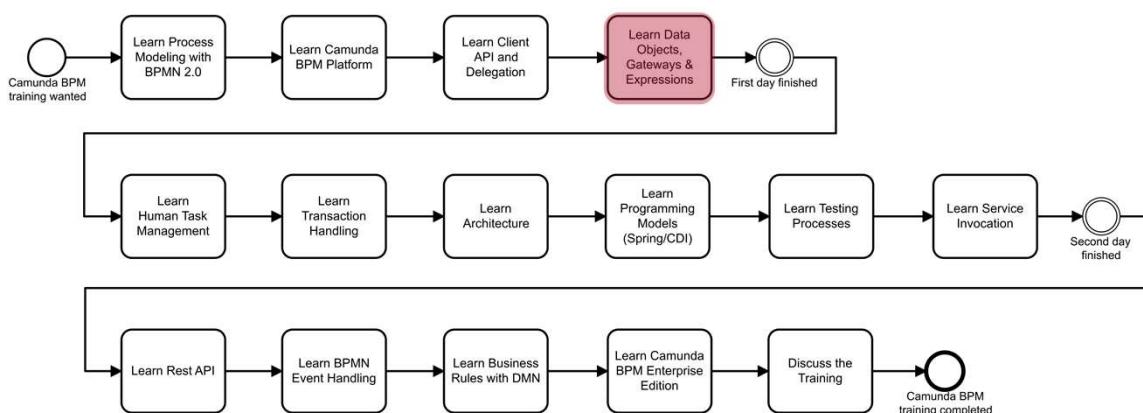
90

## Exercise 3b



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

91



92



## Accessing Process Variables

### Client Code:

```
Map<String, Object> variables = new HashMap<String, Object>();  
variables.put("employee", "Jim");  
runtimeService().startProcessInstanceByKey("VacationRequestProcess", variables);
```

### Delegate Code:

```
public class SendRejectionNotificationDelegate implements JavaDelegate {  
  
    @Override  
    public void execute(DelegateExecution execution) throws Exception {  
        // Send the rejection to the employee...  
        String reason = (String) execution.getVariable("rejectionReason");  
        String employee = (String) execution.getVariable("employee");  
  
        // create a message to the employee  
        execution.setVariable("message", "Some message to " + employee + " about " + reason);  
    }  
}
```



## Process Variables

- Are not modeled in BPMN 2.0
- Behave like a `java.util.Map`
- Are persisted with the process instance
- Recommended: Primitive types, Date & String
- Objects can be serialized as XML or JSON
- Default fallback: Serializable Java Objects (not recommended!)
- Serialization can be exchanged:  
<https://docs.camunda.org/manual/latest/user-guide/data-formats/data-formats-in-processes/#extending-serialization>





## Object Variables

Saving values:

```
Customer customer = new Customer();
...
ObjectValue customerJson = Variables
    .objectValue(customer)
    .serializationDataFormat(SerializationDataFormats.JSON)
    .create();
execution.setVariable("customer", customerJson);
```

*Serialize explicitly*

Reading values:

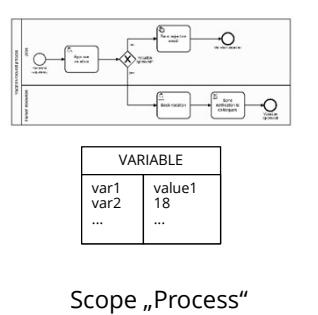
```
ObjectValue customerJson = execution.getVariableTyped("customer");
Customer customer = (Customer) customerJson.getValue();
```

*Deserialize transparently*

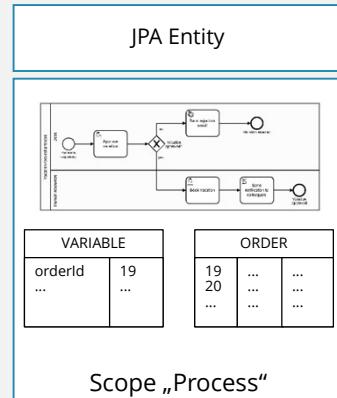


## Storage

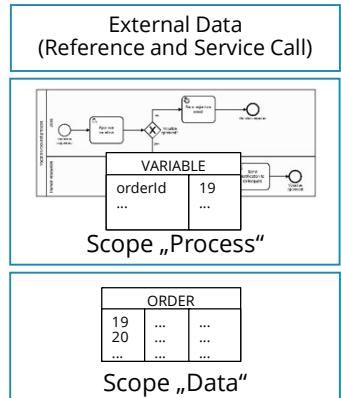
### Process Variable



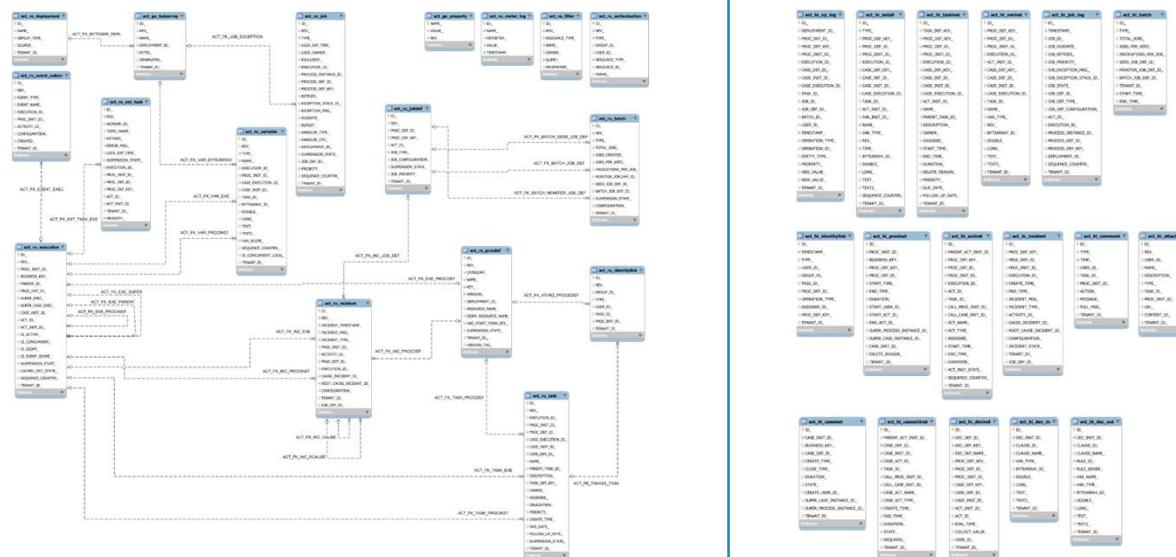
### JPA Entity



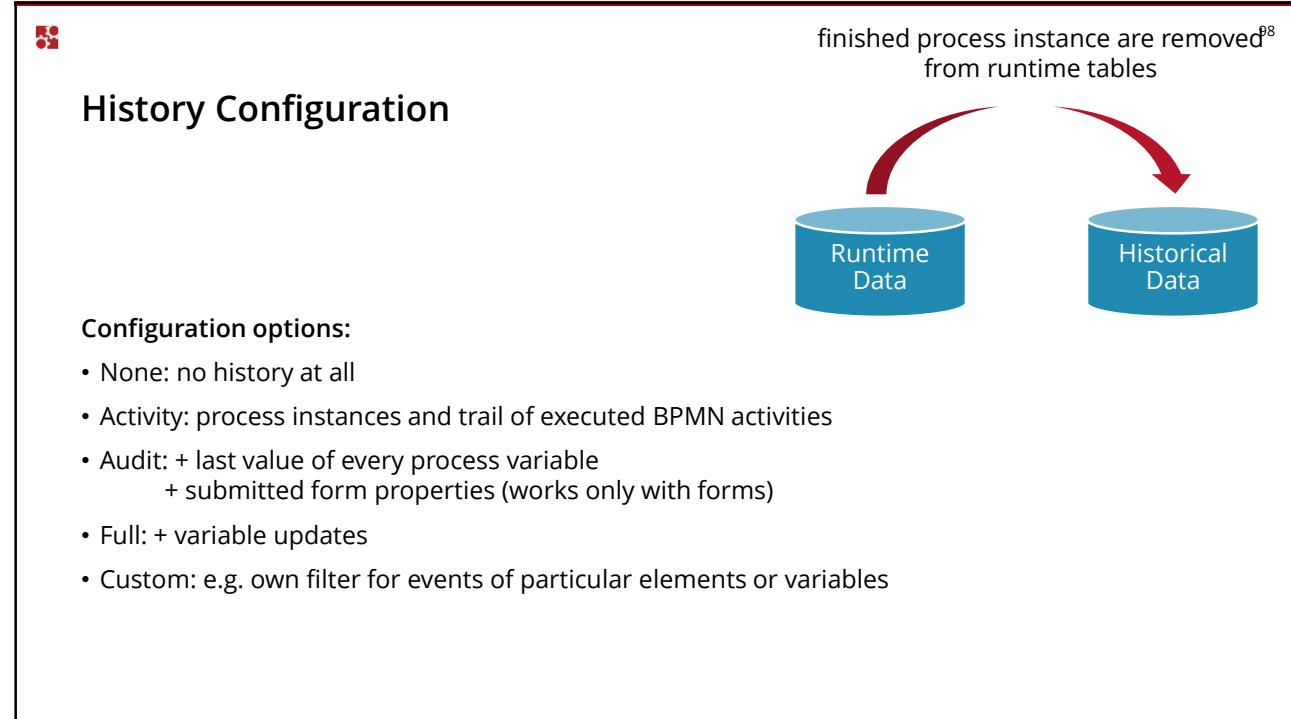
### External Data (Reference and Service Call)



## Database Schema

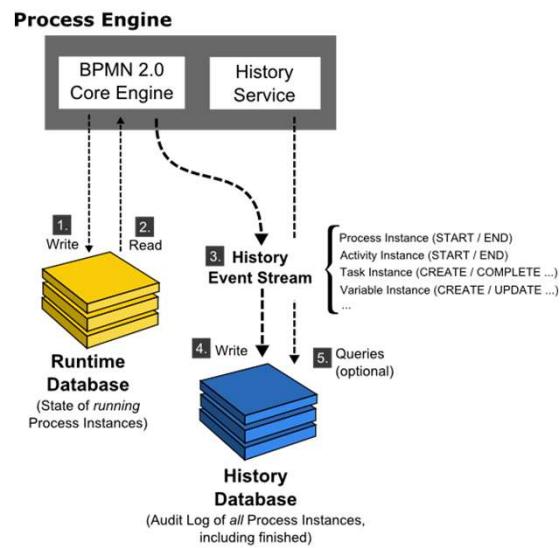


<https://docs.camunda.org/manual/latest/user-guide/process-engine/database/#database-schema>





## Asynchronous History



## History Cleanup

- `historyTimeToLive` for each process definition: Number of days
- defined by business needs
- either in the process model
- or by API-Call
- or in the Cockpit

The screenshot shows the **History Configuration** interface. It includes a **History Time To Live** input field set to 180, a **Documentation** section, and a table titled **Cleanable Data** showing process definitions with their history time to live settings.

Process Definition	Version	Tenant ID	Finished	Cleanable	History Time To Live
Insurance Application	3		364	0	365 days
Document Request	2		84	0	365 days
Vacation request	1		2	0	180

- run periodically, configure by weekdays
- Run manually

Next cleanup will run in 7 hours

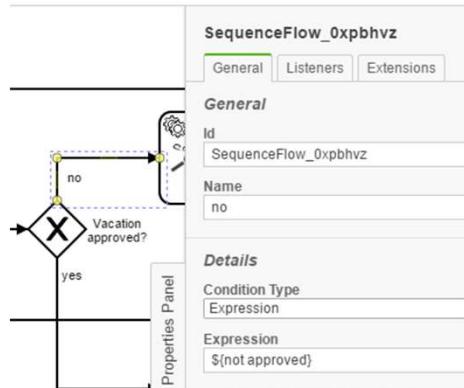
Cleanup View  
Enterprise Feature

Cleanup Now



## Expressions

- Work on data objects
- May work on Spring/CDI beans
- Used in:
  - Java service tasks
  - Event listeners
  - Conditional sequence flows
- Can use attributes or methods



```
 ${myVar}  
 ${myBean.myProperty}
```

```
 ${printer.print()}  
 ${myBean.addNewOrder('orderName')}  
 ${myBean.doSomething(myVar, execution)}
```



## Unified Expression Language (UEL)

EL Expression	Result
<code> \${1 &gt; (4/2)}</code>	false
<code> \${4.0 &gt;= 3}</code>	true
<code> \${100.0 == 100}</code>	true
<code> \${(10*10) ne 100}</code>	false
<code> \${'a' &lt; 'b'}</code>	true
<code> \${'hip' gt 'hit'}</code>	false
<code> \${4 &gt; 3}</code>	true
<code> \${1.2E4 + 1.4}</code>	12001.4
<code> \${3 div 4}</code>	0.75
<code> \${10 mod 4}</code>	2
<code> \${empty param.Add}</code>	False if the request parameter named Add is null or an empty string.
<code> \${pageContext.request.contextPath}</code>	The context path.
<code> \${sessionScope.cart.numberOfItems}</code>	The value of the numberOfItems property of the session-scoped attribute named cart.
<code> \${param['mycom.productId']}</code>	The value of the request parameter named mycom.productId.
<code> \${header["host"]}</code>	The host.
<code> \${departments[deptName]}</code>	The value of the entry named deptName in the departments map.
<code> \${requestScope['javax.servlet.forward.servlet_path']}</code>	The value of the request-scoped attribute named javax.servlet.forward.servlet_path.
<code> #{customer.IName}</code>	Gets the value of the property IName from the customer bean during an initial request. Sets the value of IName during a postback.

from  
<http://java.sun.com/javaee/5/docs/tutorial/doc/bnahq.html#bnain>



## Predefined Expressions

```
 ${currentUser()}
 ${currentUserGroups()}

from org.camunda.bpm.engine.impl.el.CommandContextFunctionMapper

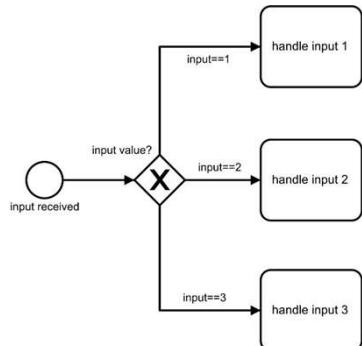
${now()} -> java.util.Date
${dateTime()} -> org.joda.DateTime

from org.camunda.bpm.engine.impl.el.DateTimeFunctionMapper
```

<https://docs.camunda.org/manual/latest/user-guide/process-engine/expression-language/#internal-context-functions>



## Exclusive Gateway (Decision)



```
<bpmn:exclusiveGateway id="exclusiveGateway" name="input value?">
</bpmn:exclusiveGateway>
<bpmn:sequenceFlow id="fLow3" name="input==2" sourceRef="exclusiveGateway" targetRef="handleInput2Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==2}</bpmn:conditionExpression>
</bpmn:sequenceFlow>

<bpmn:sequenceFlow id="fLow2" name="input==1" sourceRef="exclusiveGateway" targetRef="handleInput1Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==1}</bpmn:conditionExpression>
</bpmn:sequenceFlow>

<bpmn:sequenceFlow id="fLow4" name="input==3" sourceRef="exclusiveGateway" targetRef="handleInput3Task">
  <bpmn:conditionExpression xsi:type="bpmn:tFormalExpression">${input==3}</bpmn:conditionExpression>
</bpmn:sequenceFlow>
```



105

## Service Task With UEL Expression



```
<bpmn:serviceTask id="javaService"
    name="My Java Service Task"
    camunda:expression="${printer.printMessage()}" />
<bpmn:serviceTask id="javaService"
    name="My Java Service Task"
    camunda:expression="${printer.printMessage(execution, myVar)}" />
```

105



106

## Camunda Spin Project Example

```
{
  "name" : "jonny",
  "address" : {
    "street" : "12 High Street",
    "post code" : 1234
  }
}
```

Imagine a response from a web service call

Maybe a condition on a sequence flow

Access in an expression:

```
 ${JSON(customer).prop("address").prop("post code").numberValue() == 1234}
```

Access JSON variable value in an expression:

```
 ${customer.jsonPath("$.address.post code").numberValue() == 1234}
```

106



## Camunda Spin Project

### Motivation

- Handle serialized formats such as JSON or XML while interacting with external systems.
- Process such data like parsing, manipulating or mapping from/to Java objects.

### Features

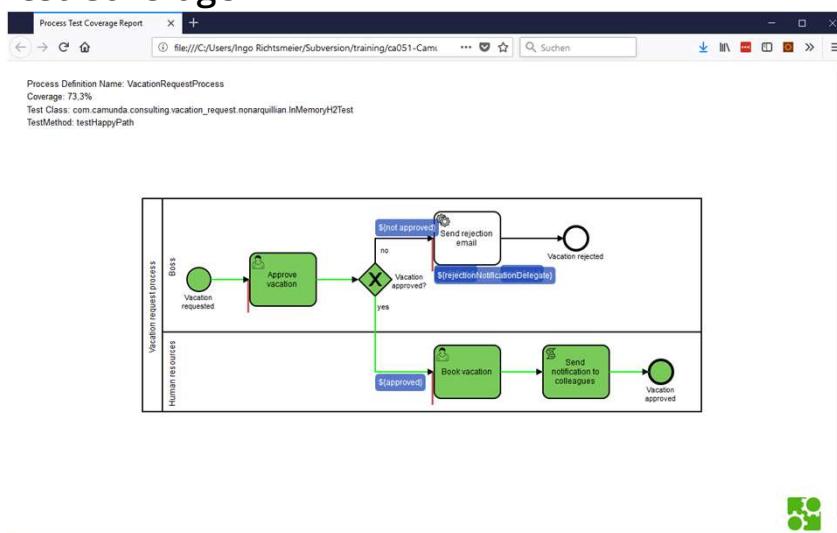
- Provide data format functionality
- Plugged into the engine
- Wrapper for processing data formats like XML and JSON
- Integrated with the engine's data handling functionality
- Designed to be extensible

<https://docs.camunda.org/manual/latest/user-guide/data-formats/>

<https://docs.camunda.org/manual/latest/reference/spin/>



## Process Test Coverage



<https://github.com/camunda/camunda-bpm-process-test-coverage>



## Process Test Coverage

- Coverage per class and per test method

- Maven dependency:

```
<dependency>
  <groupId>org.camunda.bpm.extension</groupId>
  <artifactId>camunda-bpm-process-test-coverage</artifactId>
  <version>0.3.2</version>
  <scope>test</scope>
</dependency>
```

- process engine configuration class:

```
org.camunda.bpm.extension.process_test_coverage.junit.rules.  
ProcessCoverageInMemProcessEngineConfiguration
```

- Usage in Junit:

```
@ClassRule  
@Rule  
public static ProcessEngineRule rule = TestCoverageProcessEngineRuleBuilder.create().build();
```

## Exercise 4

<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks





## Improve JUnit with Simple Mocks

### 1. Make Delegate a Bean with @Component

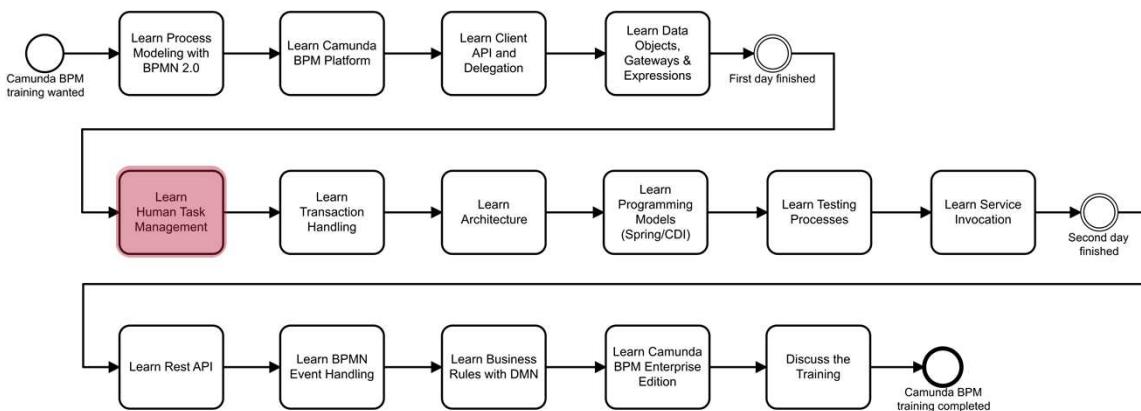
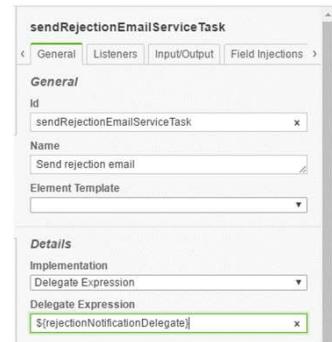
```
@Component("rejectionNotificationDelegate")
public class SendRejectionNotificationDelegate implements JavaDelegate {

    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Send the rejection to the employee...
    }
}
```

### 2. Use Indirection with DelegateExpression in bpmn process

### 3. Use Mocks.register("bean", new MockDelegate()) in the test method

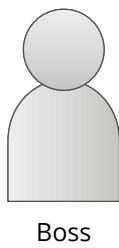
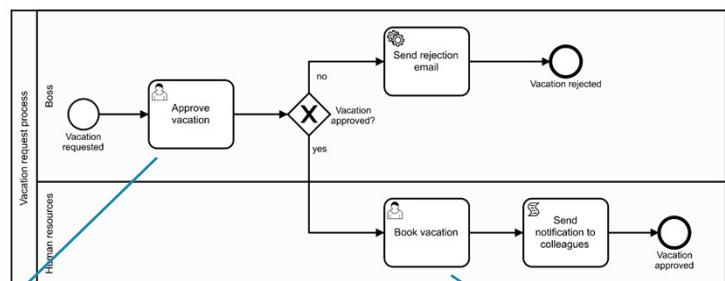
```
@Test
@Deployment(resources = "vacation-request.bpmn")
public void testRejectionVacation() {
    Mocks.register("rejectionNotificationDelegate",
        new LoggerDelegate());
    ...
}
```



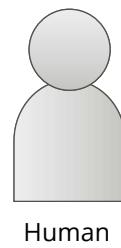


113

## Task Management



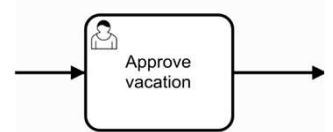
Boss



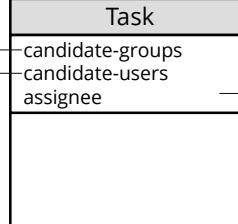
Human Resources

113

## Task Management



create ↓ ↑ complete



claim

114

114

## Tasklist in Camunda BPM

The screenshot shows the Camunda Tasklist interface. On the left, there's a sidebar with 'My Tasks' and 'My Group Tasks (7)'. The main area displays a list of tasks. Three large blue circles with numbers 1, 2, and 3 are overlaid on the screen to guide the user through the process:

- 1**: Shows the task list with items like 'Approve vacation', 'Approve Invoice', and 'Approve Invoice'.
- 2**: Shows a detailed view of a task titled 'Approve vacation' with sub-information like 'Vacation request' and 'Created 2 minutes ago'.
- 3**: Shows an approval form for the vacation request, with fields for 'Employee' (set to 'John'), 'From' (set to '2018-07-17T00:00:00'), 'Until' (set to '2018-07-24T00:00:00'), and an 'Approve?' checkbox.

115

## Groups & Personal Task List (Via Filters)

The screenshot shows the 'Edit filter' dialog in the Camunda Tasklist interface. The dialog has several sections:

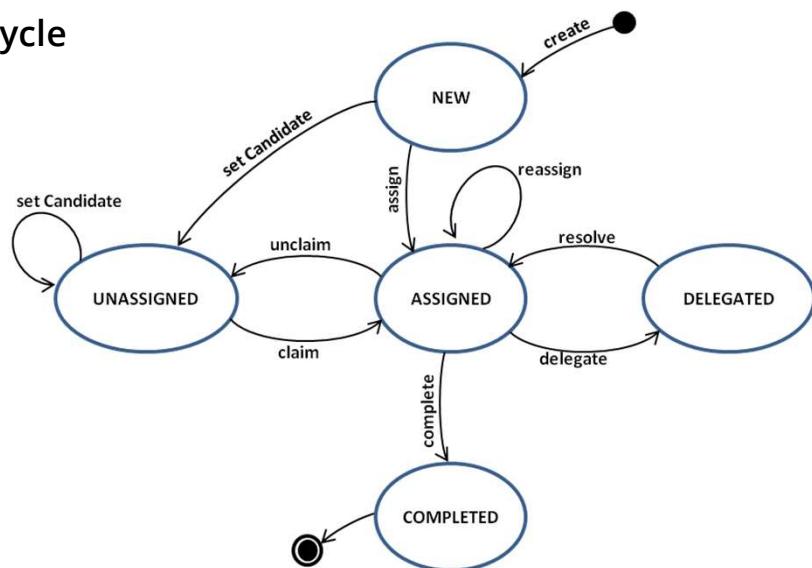
- General**: Contains a 'Criteria' section with two entries: 'Unassigned' under 'Key' and '\$[currentUserGroups()]' under 'Value'.
- Permissions**: Contains a checkbox 'Include assigned tasks' which is unchecked.
- Variables**: An empty section.
- Buttons**: 'Delete filter', 'Close', and a red 'Save' button.

116



117

## Task Lifecycle



117



118

## Task Management API

The screenshot shows the Task Management API interface with the following components:

- Left Sidebar:** Lists "My Tasks", "My Group Tasks (10)", and various "App" and "Pref" items.
- Central Area:** Displays a BPMN diagram for the "Vacation request process".
  - Participants: Boss, Human resources.
  - Start event: "Vacation requested".
  - Activity: "Approve vacation".
  - Decision diamond: "Vacation approved?".
  - Path 1 (no): "Send rejection email" -> End event "Vacation rejected".
  - Path 2 (yes): "Book vacation" -> "Send notification to colleagues" -> End event "Vacation approved".
- Bottom Right:** A detailed view of the "Approve vacation" task.
  - Form tab: "Approve the vacation request".
  - Employee: "Mary".
  - From: "2016-12-20T00:00:00".
  - Until: "2016-12-23T23:59:00".
  - Approve? checkbox.
  - Buttons: "Save" and "Complete".
- Annotations:**
  - A blue arrow points from the sidebar's "taskService.createTaskQuery()" to the BPMN diagram.
  - A blue arrow points from the sidebar's "formService.getTaskFormKey()" to the task form.
  - A blue arrow points from the task form's "taskService.complete()" button to the task form itself.

118



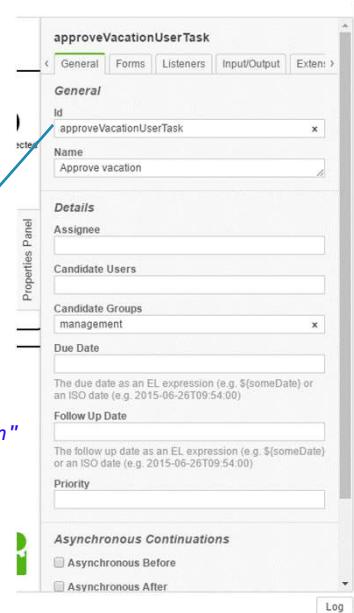
## Key vs. ID vs. Name

```
taskService.complete(task.getId(), variables);
```

- `task.getId():`
- `task.getTaskDefinitionKey():`
- `task.getName():`

Task Instance ID  
BPMN User Task ID  
BPMN User Task Name

```
<bpmn:userTask id="approveVacationUserTask" name="Approve vacation"  
camunda:candidateGroups="management">  
</bpmn:userTask>
```



## Personal and Group Task Lists

```
TaskService taskService = processEngine.getTaskService();

List<Task> personalTaskList = taskService.createTaskQuery()
    .taskAssignee("john").list();

List<Task> groupTaskList = taskService.createTaskQuery()
    .taskCandidateUser("john").list();

groupTaskList.addAll(taskService.createTaskQuery()
    .taskCandidateGroup("marketing").list());
```



121

## Task Query & Completion

```
TaskService taskService = processEngine.getTaskService();

List<Task> personalTaskList = taskService.createTaskQuery()
    .taskAssignee("demo").list();

Task task = personalTaskList.get(0);

Map<String, Object> variables = new HashMap<String, Object>();
variables.put("approved", false);

taskService.complete(task.getId(), variables);
```

121



122

## Tasks in Camunda BPM Assert Library

- Assertions with fluent api:

```
assertThat(processInstance).task().hasCandidateGroup("management");
```

- Helper methods:

```
complete(task());
```

```
complete(task("bookVacationUserTask"));
```

```
public static TaskQuery taskQuery()
```

122



123

## Task Listener

### Typical Use Cases:

- Send notification e-mail
- Task modifications, e.g. priority, due date, display name
- Vacation substitute
- Create task in third-party application

### Available events:

create  
update  
assignment  
complete  
delete

```
public class TaskCreationListener implements TaskListener {  
    @Override  
    public void notify(DelegateTask delegateTask) {  
        // send a mail to manager  
    }  
}
```

123



124

## Exercise 5



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

124

## Task Forms

The screenshot illustrates a vacation approval process. On the left, a 'Start process' form is shown with fields for 'Your Name', 'Start date', 'Last day of vacation', and a 'Back' button. A blue arrow points from this form to a BPMN diagram. The diagram shows a 'Vacation requested' event leading to a user task 'Approve vacation'. This task is assigned to 'Boss'. From the task, a decision diamond 'Vacation approved?' branches into 'no' (leading to 'Send rejection email') and 'yes' (leading back to the start process). Below the diagram is a detailed view of the 'Approve vacation' task, which includes a form key 'embedded app.forms/embedded/task-form.html', form fields, and a claim button.

125

## Task Forms in Camunda Tasklist

### Generic form

This screenshot shows the configuration of a generic form for a 'Second User Task'. It includes sections for 'Form Key', 'Form Fields', and 'Listeners'. The 'Form Fields' section lists variables: 'Employee' (type: String, value: 'Mary'), 'From' (type: Date, value: '2016-12-20T00:00:00'), 'Until' (type: Date, value: '2016-12-23T23:59:00'), and 'Approve?' (checkbox). Buttons for 'Save' and 'Complete' are at the bottom.

### Generated form

This screenshot shows a generated task form titled 'Start process'. It contains fields for 'Firstname', 'Lastname', and 'Date of Birth'. At the bottom are 'Back' and 'Start' buttons.

### Embedded form

This screenshot shows an embedded task form titled 'Approve vacation'. It displays a vacation request for 'Employee' (Mary), 'From' (2016-12-20T00:00:00), 'Until' (2016-12-23T23:59:00), and an 'Approve?' checkbox. Buttons for 'Save' and 'Complete' are at the bottom.

### External form

This screenshot shows an external task form titled 'Review Template'. It displays a review template with fields for 'Address' (Potsdamer Platz 100, Berlin, Germany) and 'Comments'. Buttons for 'Save' and 'Complete' are at the bottom.

126



## Generic Task Forms

- `formKey=""`
- Completely generic, shows all process variables after load
- Add variables manually on demand
- Edit existing variables

Second User Task  
Generic Form Process

Set follow-up date Set due date Add groups Demo Demo

Form History Diagram Description

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

Business Key	Name	Type	Value
Add a variable +	Variable name	Variable type	Value
Remove ×			

Load Variables ⓘ Complete

127



## Generated Task Forms (From Form Fields)

- Form Data Metadata provided by BPMN 2.0 XML
- Rendered in task list

Employee  
Jim

Start date  
09/11/2016

Last day of vacation  
11/11/2016

Approved by  
Peter Meter

bookVacationUserTask

Forms Form Type Form Data

Form Fields employee from until approvedBy

Form Field ID approvedBy Type string Label Approved by Default Value

Validation Add Constraint + Name readonly Config

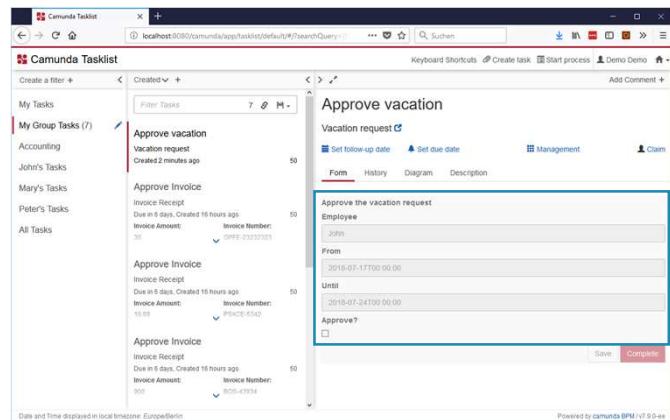
Properties

128



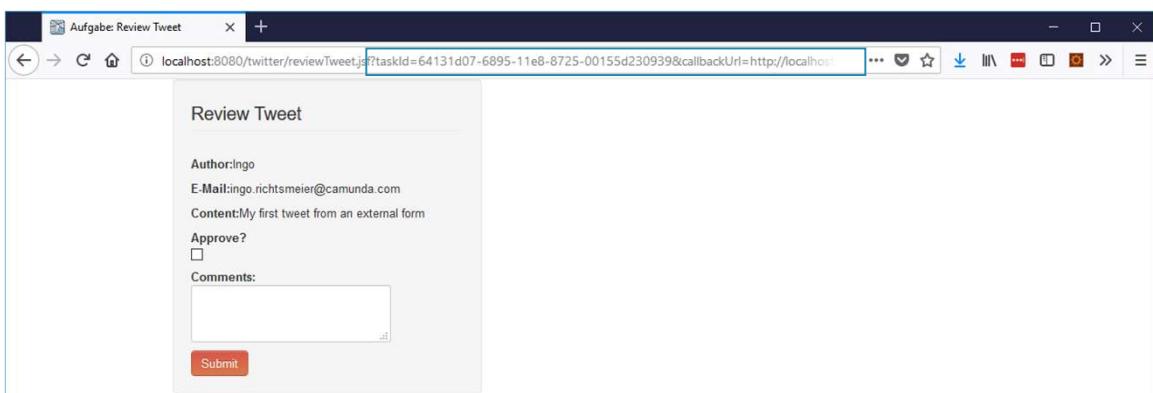
## Embedded Task Forms

- formKey="embedded:app:approve-vacation.html"
- HTML-Form provided by Process Application (HTML-File)
- Mix HTML and Angular-JS JavaScript
- Rendered in task list



## External Task Form

- formKey="app:reviewTweet.jsf"
- Params taskId and callbackURL needed





## Embedded Task Forms Form SDK

- *Form handling*: Attach to a form existing in the DOM or load a form from a URL.
- *Variable handling*: Load and submit variables used in the form.
- *Script handling*: Execute custom JavaScript in Forms
- *AngularJS Integration*: The Forms SDK optionally integrates with AngularJS to take advantage of AngularJS form validation and other AngularJS goodies.

cam-variable-name: Bind input to the model

cam-variable-type: Type of process variable

### Form templates are generated from the archetypes

<https://docs.camunda.org/manual/latest/reference/embedded-forms/>



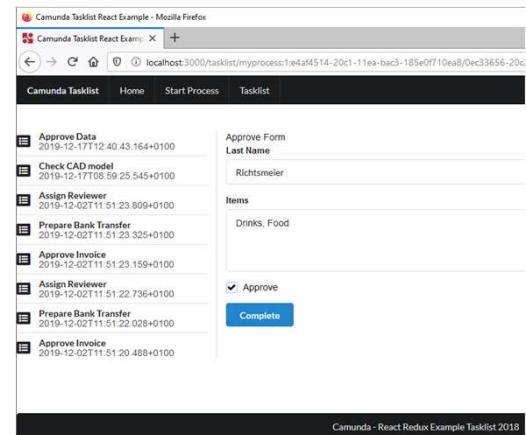
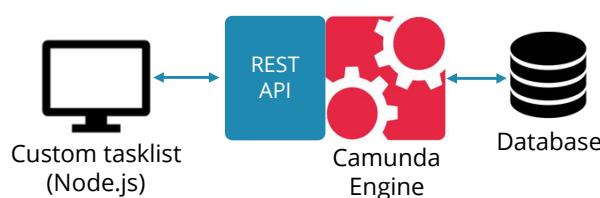
## Simple Form Example

```
<strong>Here you can request your vacation</strong>

<form class="form-horizontal">
  <div class="control-group">
    <label class="control-label">Your Name</label>
    <div class="controls">
      <input type="text"
        cam-variable-name="employee"
        cam-variable-type="String"
        required
        class="form-control" />
    </div>
  </div>
  <div class="control-group">
    <label class="control-label">Start date</label>
    <div class="controls">
      <input type="text"
        cam-variable-name="from"
        cam-variable-type="Date"
        required
        class="form-control" />
    </div>
  </div>
  <div class="control-group">
    <label class="control-label">Last day of vacation</label>
    <div class="controls">
      <input type="text"
        cam-variable-name="until"
        cam-variable-type="Date"
        required
        class="form-control" />
    </div>
  </div>
  <script cam-script type="text/form-script">
    // custom JavaScript goes here
  </script>
</form>
```

## Custom Tasklists

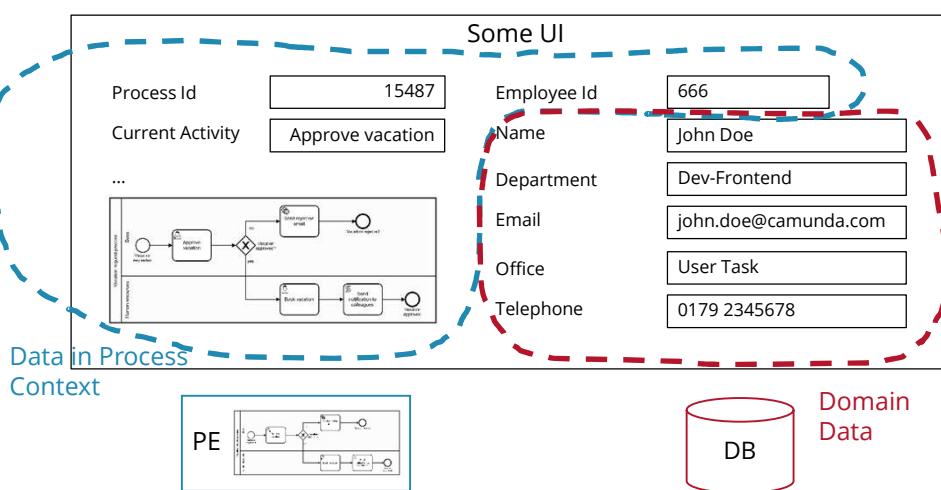
- Leverage the REST API
- Can use any technology (Vue.js, React, Angular, ...)
- Be creative with the Form Key



<https://blog.camunda.com/post/2018/02/custom-tasklist-examples/>

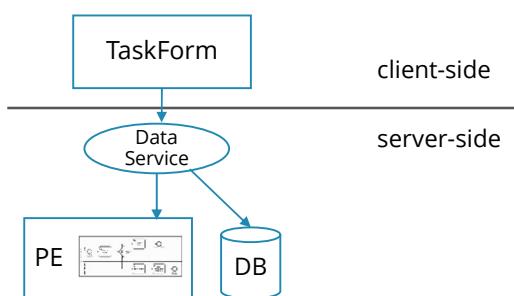
<https://github.com/camunda-consulting/code/tree/master/snippets/camunda-tasklist-examples/>

## Typical: Process Data and External Data Are Mixed





## How to Call Data Service



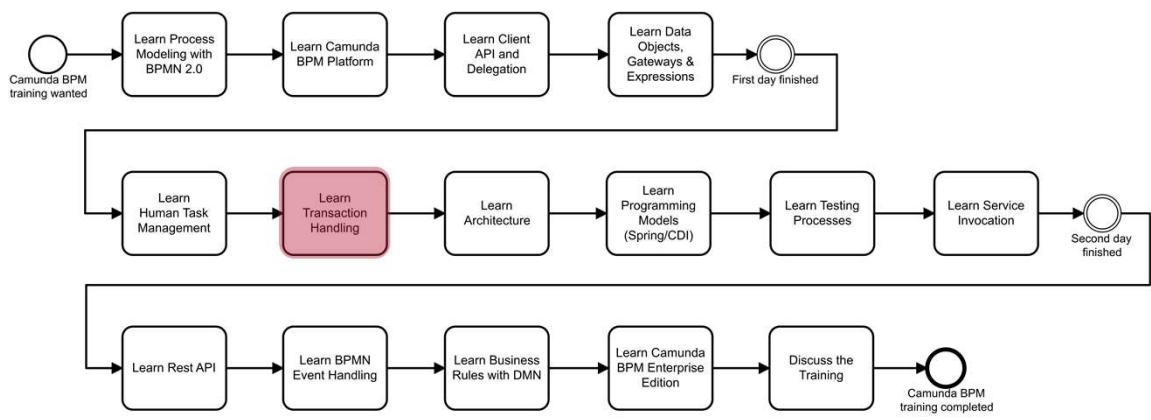
### Composed Service

- The UI calls a combined service
- This service calls the Data Service and the Process Service
- TX possible

## Exercise 6

<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

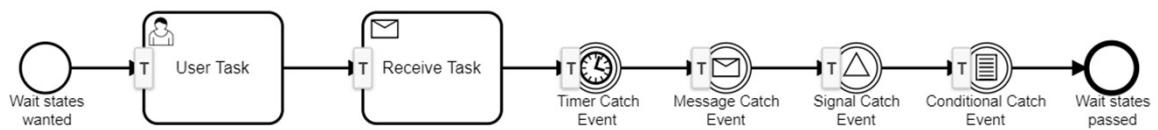




137

138

## Wait States



Also called:

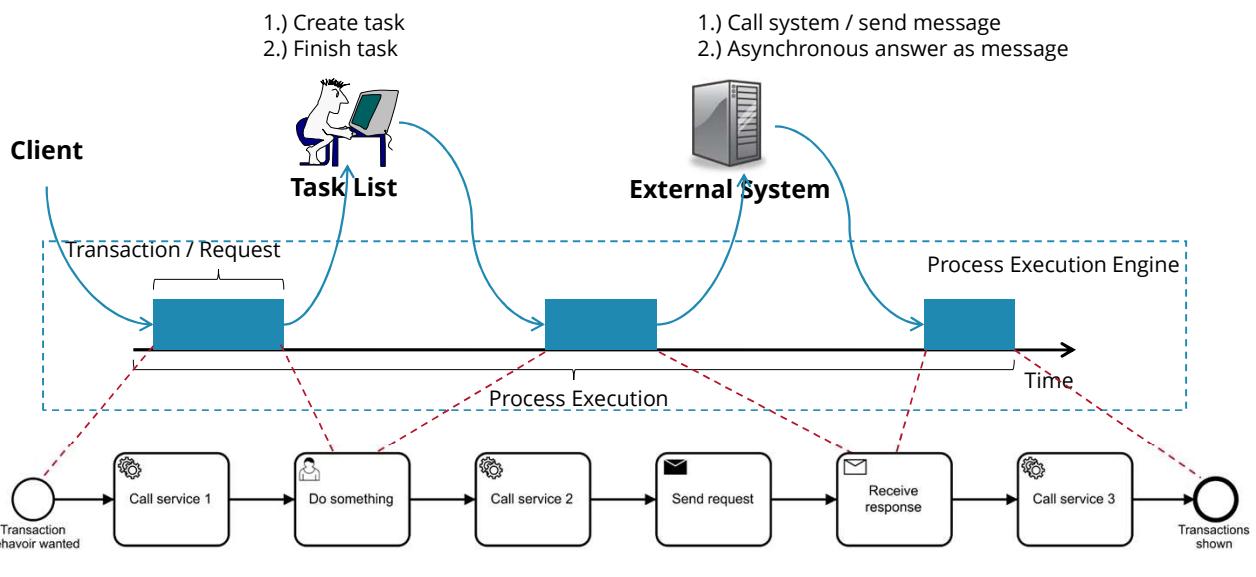
- Transaction Boundaries
- Save Points

<https://github.com/camunda/camunda-modeler-plugins/blob/master/camunda-transaction-boundaries-plugin>

138



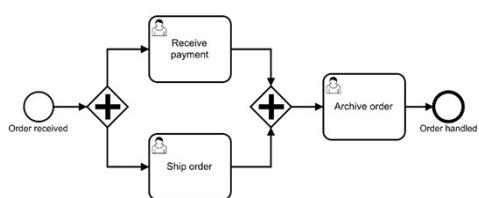
## Process Engine and Transactions



## Parallel Gateway – Engine Internals

### Splitting gateway:

- For every outgoing sequence flow:
- Move the token until it can't go any further
- Then commit transaction



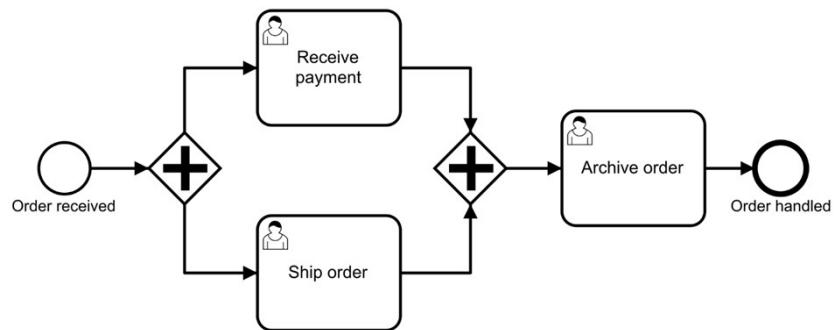
### Joining gateway:

- An incoming sequence flow token arrives:
- Check number of tokens waiting at joining gateway
- If one or more tokens is missing
  - Wait and
- If no other token is active in the transaction, commit
- If token from last sequence flow arrives, continue



141

## How Many Transactions?



141



142

## How Many Transactions?



142



143

## How Many Transactions? (The Other Way Around)

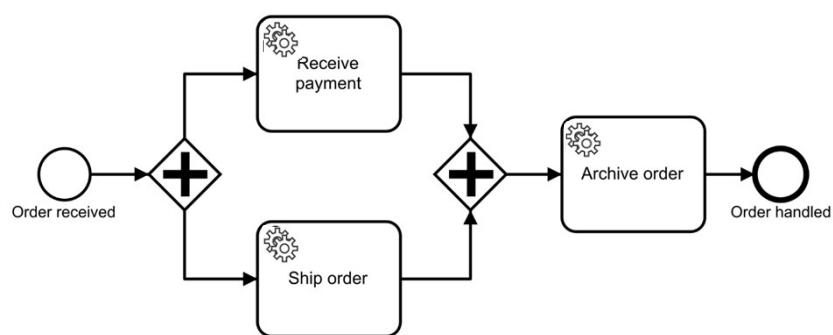
Hidden solution

143



144

## How Many Transactions?



What if all tasks were Service Tasks?

144



145

## How Many Transactions?

Hidden solution

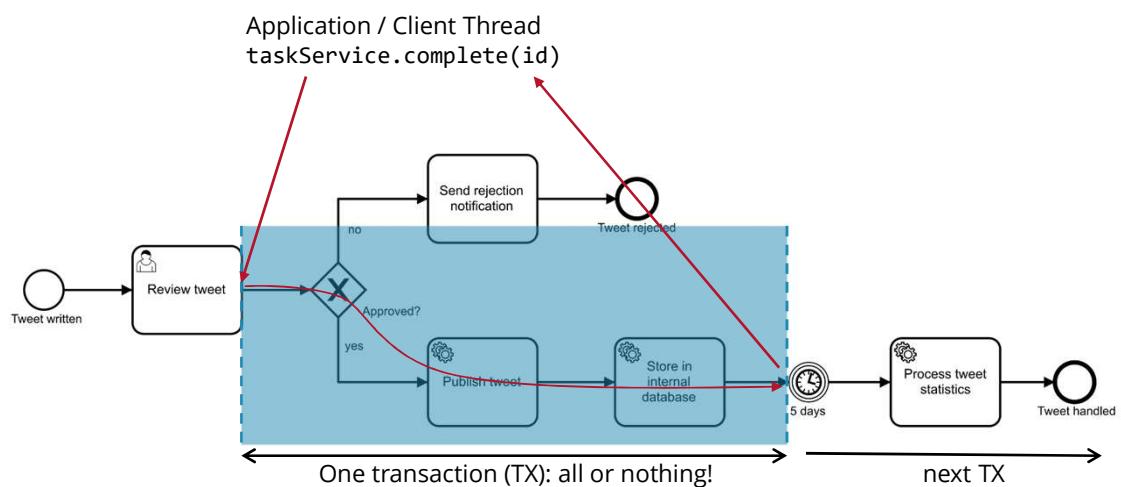
What if all tasks were Service Tasks?

145



146

## Transaction Boundaries



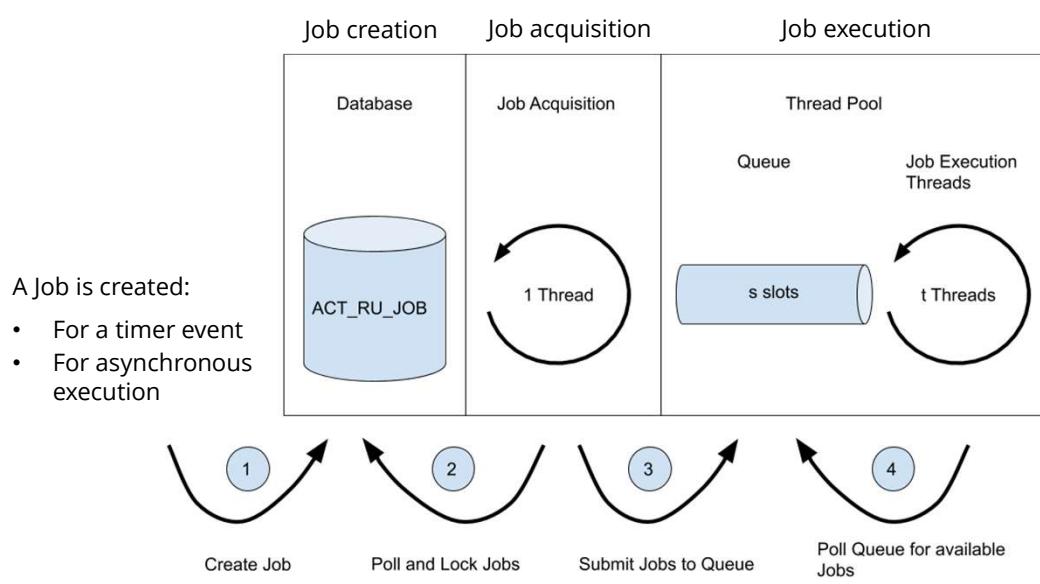
146

## Demo



147

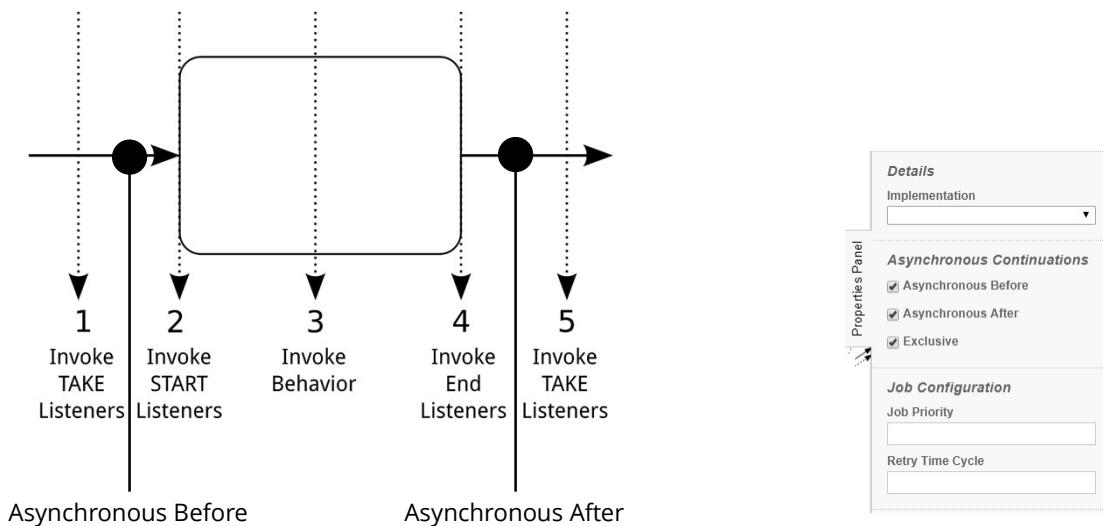
## The Job Executor



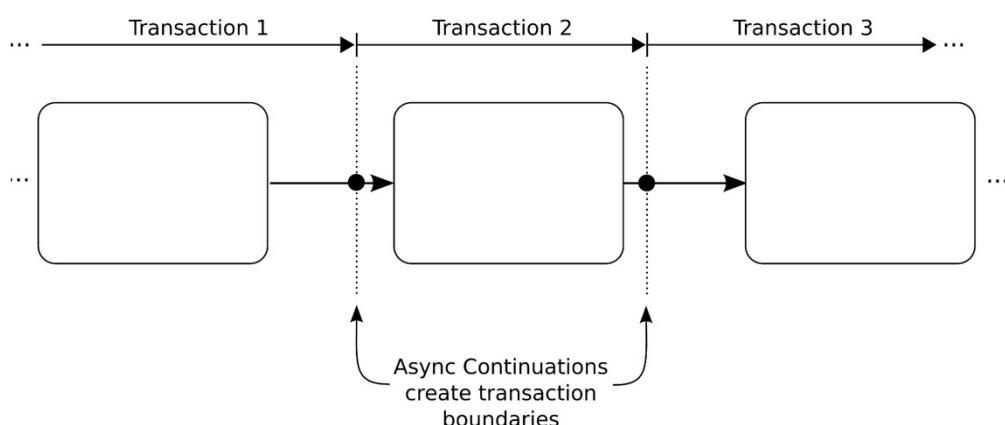
148



## Asynchronous Continuation: Execution of a BPMN Element



## Asynchronous Continuation = Transaction Boundaries/Save Points

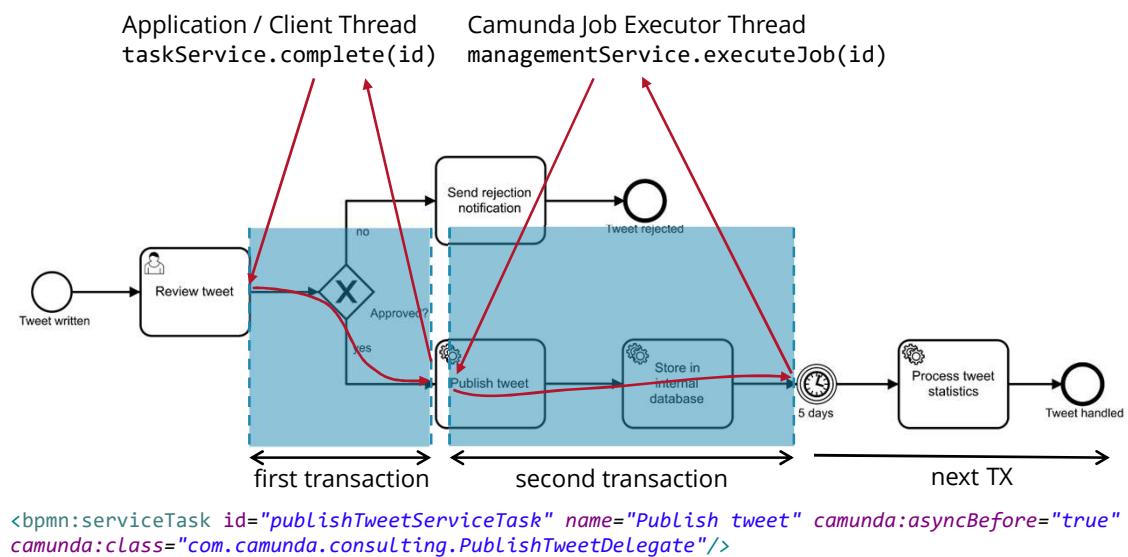


```
<bpmn:serviceTask id="publishTweetServiceTask" name="Publish tweet" camunda:asyncBefore="true" camunda:asyncAfter="true" camunda:class="com.camunda.consulting.PublishTweetDelegate"/>
```



151

## Transaction Boundaries



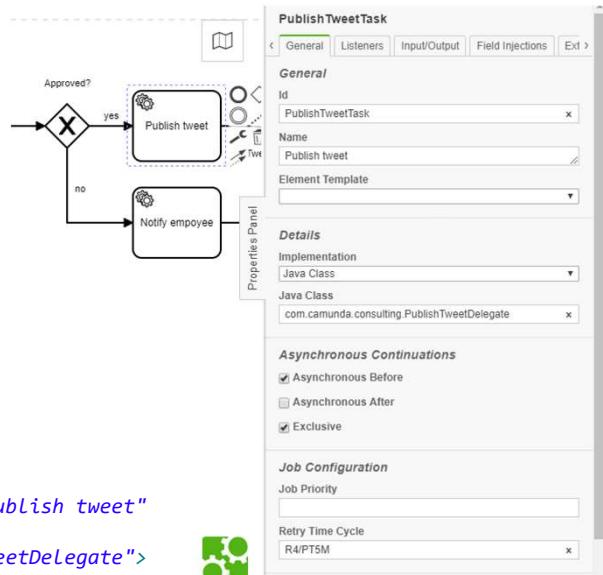
151



152

## Retry Strategy

```
<bpmn:serviceTask id="PublishTweetTask" name="Publish tweet" camunda:asyncBefore="true" camunda:class="com.camunda.consulting.PublishTweetDelegate">
<bpmn:extensionElements>
<camunda:failedJobRetryTimeCycle>R4/PT5M</camunda:failedJobRetryTimeCycle>
</bpmn:extensionElements>
</bpmn:serviceTask>
```



152



## Async Continuation in JUnit Test

- In detail:

```
List<Job> jobList = managementService
    .createJobQuery()
    .processInstanceId(processInstanceId)
    .list();
assertThat(jobList).hasSize(1);
Job job = jobList.get(0);
managementService.executeJob(job.getId());
```

- camunda-bpm-assert short cut:

```
assertThat(processInstance).isWaitingAt("PublishTweetTask");
execute(job());
```

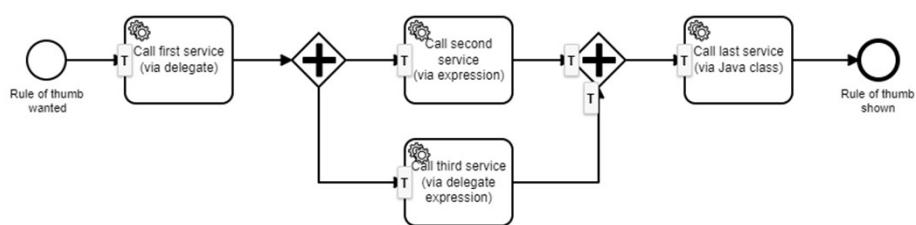
- In case of parallel flows:

```
assertThat(processInstance).isWaitingAt("ParallelTask1");
Job job = jobQuery().activityId("ParallelTask1").singleResult();
execute(job);
```



## Rule of Thumb for Save Points

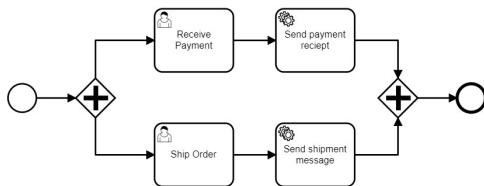
- Before each service task that is not an external service task
- Before each joining parallel gateway





## Detailed Best Practices for Save Points

- Before start event of process instance
- After user task
- Before invocations of external systems especially over network
- After non-transactional, non-idempotent side-effects
- After expensive external computation
- Before parallel joins like parallel and inclusive gateways or at the end of each instance of parallel multi-instance activities



### NOT

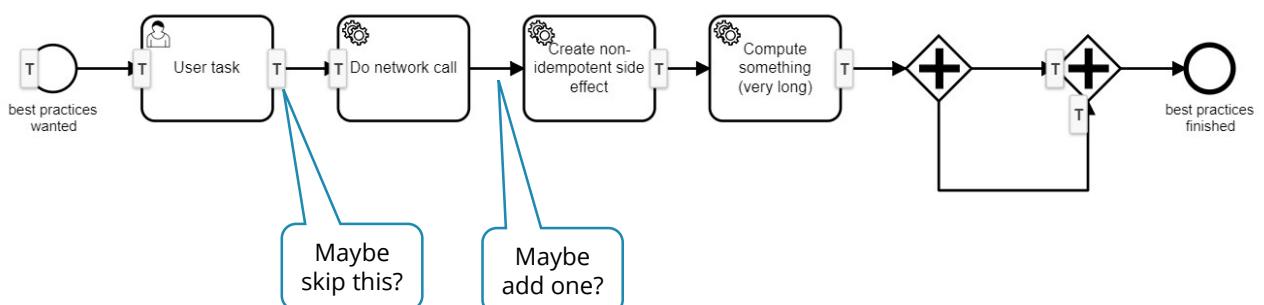
- Before tasks that wait anyway (user tasks, receive tasks, external service tasks, catching events like timer, message, signal)
- Before XOR & Event-based gateways
- Before splitting AND & OR gateways

### UNLESS

- Listeners, beans or complex expressions, e.g. using SPIN, that could fail are involved



## Best Practices for Save Points



Add save points automatically:

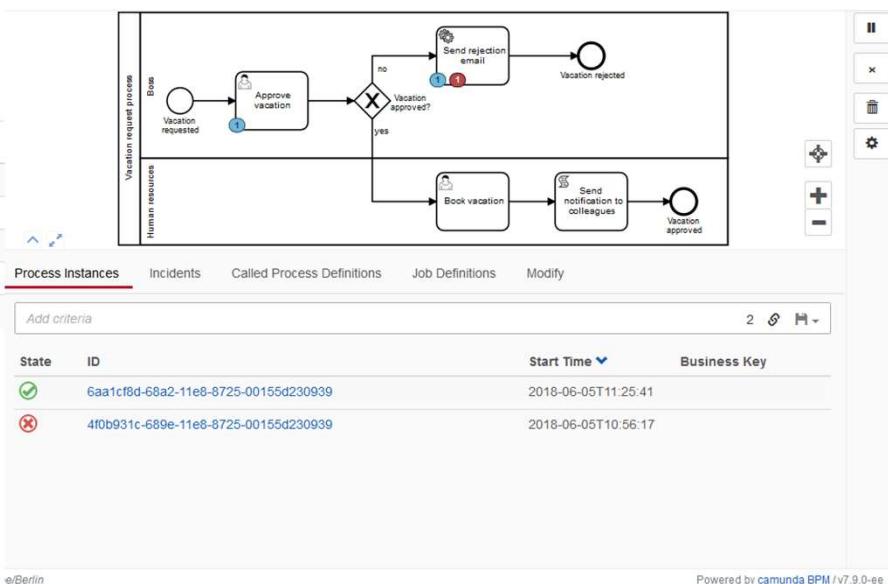
<https://github.com/camunda-consulting/code/tree/master/snippets/engine-plugin-add-save-points>  
<https://github.com/camunda-consulting/code/tree/master/snippets/async-joins>



## Incidents Overview

5 process definitions deployed

State	Incidents	Running Instances
✓	0	7
✓	0	13
✗	1	7
✓	0	1
✗	1	2



e/Berlin

Powered by camunda BPM / v7.9.0-ee

157



## Incidents Details

Camunda Cockpit | Vacation request

Information Filter

Instance ID: 4f0b931c-689e-11e8-8725-00155d230939  
Business Key: null  
Definition Version: 1  
Definition ID: VacationRequestProcess:1:89192dff...  
Definition Name: VacationRequestProcess  
Definition Key: Vacation request  
Tenant ID: null  
Deployment ID: 890fb1b-6893-11e8-8725-00155d2...  
Super Process Instance ID: null  
Related:

- Migration

Variables Incidents Called Process Instances User Tasks External Tasks Modify

Message	Timestamp	Activity	Cause Process Instance ID	Root Cause Process Instance ID	Type	Action
Could not send notification...	2018-06-05T11:24:01	Send rejection email			Failed Job	<button>retry</button>

Date and Time displayed in local timezone: Europe/Berlin

Powered by camunda BPM / v7.9.0-ee

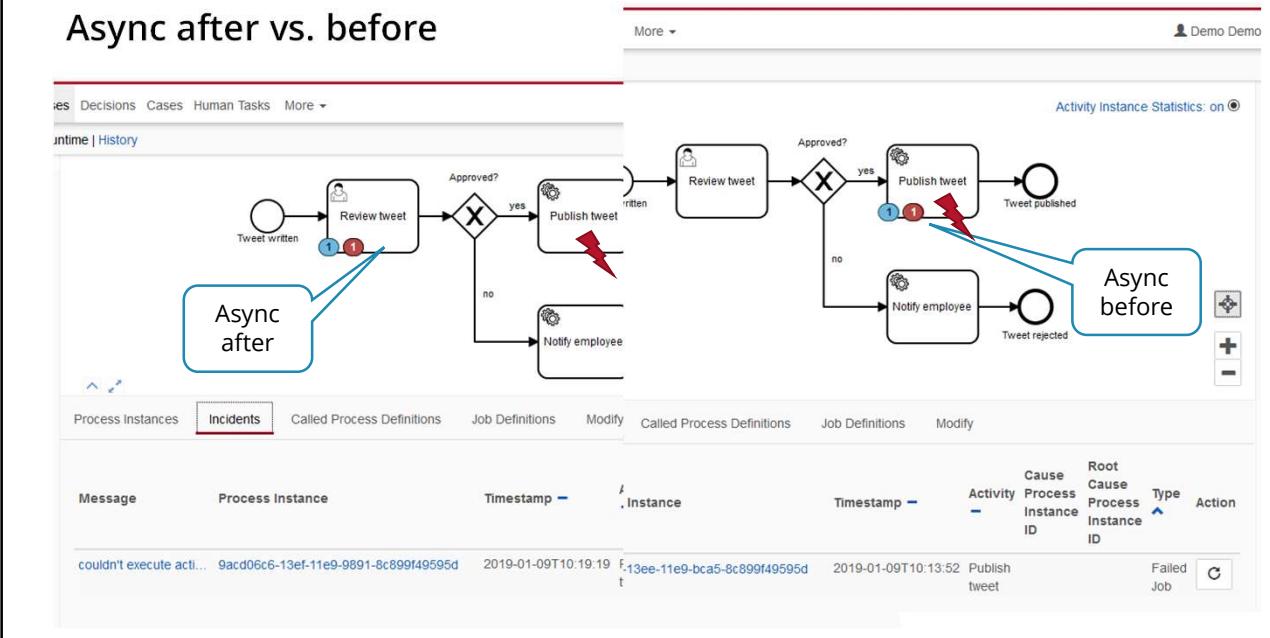
Retry failed Job

158



159

## Async after vs. before

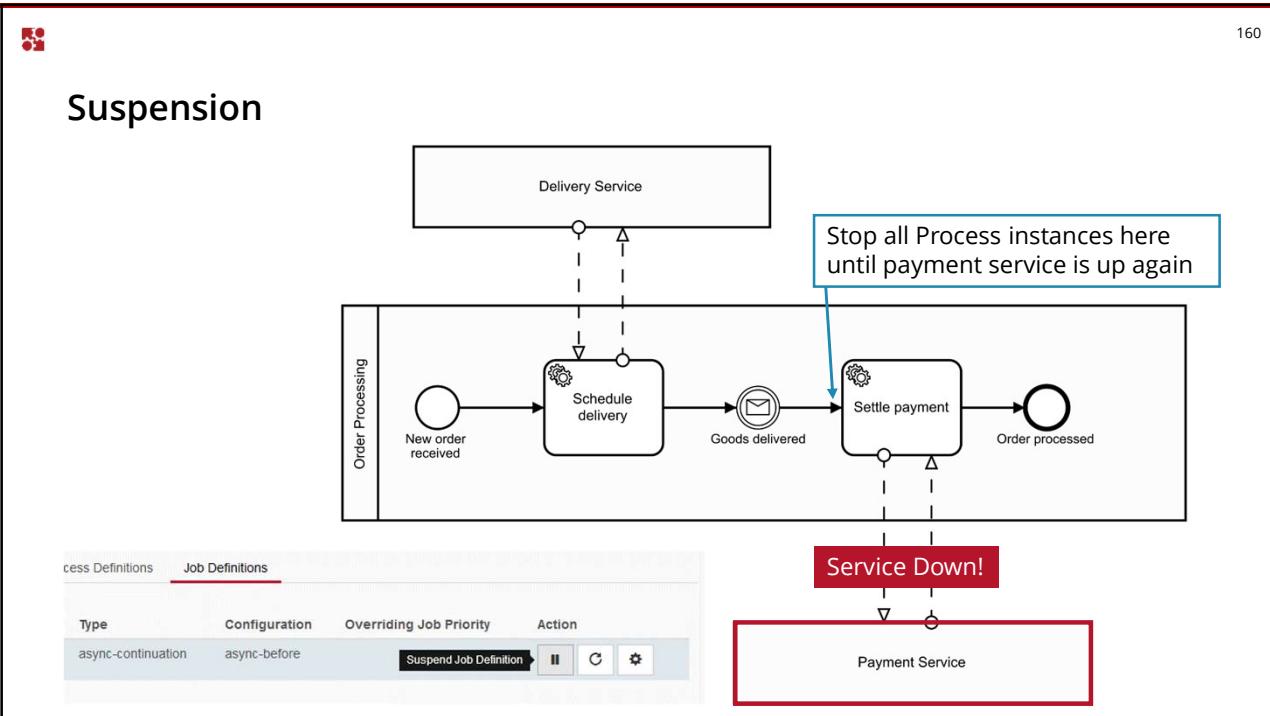


159



160

## Suspension



160

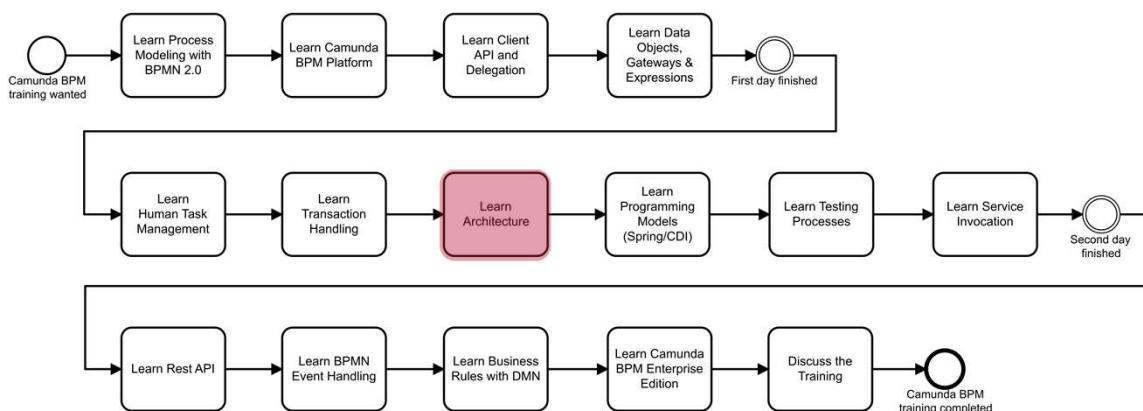


## Exercise 7



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

161

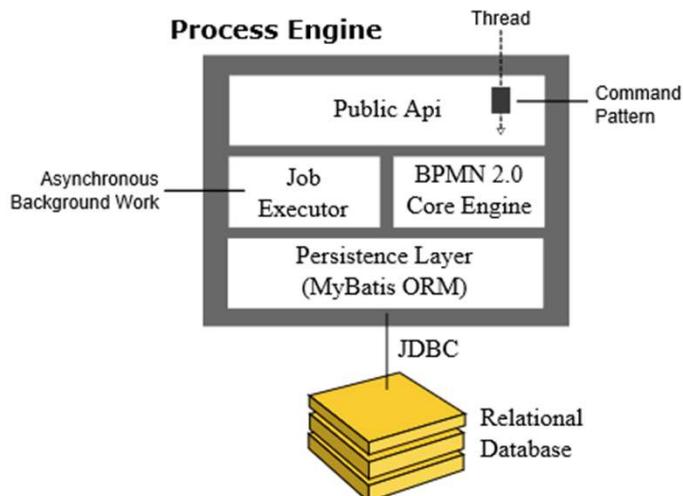


162



163

## Process Engine Architecture

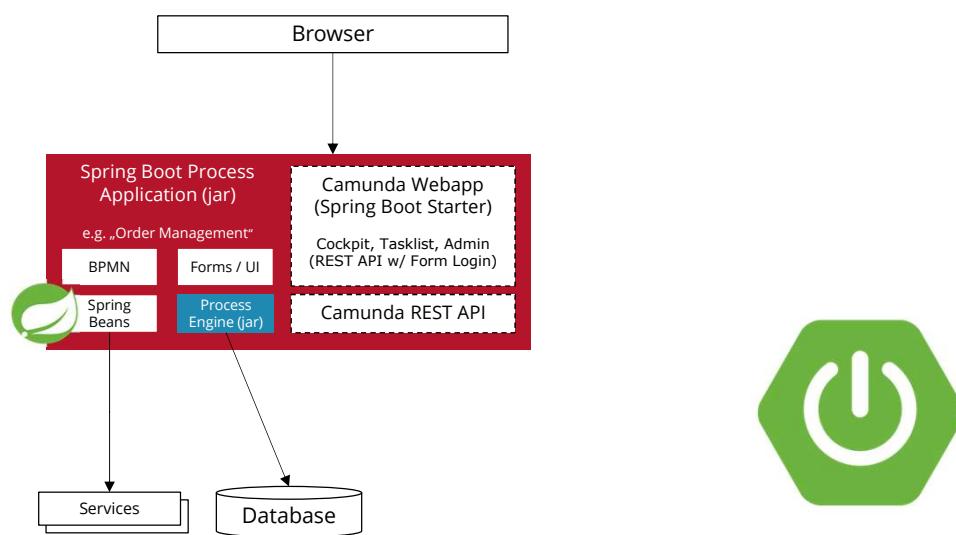


163



164

## Spring Boot: Embedded Process Engine, Webapp & REST API



164



165

## Start an Engine in Pure Java

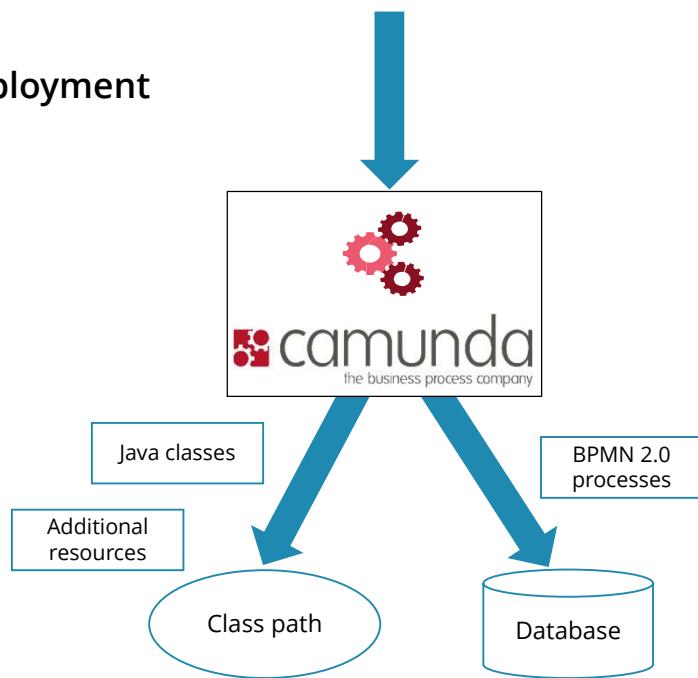
```
ProcessEngine processEngine =  
    ProcessEngineConfiguration.createStandaloneInMemProcessEngineConfiguration()  
        .setDatabaseSchemaUpdate(ProcessEngineConfiguration.DB_SCHEMA_UPDATE_FALSE)  
        .setJdbcUrl("jdbc:h2:mem:my-own-db;DB_CLOSE_DELAY=1000")  
        .setJobExecutorActivate("true")  
        .buildProcessEngine();  
  
processEngine.getRepositoryService().createDeployment()  
    .addClasspathResource("jax.bpmn").deploy();
```

165



166

## Process Deployment



166



167

## Deploying Programmatically With Java

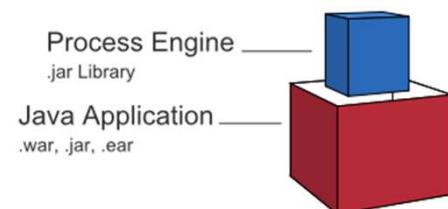
```
repositoryService.createDeployment()  
    .addClasspathResource("process.bpmn")  
    .enableDuplicateFiltering(true)  
    .deploy();
```

167

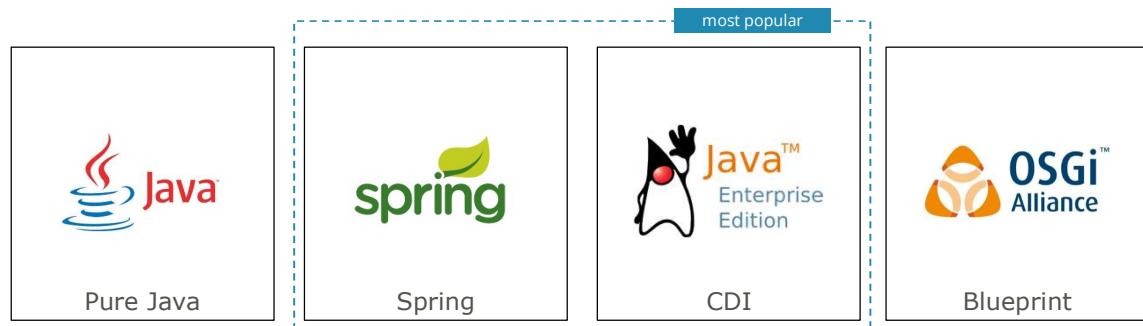


168

## Embedded Process Engine



Various environments possible:



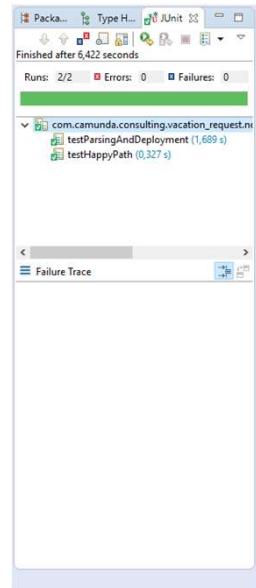
Basic question: Who controls the lifecycle of the engine (startup / shutdown)?

168



## Example: Unit Test

```
import static org.camunda.bpm.engine.test.assertions.ProcessEngineTests.*;  
  
public class VacationRequestProcessTest {  
  
    @Rule  
    public ProcessEngineRule rule = new ProcessEngineRule();  
  
    @Test  
    @Deployment(resources = "vacation-request.bpmn")  
    public void testHappyPath() {  
        ProcessInstance processInstance = runtimeService()  
            .startProcessInstanceByKey("PROCESS_DEFINITION_KEY");  
        assertThat(processInstance).isWaitingAt("approveVacationUserTask");  
        complete(task(), withVariables("approved", true));  
        assertThat(processInstance).isWaitingAt("bookVacationUserTask");  
        complete(task());  
        assertThat(processInstance).isEnded();  
    }  
}
```



## Process Engine as a Spring Component

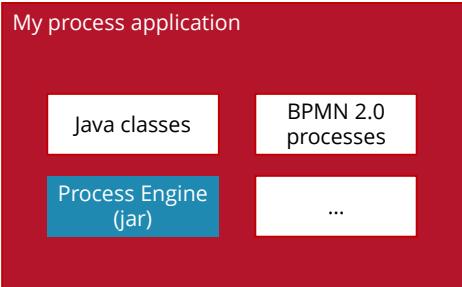
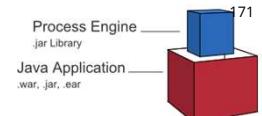
```
ClassPathXmlApplicationContext applicationContext =  
    new ClassPathXmlApplicationContext("/spring-context.xml");  
ProcessEngine processEngine = (ProcessEngine) applicationContext.getBean("processEngine");  
  
processEngine.getRepositoryService().createDeployment()  
    .addClasspathResource("vacation-request.bpmn").deploy();  
  
processEngine.getRuntimeService().startProcessInstanceByKey("vacation-request-process");  
  
Task task = processEngine.getTaskService().createTaskQuery().taskAssignee("john").singleResult();  
  
//Simulate task completion  
processEngine.getTaskService().complete(task.getId());
```

Sample Spring Configuration:

<https://docs.camunda.org/get-started/spring/embedded-process-engine/>



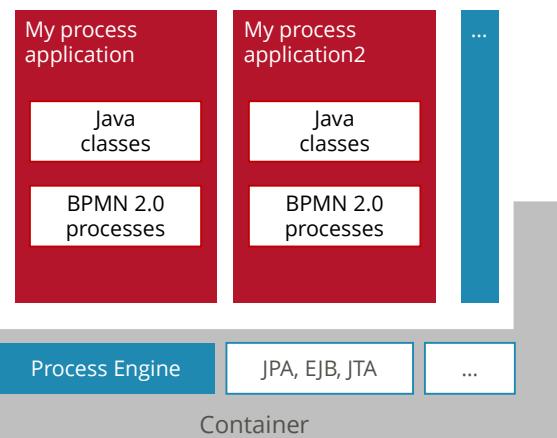
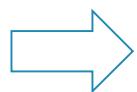
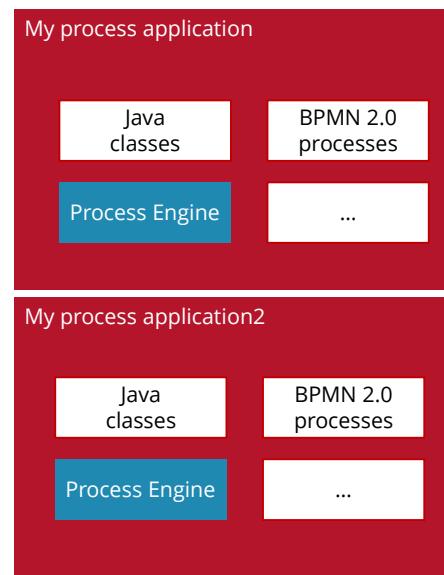
## One Process Application



171

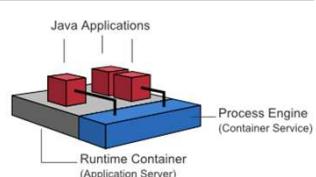
172

## Multiple Process Applications



Supported on:

Tomcat, JBoss AS/EAP, Wildfly AS,  
WebSphere AS and WebLogic



172



## Camunda Process Application: processes.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">

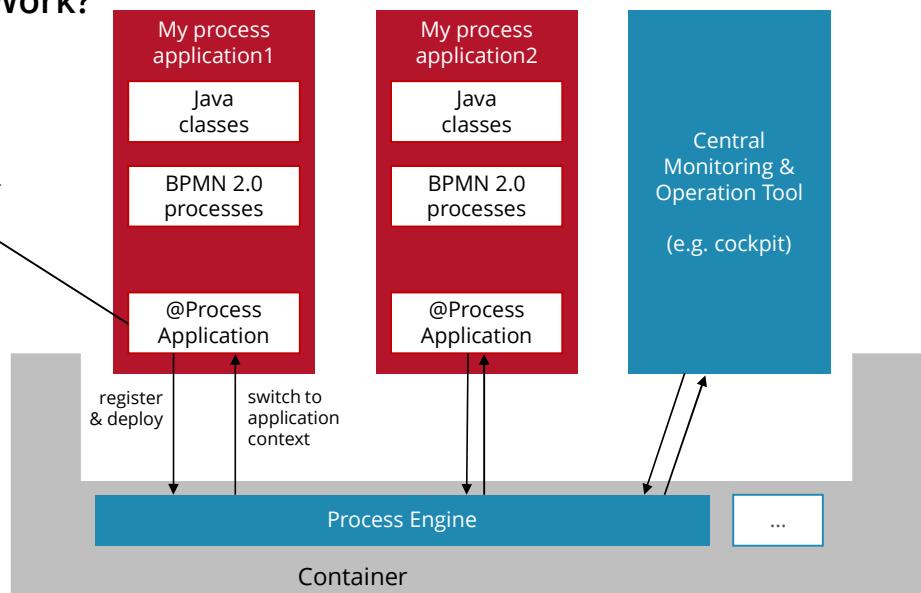
    <process-archive>
        <process-engine>default</process-engine>
        <properties>
            <property name="isDeleteUponUndeploy">false</property>
            <property name="isScanForProcessDefinitions">true</property>
        </properties>
    </process-archive>

</process-application>
```



## How Does This Work?

- Might be
- ServletContextListener
  - @Startup-EJB
  - Spring-Bean
  - ...



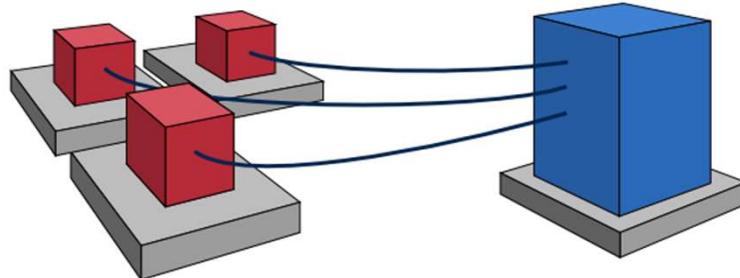


175

## Standalone / Remote Process Engine Server

Remote Applications  
Communication via Rest Webservices

Standalone  
Process Engine Server



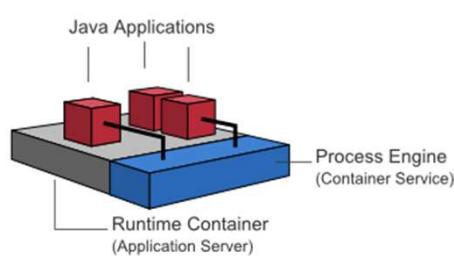
175



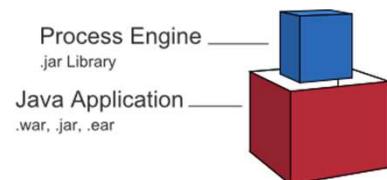
176

## Process Engine Modes

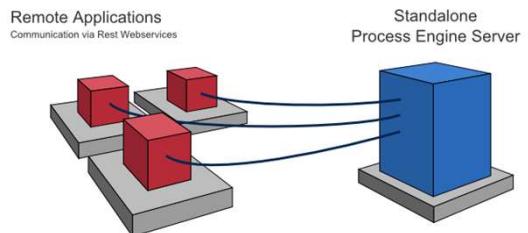
Shared, Container Managed  
Process Engine



Embedded Process Engine

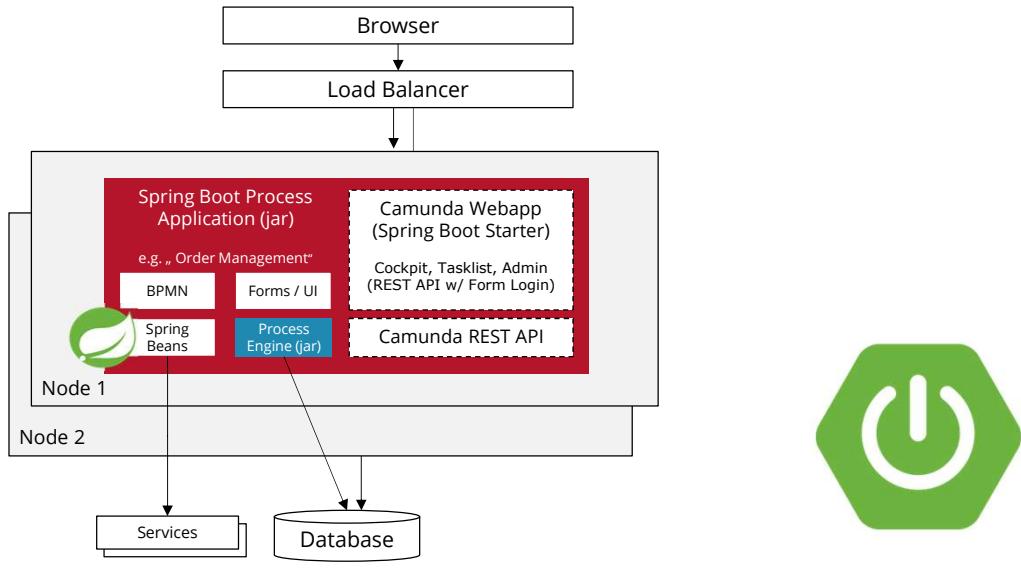


Remote Process Engine



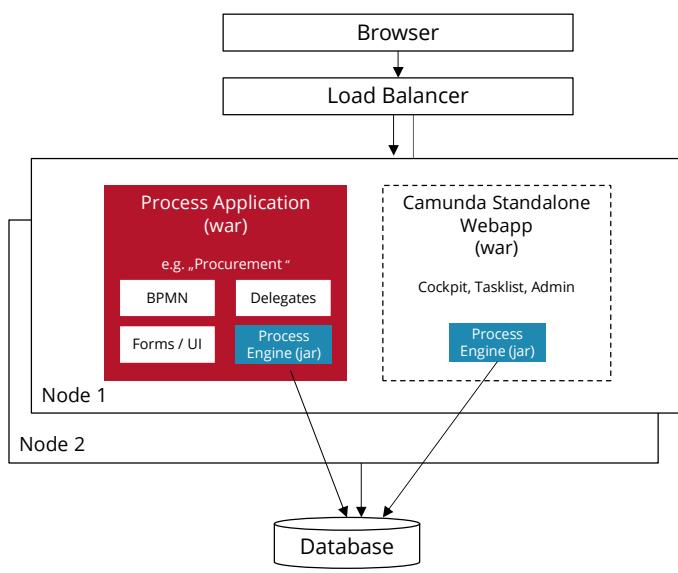
176

## Spring Boot: Embedded Process Engine, Webapp & REST API



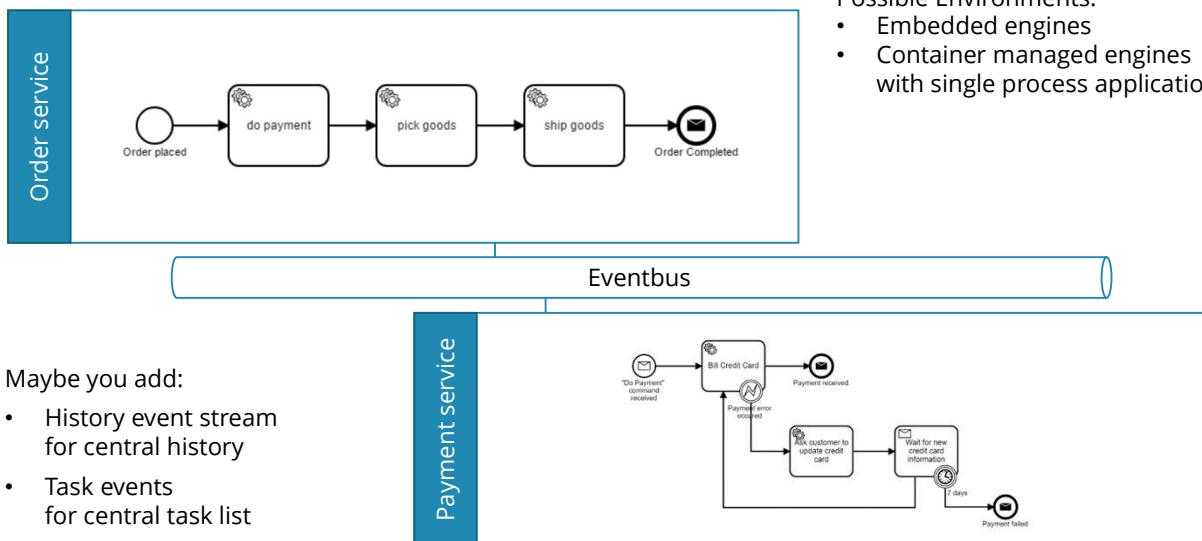
177

## Example Architecture With Embedded Process Engines



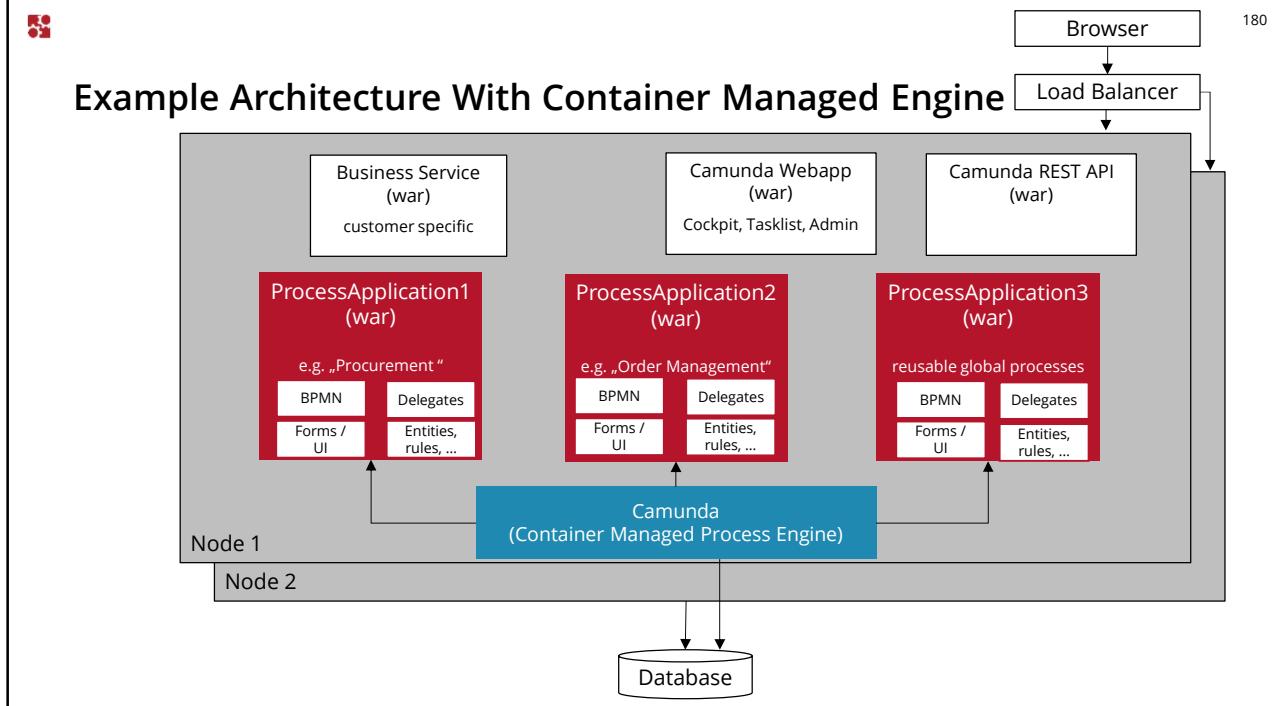
178

## Orchestration With Microservices



179

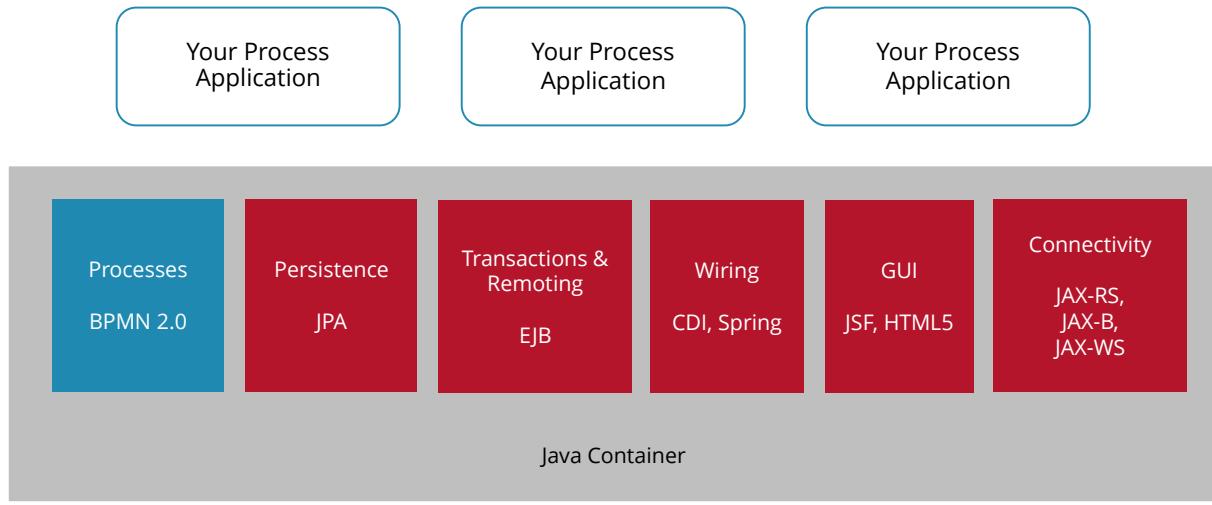
## Example Architecture With Container Managed Engine



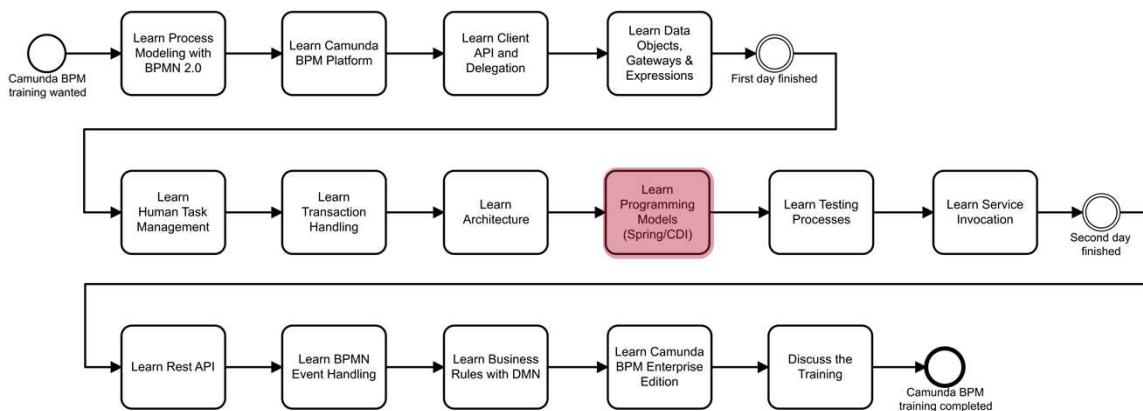
180



## Process as „First Class Citizen“ in Development Stack



181



182



## Camunda BPM Provides

### Spring Integration

- Control lifecycle of process engine via Spring
- Inject engine and services via Spring
- Call named beans via Expression Language (works with container managed engine too)
- Limited TX integration in container managed engine on Tomcat (without JTA)

### CDI Integration

- Inject engine and services via CDI
- Call named beans via Expression Language (works with container managed engine too)
- Contextual Process Implementation and CDI Events (see next slide)



## Obtaining Engine Services Through Dependency Injection

Inject Process Engine Services  
into Spring / CDI / EJB Beans

```
@Inject  
private RuntimeService runtimeService;  
  
@POST  
@Consumes(MediaType.APPLICATION_JSON)  
public void bookingRequest(Booking booking) {  
    runtimeService.startProcessInstanceByMessage("bookingRequest");  
}
```



## Calling Your Names

The `@Component` qualifier allows to give a bean an Expression Language name:

```
@Component  
public class SimpleGreeter implements Greeter,  
Serializable { ... }
```

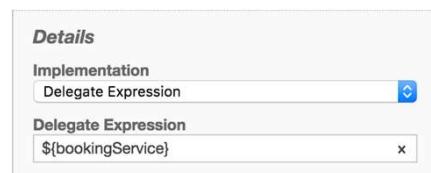
The EL-Name of this bean is "simpleGreeter". This allows us to invoke the bean from an Camunda BPM process:

```
<serviceTask id="taskId" name="service"  
camunda:expression="${simpleGreeter.greet('you')}">  
</serviceTask>
```

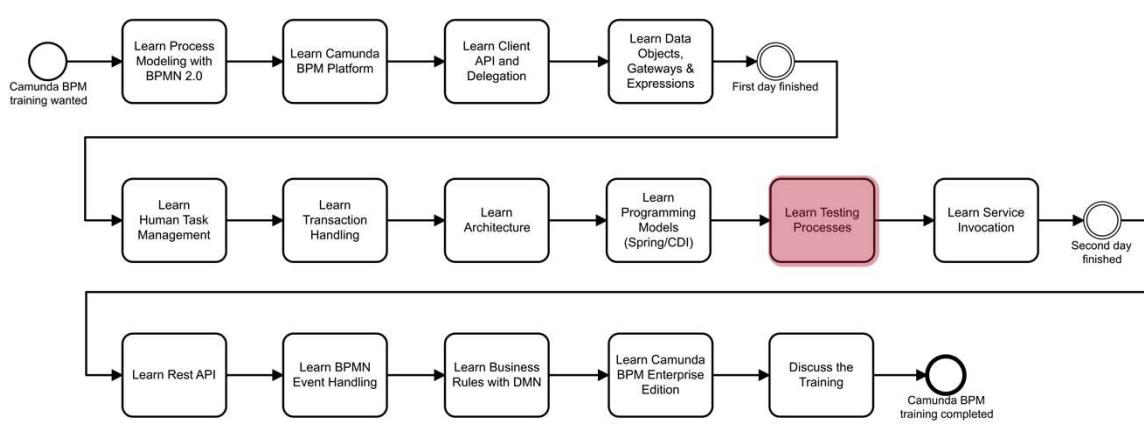
The resolved object is managed by the container (Spring/CDI) and can profit from other services (injection / lifecycle / ... ).



## Delegate Expression



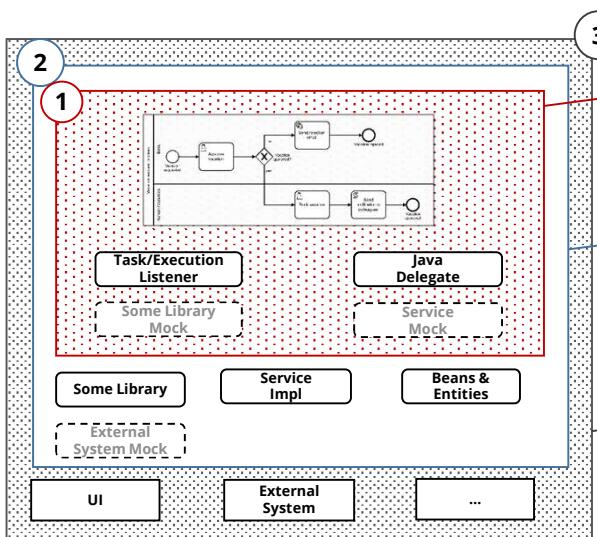
```
@Component  
public class BookingService implements JavaDelegate {  
    private final static Logger LOG = Logger.getLogger(BookingService.class.getName());  
  
    public void execute(DelegateExecution execution) throws Exception {  
        try {  
            // perform booking  
        } catch (Exception e) {  
            LOG.log(Level.WARNING, "Exception while performing booking", e);  
            throw new BpmnError("booking error");  
        }  
    }  
}
```



187

188

## Testing Scopes



Goal: Process Model with Data, EL & Adapter Logic

- In Memory
- Single Thread
- No Container
- No JobExecutor

Goal: Close to Real-Life environment

- Container
- Arquillian & co

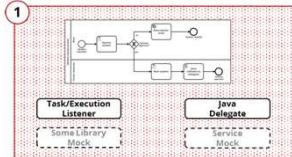
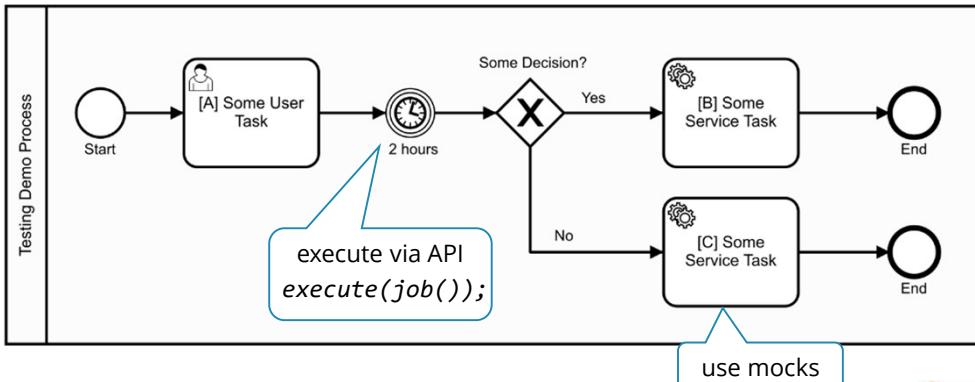
Goal: It REALLY works

- Integration Tests
- Human Driven

188



## Example



## Separate Process- and Business Logic

```
@Component("rejectionNotificationDelegate")
public class SendRejectionNotificationDelegate implements JavaDelegate {
    private EmailService emailService;
    @Inject
    public SendRejectionNotificationDelegate(EmailService emailService) {
        this.emailService = emailService;
    }
    @Override
    public void execute(DelegateExecution execution) throws Exception {
        // Input mapping
        String reason = (String) execution.getVariable("rejectionReason");
        String employee = (String) execution.getVariable("employee");
        String message = "Some message to " + employee + " about " + reason;
        // Service call
        String emailId = emailService.sendEmail(employee, "manager", reason, message);
        // Output mapping
        execution.setVariable("message", message);
        execution.setVariable("emailId", emailId);
    }
}
```

Separate business- from process logic

```
@Service
public class EmailService {
    public String sendEmail(...) {
        // Connect to mail server
        // send the email
        // maybe receive an mail id
        return messageId;
    }
}
```

This is what we want to mock



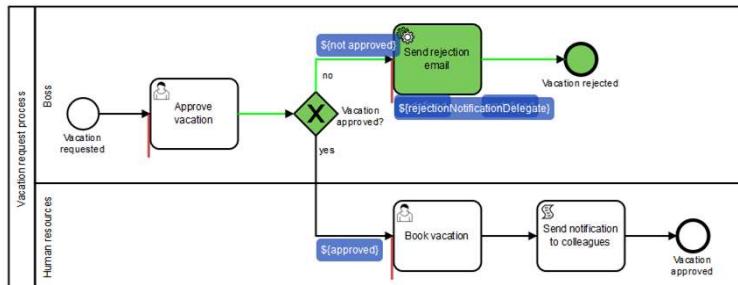
## Mock the Business Logic

```
@Mock  
private EmailService emailService;  
  
@Before  
public void setup() {  
    init(rule.getProcessEngine());  
  
    MockitoAnnotations.initMocks(this);  
    Mocks.register("rejectionNotificationDelegate", new SendRejectionNotificationDelegate(emailService));  
}  
  
@Test  
@Deployment(resources = "vacation-request.bpmn")  
public void testVacationNotApproved() {  
    Mockito.when(emailService.sendEmail(anyString(), anyString(), anyString(), anyString()))  
        .thenReturn("mockEmailId");  
    ProcessInstance processInstance = runtimeService()  
        .createProcessInstanceByKey("VacationRequestProcess")  
        .setVariables(withVariables("employee", "John", "from", new Date(), "to", new Date(),  
            "approved", false, "rejectionReason", "some reason"))  
        .startAfterActivity("approveVacationUserTask")  
        .execute();  
    assertThat(processInstance).isEnded().hasPassed("vacationRejectedEndEvent")  
        .variables().containsEntry("emailId", "mockEmailId");  
    Mockito.verify(emailService).sendEmail(eq("John"), anyString(), eq("some reason"), anyString());  
}
```

Register Delegate with mocked service

Assert variable passing between service and process

## Real Unit Test



```
@Test  
@Deployment(resources = "vacation-request.bpmn")  
public void testVacationNotApproved() {  
    ProcessInstance processInstance = runtimeService()  
        .createProcessInstanceByKey("VacationRequestProcess")  
        .setVariables(withVariables("employee", "John",  
            "from", new Date(),  
            "to", new Date(),  
            "approved", false))  
        .startAfterActivity("approveVacationUserTask")  
        .execute();  
  
    assertThat(processInstance).isEnded().hasPassed("vacationRejectedEndEvent");  
}
```

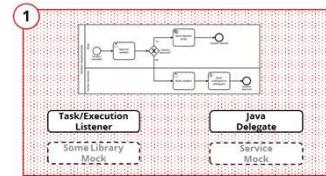
Process Instance Modification



## Testing on Scope 1

### We recommend

- JUnit
- Camunda Rule (included in camunda-engine)
- Camunda-bpm-assert (<https://github.com/camunda/camunda-bpm-assert>)
- Mock-Provider of your choice, e.g. Mockito or EasyMock or Jmockit (maybe + PowerMock)
- Process Instance Modification to start before the unit to test



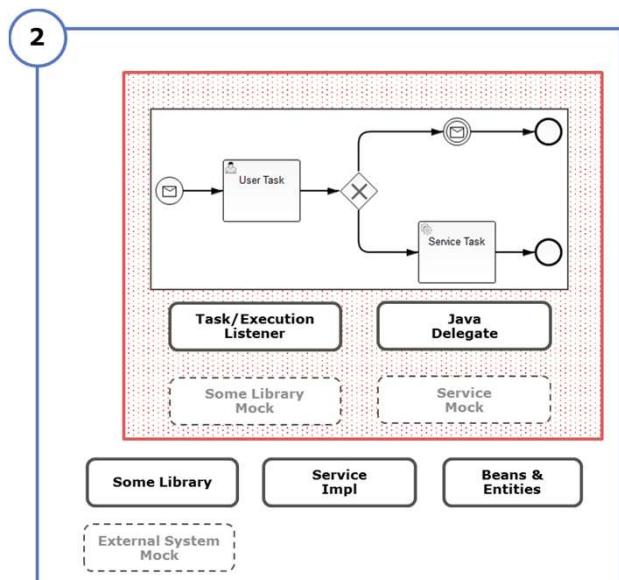
### Benefits

- In Memory Process Engine: Fast & always clean
- Single Thread / No Job Executor: Much easier to understand, no timing issues
- No Container: Less environment, faster turnaround times
- Runs quickly on every developer machine! By "right-click -> Run as Junit"!



## Testing on Scope 2

- Infrastructure needed (e.g. CDI, Transactions, JPA, ...)
- Possible Alternatives
  - Start own test environment
    - Minimalistic (Needle, ...)
    - Real (Weld, Hibernate, OpenEJB, ...)
    - Often Spring-Driven
  - Run tests as integration tests
  - Arquillian
- Forces:  
Effort for setup & maintenance, effort to run, possibility to run on developer machine

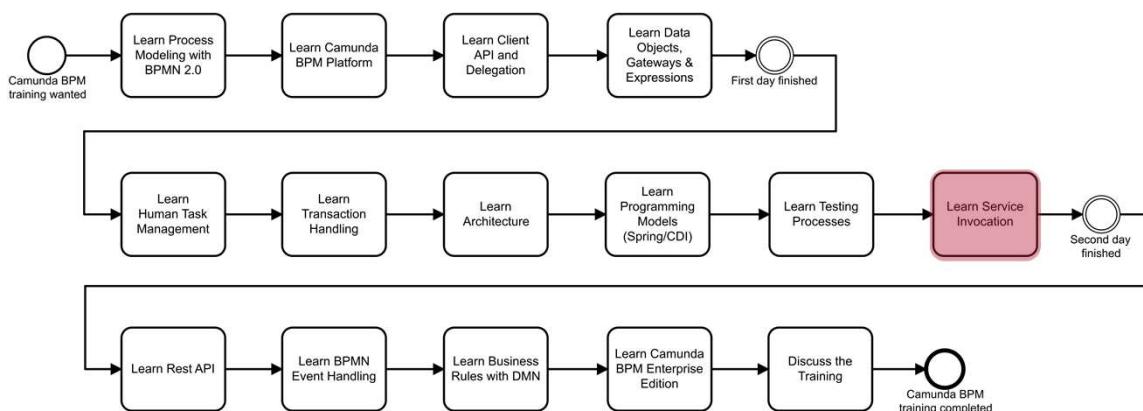


## Exercise 8



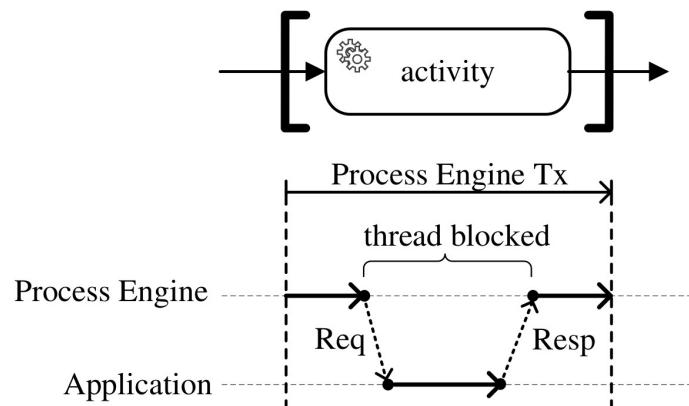
<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

195



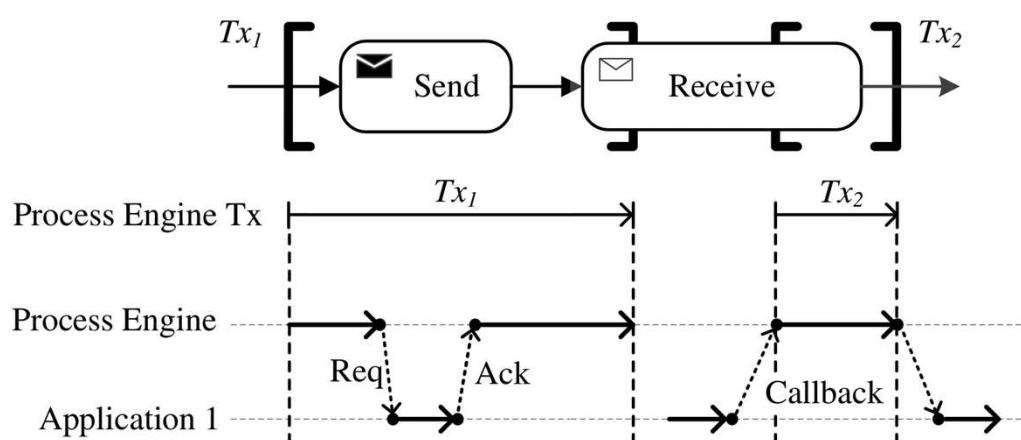
196

## Synchronous Service Invocation: Request / Response



197

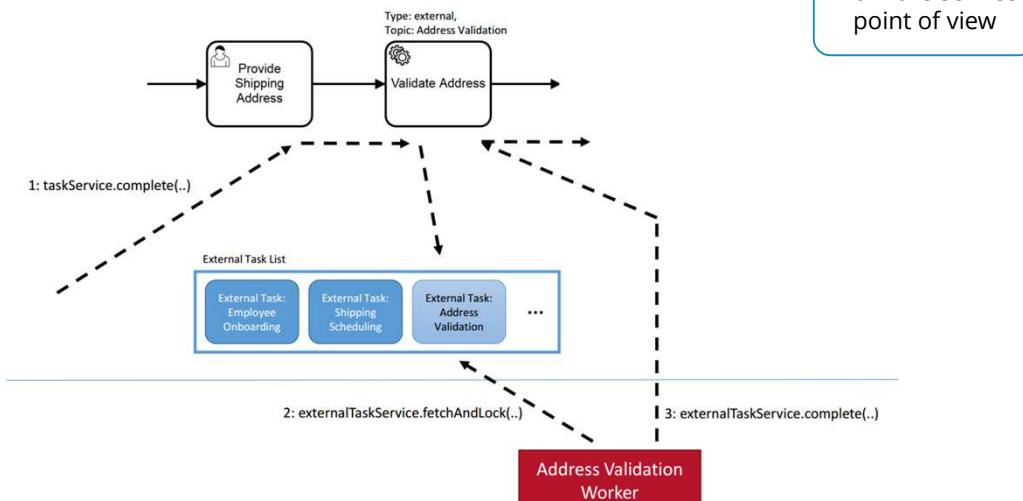
## Asynchronous Service Invocation: Req / Ack / Callback



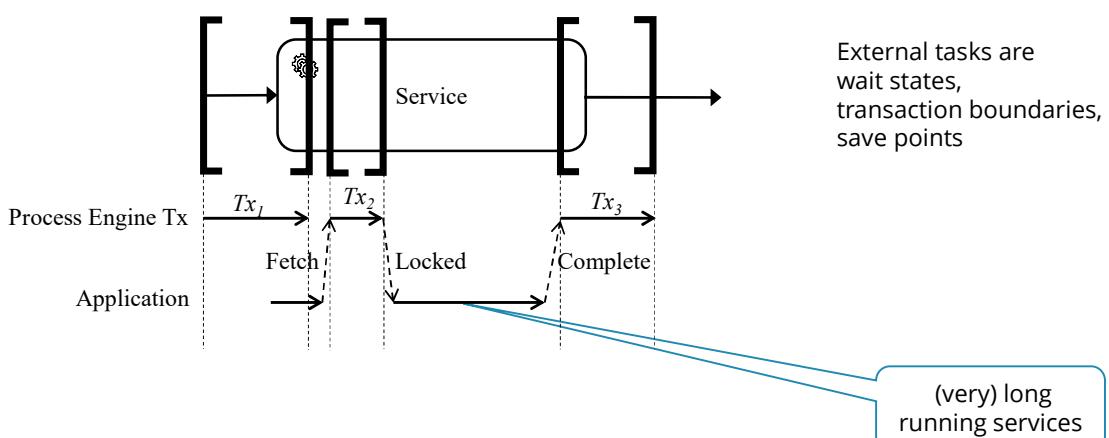
198



## External Service Task: Don't Call Us! We Call You!



## External Service Task Transactions

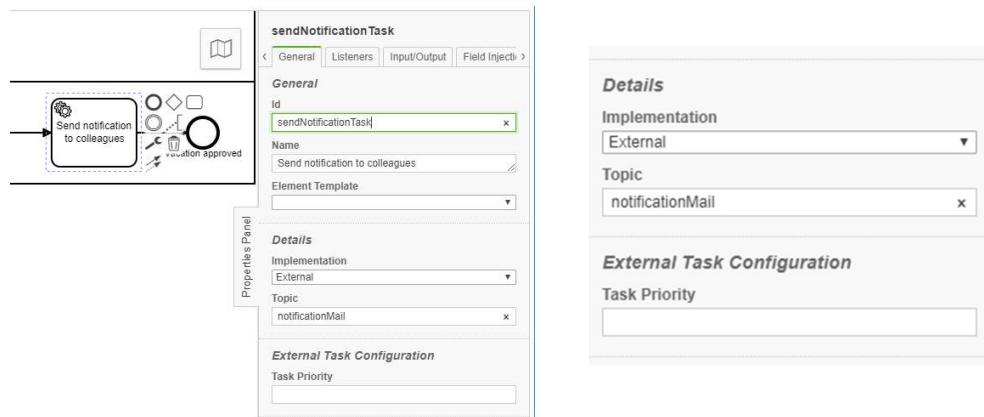


External Service Task:

<https://docs.camunda.org/manual/latest/user-guide/process-engine/external-tasks/>



## External Task Configuration



## External Task Client for Java Features

- Complete External Tasks
- Extend the lock duration of External Tasks
- Unlock External Tasks
- Report BPMN errors and failures
- Share variables with the Workflow Engine

<https://docs.camunda.org/manual/latest/user-guide/ext-client/>

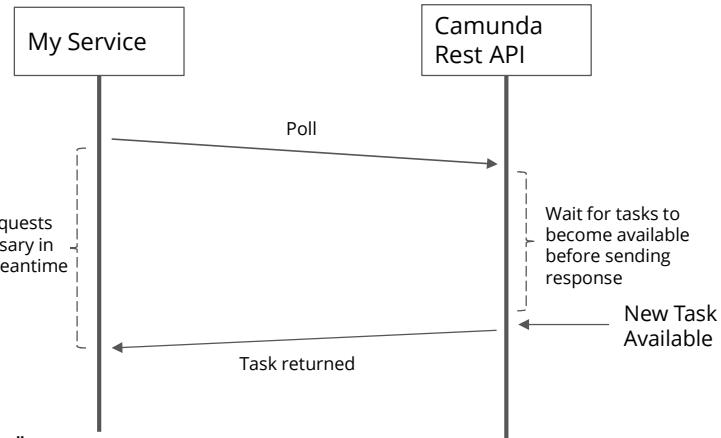


## Long Polling

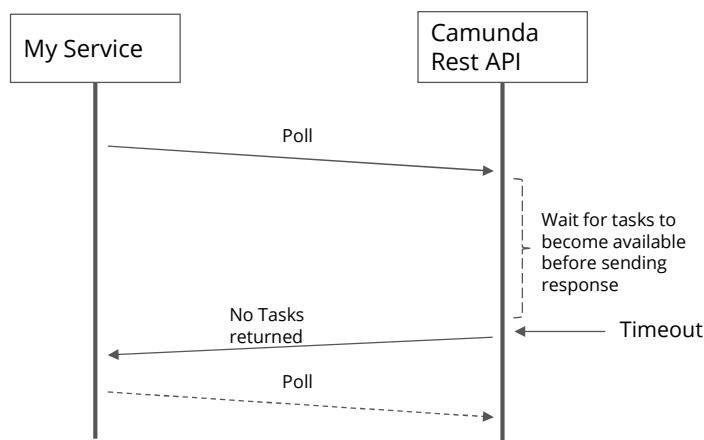
Reduce latency between create and work

POST /external-task/fetchAndLock  
Request Body:

```
{  
  "workerId": "myService",  
  "asyncResponseTimeout": "60000",  
  "maxTasks": 1,  
  "usePriority": true,  
  "topics": [{"topicName": "createOrder",  
    "lockDuration": 10000, "variables": ["orderId"]}]  
}
```

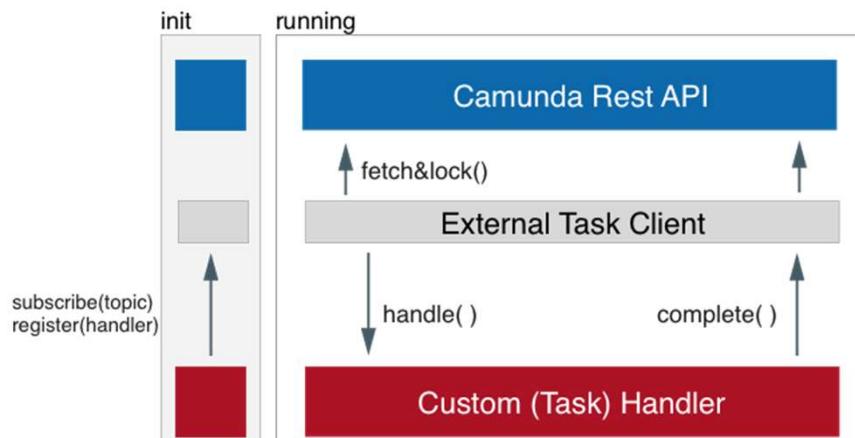


## What if no task is created?





## External Task Client



## External Task Client Setup

```
<properties>
    <camunda-ext-task-client.version>1.1.0</camunda-ext-task-client.version>

    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
    <version.java>1.8</version.java>
</properties>

<dependencies>
    <dependency>
        <groupId>org.camunda.bpm</groupId>
        <artifactId>camunda-external-task-client</artifactId>
        <version>${camunda-ext-task-client.version}</version>
    </dependency>

    <!-- use logback as logger -->
    <dependency>
        <groupId>ch.qos.logback</groupId>
        <artifactId>logback-classic</artifactId>
        <version>1.1.2</version>
    </dependency>
</dependencies>
```



## Implementation

```
public static void main(String[] args) {
    // bootstrap the client
    ExternalTaskClient client = ExternalTaskClient.create()
        .baseUrl("http://localhost:8080/rest")
        .lockDuration(60000)
        .maxTasks(2)
        .build();

    // subscribe to the topic
    TopicSubscriptionBuilder subscriptionBuilder = client.subscribe("notificationMail");
    subscriptionBuilder
        .handler((externalTask, externalTaskService) -> {
            String receiver = externalTask.getVariable("receiver");
            String subject = externalTask.getVariable("subject");
            String message = externalTask.getVariable("message");
            if (mailServerAvailable()) {
                LOGGER.info("send {} with subject {} to {}", message, subject, receiver);
                Map<String, Object> variables = new HashMap<>();
                variables.put("notificationTimestamp", new Date());
                externalTaskService.complete(externalTask, variables );
            } else {
                LOGGER.info("unlock the task as mailserver is unavailable");
                externalTaskService.unlock(externalTask);
            }
        }).open();
}
```

// Long polling  
.asyncResponseTimeout(90000)  
// reduce backoff to max 8 sec  
.backoffStrategy(  
 new ExponentialBackoffStrategy(500, 2, 8000))

Business Logic  
Called sequentially  
for each task



## External Task Client in JavaScript

The screenshot shows a browser window displaying the GitHub repository for the Camunda External Task Client in JavaScript. The page includes sections for 'Installing' (with npm and yarn installation commands) and 'Usage' (with steps and code examples). A code snippet at the bottom shows how to require the client module and its configuration.

```
const { Client, logger } = require("camunda-external-task-client-js");

// configuration for the Client:
// - 'baseUrl': url to the Workflow Engine
// - 'logger': utility to automatically log important events
```

<https://github.com/camunda/camunda-external-task-client-js>

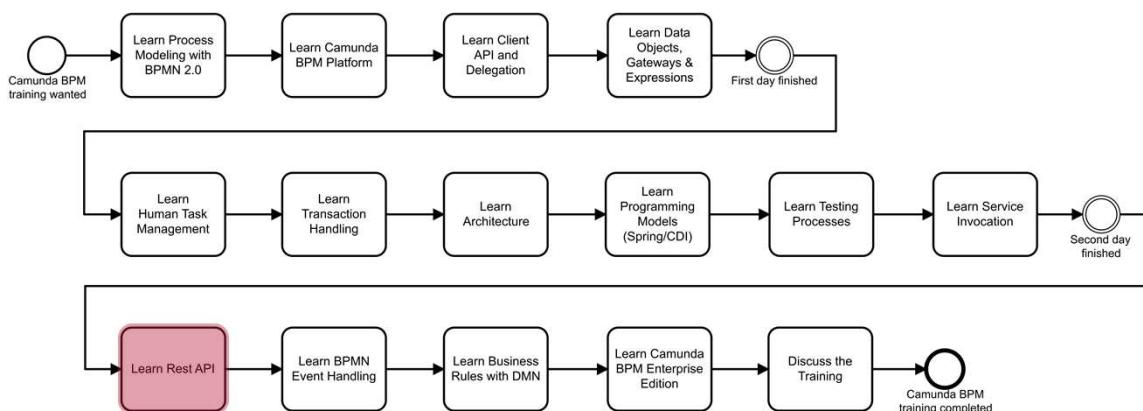


## Exercise 9



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

209

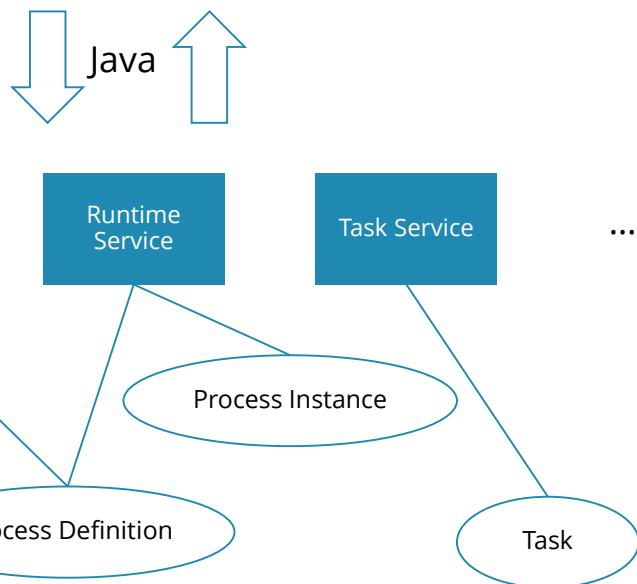


210



211

## Java API



211



212

## REST API



- Same power as Java API

212



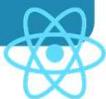
213

## Example Usage Scenarios

your custom  
tasklist



your mobile  
app



REST  
API

Tasklist



Cockpit

process engine

213



214

## Camunda BPM REST API

Some REST calls:

```
GET http://localhost:8080/rest/process-definition
```

```
POST http://localhost:8080/rest/process-definition/key/VacationRequestProcess/start
```

```
{"variables":  
  {"employee" : {"value" : "Mary", "type": "String"},  
   "from" : {"value" : "2016-12-20T00:00:00", "type": "Date"},  
   "until" : {"value" : "2016-12-28T23:59:00", "type": "Date"}  
}  
}
```

```
GET http://localhost:8080/rest/task?processDefinitionKey=VacationRequestProcess
```

Endpoint for prepackaged distribution: <http://localhost:8080/engine-rest/>

<https://docs.camunda.org/manual/latest/reference/rest/>

214



## Documentation

The screenshot shows a browser window displaying the Camunda Documentation website. The URL is https://docs.camunda.org/manual/latest/reference/rest/history/process-instance/get-process-instance-query-count/. The page title is "Get Process Instances Count". On the left, there is a navigation sidebar with sections like Introduction, User Guide, Reference (which is expanded to show Rest API, Overview, Authorization, Batch, Case Definition, Case Execution, Case Instance, Decision Definition, Decision Requirements Definition, Deployment, Engine, Execution, External Task, Filter, Group, History, Activity Instance, and Batch), and OPTIONS. The main content area has sections for Method (GET /history/process-instance/count), Parameters, and Query Parameters. A callout bubble on the right says "Most important for beginners". A sidebar on the right lists "ON THIS PAGE: Method, Parameters, Result, Response Codes, Example". At the bottom, there is a footer with links to camunda.org and docs.camunda.org, and a note about support for the Camunda Enterprise Edition.



## REST API: Embed and Extend!

- REST API as a library
- Own JAX-RS implementation
- Extension/limitation of resources possible

```
@ApplicationPath("/")
public class MyApplication extends Application {
    @Override
    public Set<Class<?>> getClasses() {
        Set<Class<?>> classes = new HashSet<Class<?>>();
        // add your own classes
        // add camunda engine rest classes that you need
        classes.add(ProcessEngineRestServiceImpl.class);
        classes.add(ProcessDefinitionRestServiceImpl.class);
    }
}
```

My process application

My JAX-RS web application

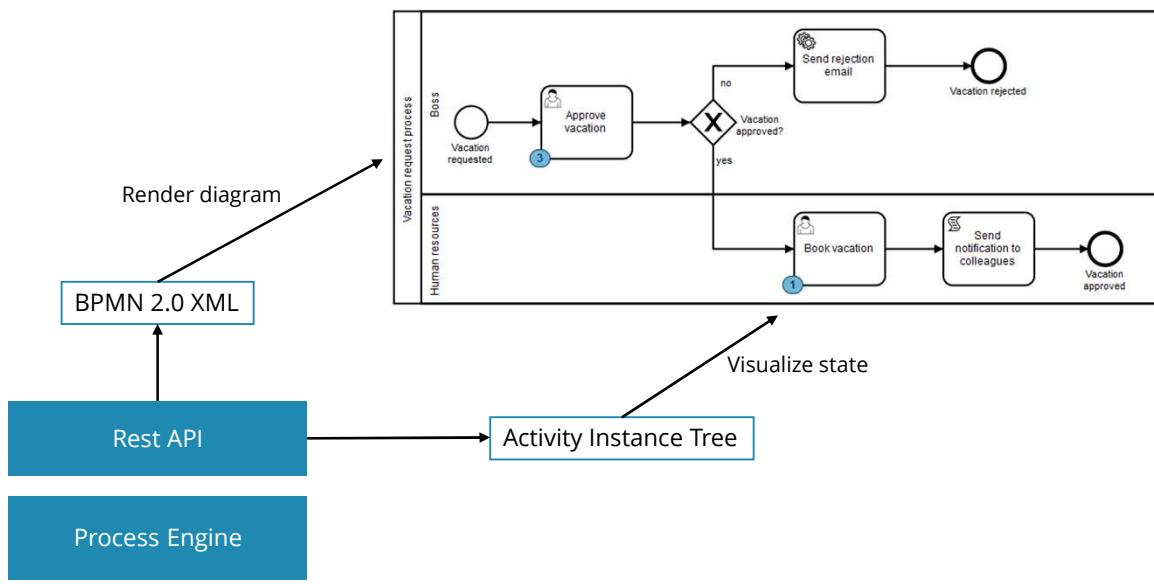
My favorite JAX-RS implementation

REST API (jar)



217

## REST API for Process Visualization



217



218

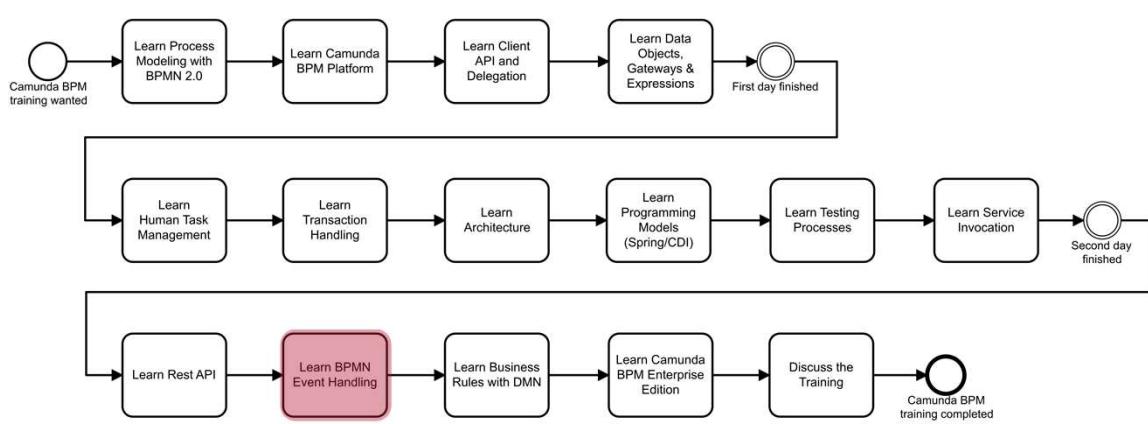
## Exercise 10

Optional for advanced rest users, a better exercise will follow tomorrow



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

218



219

220

## Event Coverage

Type	Start			Intermediate			End
	Normal	Event Subprocess	Event Subprocess non-interrupt	catch	boundary	boundary non-interrupt	
None	○						○ ○
Message	✉	✉	✉	✉	✉	✉	✉
Timer	⌚	⌚	⌚	⌚	⌚	⌚	⌚
Conditional	☰	☰	☰	☰	☰	☰	☰
Link			⇒			⇒	
Signal	△	△	△	△	△	△	△
Error	✗			✗			✗
Escalation	Ⓐ	Ⓐ		Ⓐ	Ⓐ	Ⓐ	Ⓐ
Termination							●
Compensation	⤳			⤳		⤳	⤳
Cancel				☒		☒	☒
Multiple	○○	○○	○○	○○	○○	○○	○○
Multiple Parallel	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕	⊕⊕

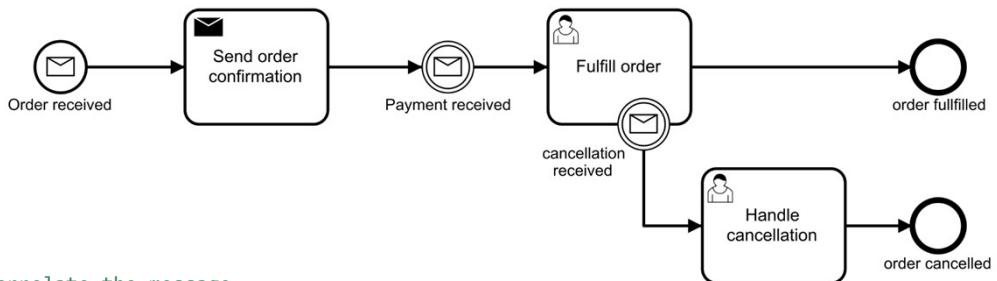
See:  
[docs.camunda.org/manual/latest/reference/bpmn20/#events](https://docs.camunda.org/manual/latest/reference/bpmn20/#events)

220



221

## Message Events



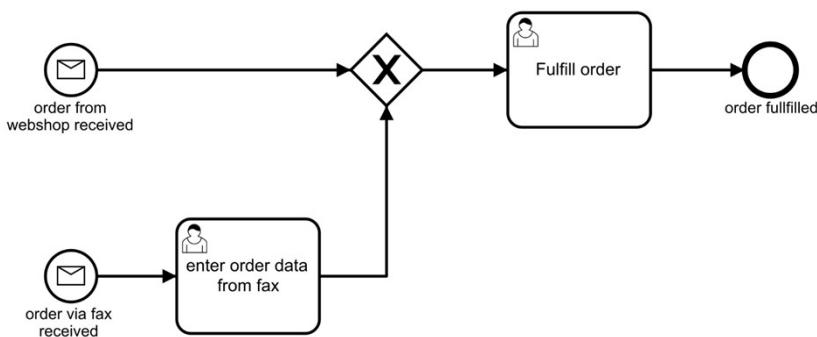
```
// correlate the message
runtimeService
    .createMessageCorrelation("Msg_PaymentReceived") // message name
    .processInstanceBusinessKey("AB-123")
    .setVariable("payment_type", "creditCard")
    .correlateWithResult();
```

221



222

## Multiple Message Start Events Possible



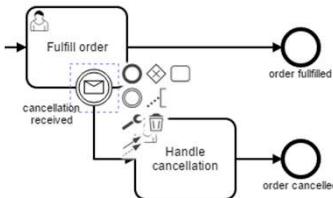
```
runtimeService
    .createMessageCorrelation("orderFromWebShop")
    .processInstanceBusinessKey("Order 123")
    .setVariable("myPayload", "myPayloadValue")
    .correlateWithResult();
```

222



223

## Configuring Message Names



BoundaryEvent\_14qb0su

General	Listeners	Extensions
---------	-----------	------------

**General**

**Id:** BoundaryEvent\_14qb0su

**Name:** cancellation received

**Details**

**Message:** orderCancelMessage (id=Message\_04dfh6o)

**Message Name:** orderCancelMessage

**Asynchronous Continuations**

Asynchronous Before

Asynchronous After

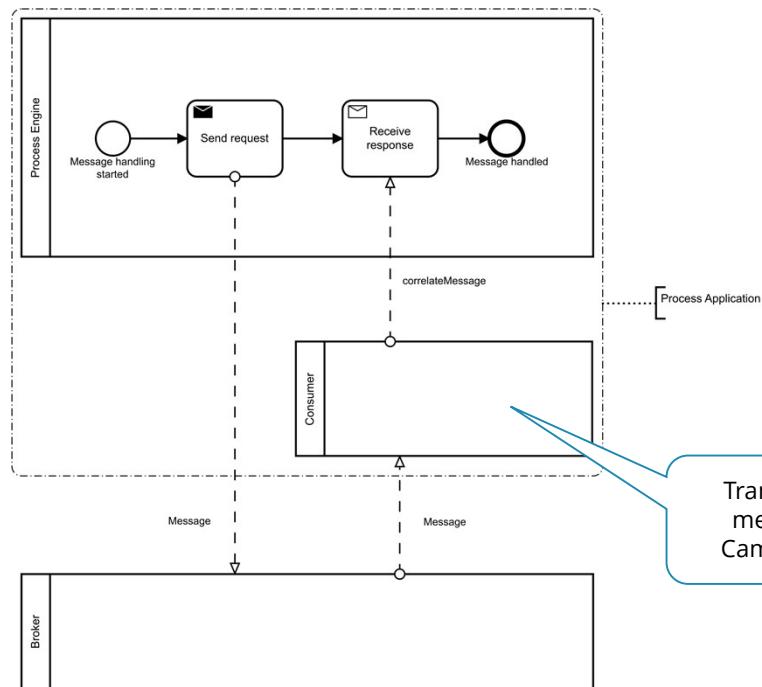
**Documentation**

Element Documentation

```
<bpmn:process id="OrderHandlingProcess" isExecutable="true">
  ...
  <bpmn:boundaryEvent id="BoundaryEvent_14qb0su"
    name="cancellation received" attachedToRef="fulFillOrderUserTask">
    <bpmn:messageEventDefinition messageRef="Message_04dfh6o" />
  </bpmn:boundaryEvent>
  <bpmn:userTask id="fulFillOrderUserTask" name="Fulfill order">
  </bpmn:userTask>
  ...
</bpmn:process>
<bpmn:message id="Message_04dfh6o" name="orderCancelMessage" />
```

223

## Messaging



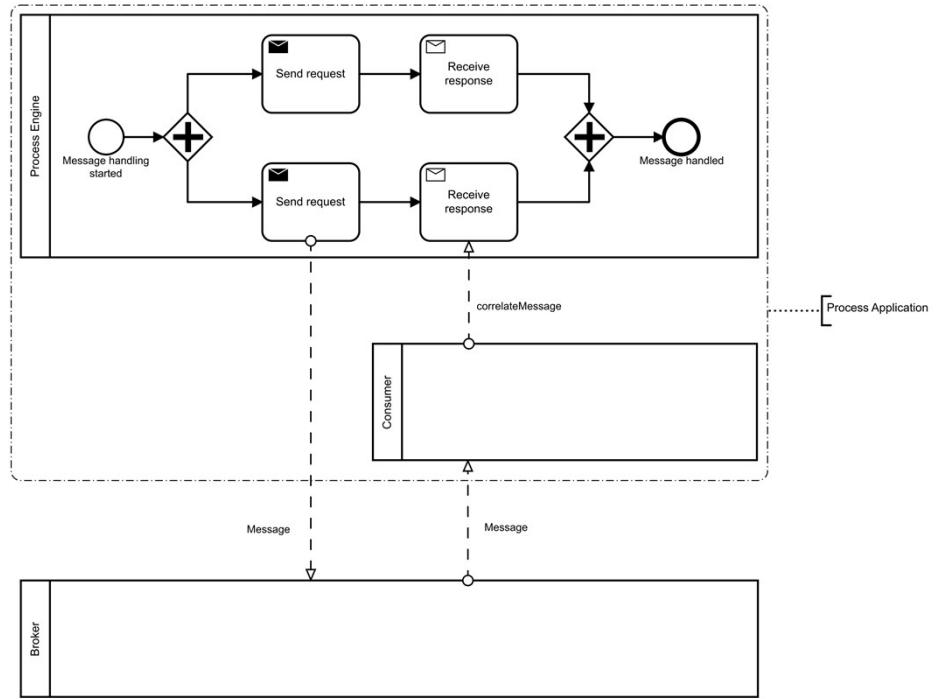
224

224



## Messaging (#TX?)

225

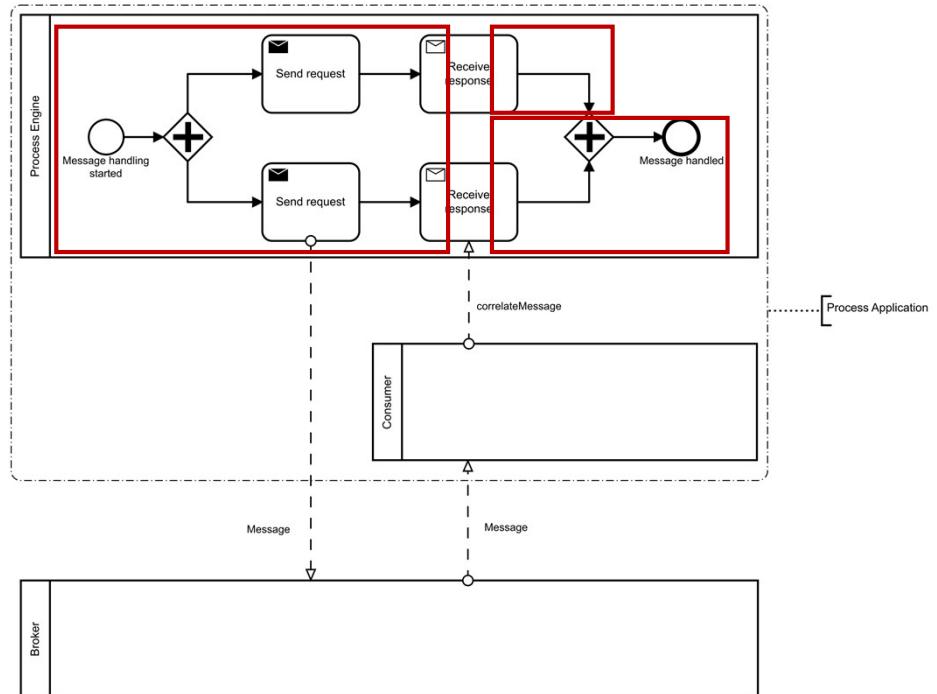


225



## Messaging

226



226



227

## Exercise 11



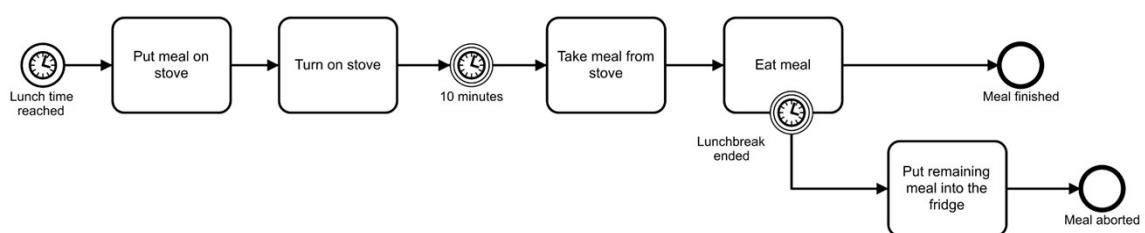
<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

227



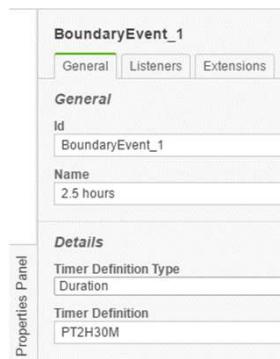
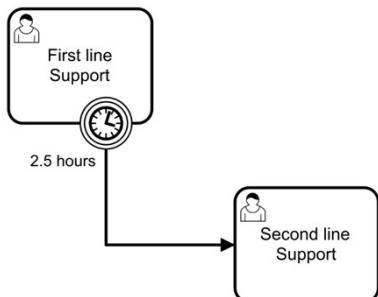
228

## Timer Events



228

## Timer Event Configuration



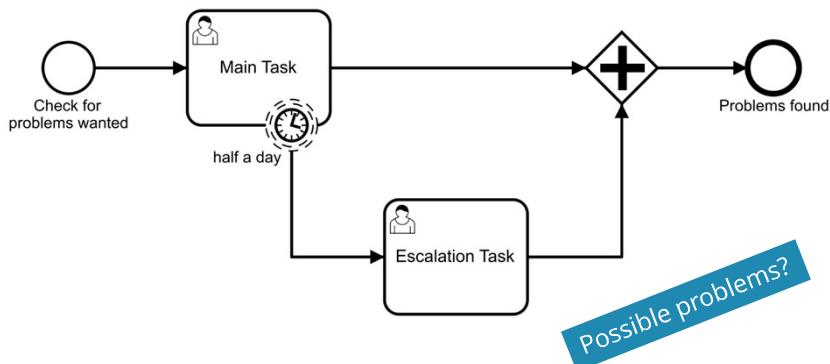
```

<bpmn:process id="Process_1" name="Timer Event Configuration Example" isExecutable="true">
  <bpmn:userTask id="firstLineSupportUserTask" name="First Line Support" />
  <bpmn:boundaryEvent id="BoundaryEvent_1" name="2.5 hours" attachedToRef="firstLineSupportUserTask">
    <bpmn:timerEventDefinition>
      <bpmn:timeDuration xsi:type="bpmn:tFormalExpression">PT2H30M</bpmn:timeDuration>
    </bpmn:timerEventDefinition>
  </bpmn:boundaryEvent>
  <!-- -->
</bpmn:process>

```

ISO8601: [https://en.wikipedia.org/wiki/ISO\\_8601](https://en.wikipedia.org/wiki/ISO_8601)

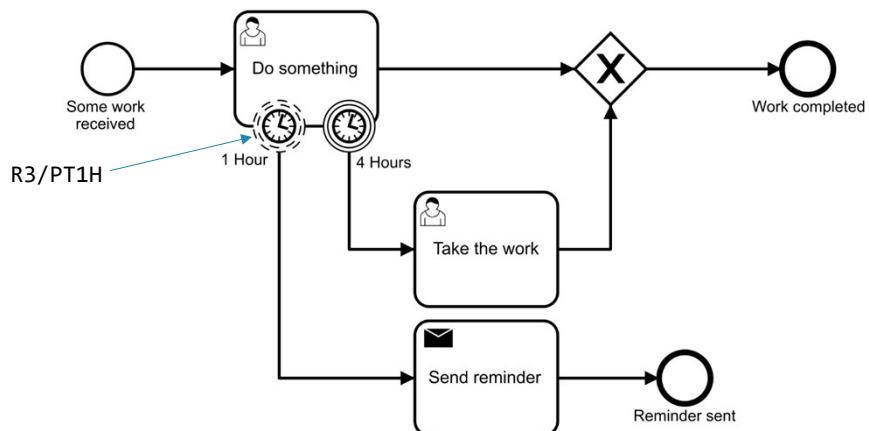
## Non Interrupting Timer Event





231

## Pattern for Two Stage Escalation



231



232

## Time-triggered Events

To reduce the clutter  
in the process model

**Task Listener**

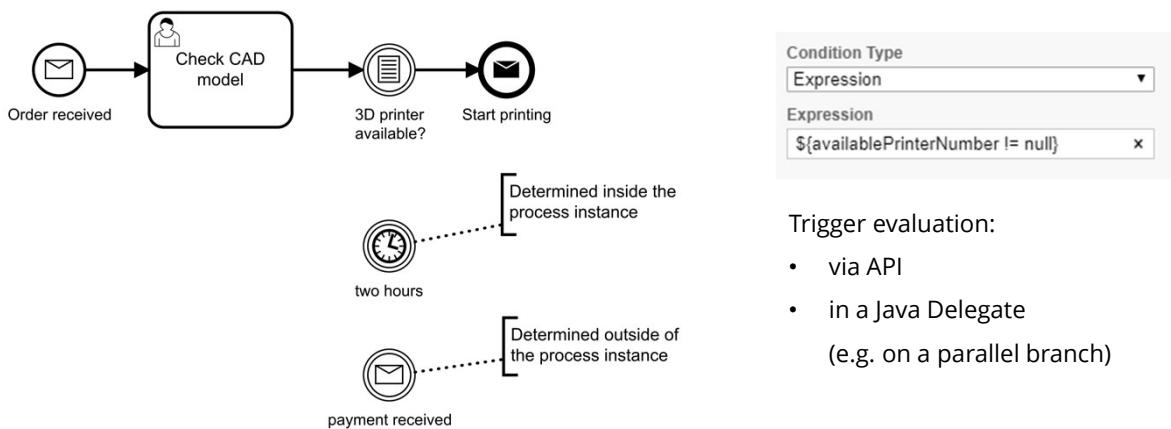
Event Type	timeout
Listener Id	automaticReassign
Listener Type	Java Class
Java Class	com.camunda.training.ReassignTaskl
Timer Definition Type	Duration
Timer Definition	PT2H30M

232



233

## Conditional Events

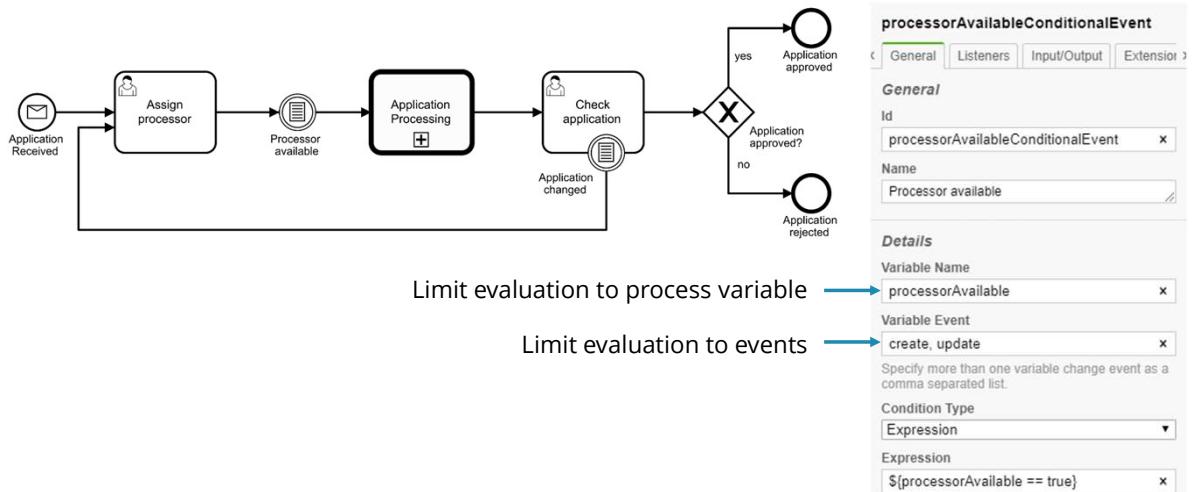


233



234

## Conditional Events



234



## Transient Variables

- Not stored in the database
- No update and delete
- Available with TypedValue

```
//primitive values
TypedValue typedTransientStringValue = Variables.stringValue("foobar", true);

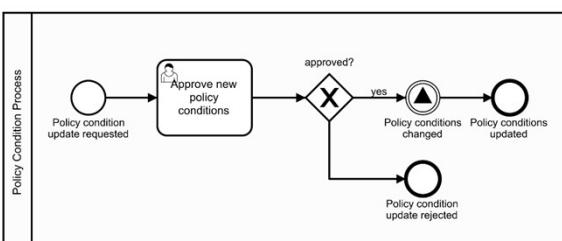
//object value
com.camunda.consulting.Customer customer = new com.camunda.consulting.Customer();
TypedValue typedTransientObjectValue = Variables.objectValue(customer, true).create();

//file value
TypedValue typedTransientFileValue = Variables.fileValue("file.txt", true)
    .file(new File("path/to/the/file.txt"))
    .mimeType("text/plain")
    .encoding("UTF-8")
    .create();
```

**boolean isTransient**



## Signal Events - Broadcast to All Process Instances

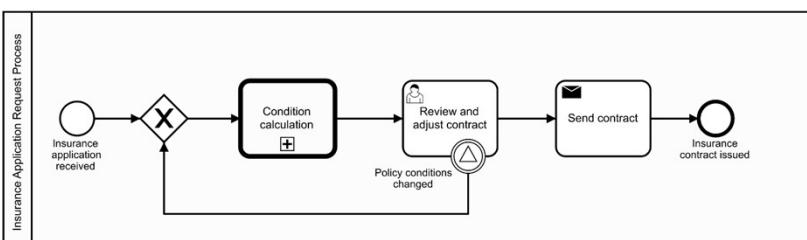


policyConditionChangedSignalEvent

General	Listeners	Input/Output	Extensions
<b>Id</b> policyConditionChangedSignalEvent			
<b>Name</b> Policy conditions changed			

**Details**

<b>Signal</b> policy_condition_changed_signal (id=Signal_118jje)
<b>Signal Name</b> policy_condition_changed_signal



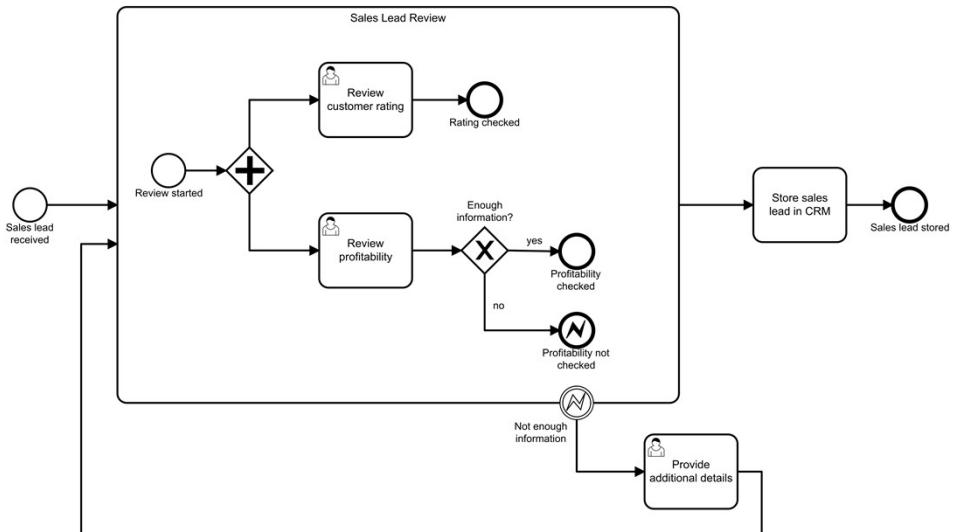
Properties Panel

policyConditionChangedSignalEvent
-----------------------------------



237

## Error Event

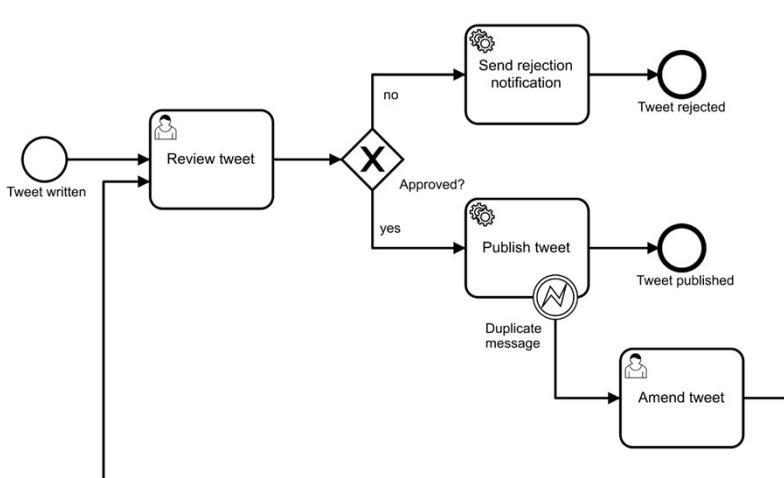


237



238

## Error Handling



BoundaryEvent_0y033u6	
	General
Id	BoundaryEvent_0y033u6
Name	Duplicate message
Details	
Error	Duplicate message error (id=Error_110prwb)
Error Name	Duplicate message error
Error Code	duplicate_message
Error Code Variable	publishFailedCode
Error Message Variable	publishFailedMessage

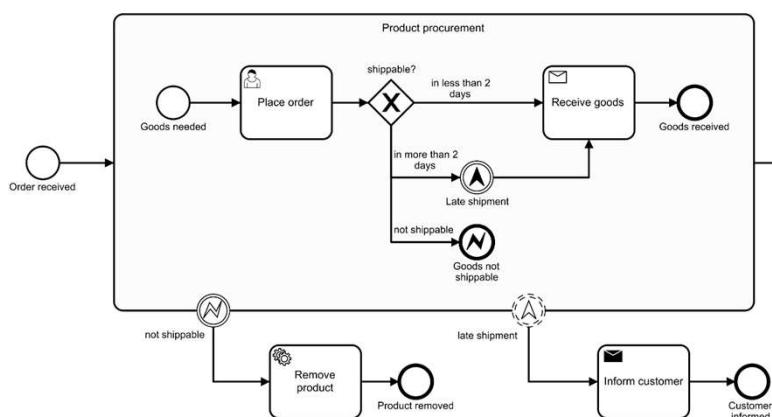
238

# Demo



239

## Escalation and Error Events



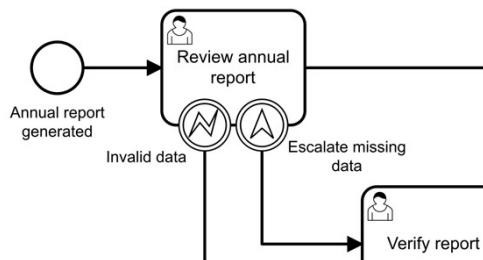
Properties Panel	
<b>IntermediateThrowEvent_0t7joxw</b>	
<a href="#">General</a>	<a href="#">Listeners</a> <a href="#">Input/Output</a> <a href="#">Ext</a>
<b>General</b>	
<b>Id</b>	IntermediateThrowEvent_0t7joxw
<b>Name</b>	Late shipment
<b>Details</b>	
<b>Escalation</b>	Late Shipment (id=Escalation_1iofngp)
<b>Escalation Name</b>	Late Shipment
<b>Escalation Code</b>	lateShipmentEscalation

240



241

## Error and Escalation Event on User Tasks



```
<button cam-error-code="bpmn-error-543"
       cam-error-message="anErrorMessage"
       class="form-control">Abort</button>

<button cam-escalation-code="escalation-123"
       class="form-control">Escalate</button>
```

```
taskService.handleBpmnError(task.getId(),
    "errorCode", ex.getMessage(), variables);

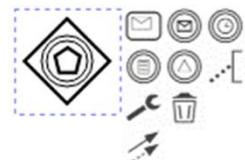
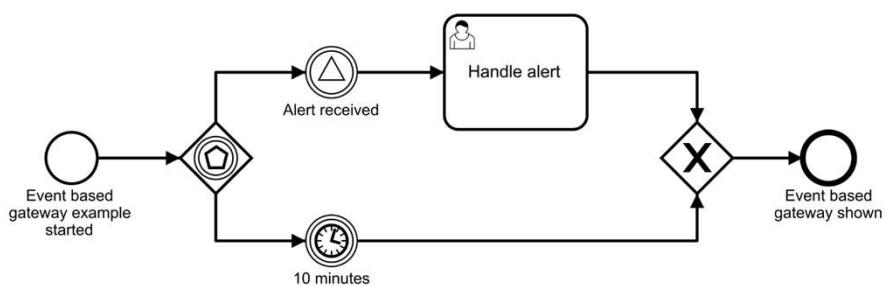
taskService.handleEscalation(task.getId(),
    "myEscalationcode", variables);
```

241



242

## Event-Based Gateway – Simple Example



242



## Event-Based Gateway – Real World Example

### Please model the following process:

If an insurant could be possibly subrogated against, I get information about that. I check that case and if the possibility is really there, I send a request for payment to the insurant and make me a reminder. If recourse is not possible, I close the case.

When we receive the money, I make a booking and close the case. If the insurant disagrees with the recourse, I'll have to check the reasoning of that. If he is right, I simply close the case. If he is wrong, I forward the case to a collection agency.

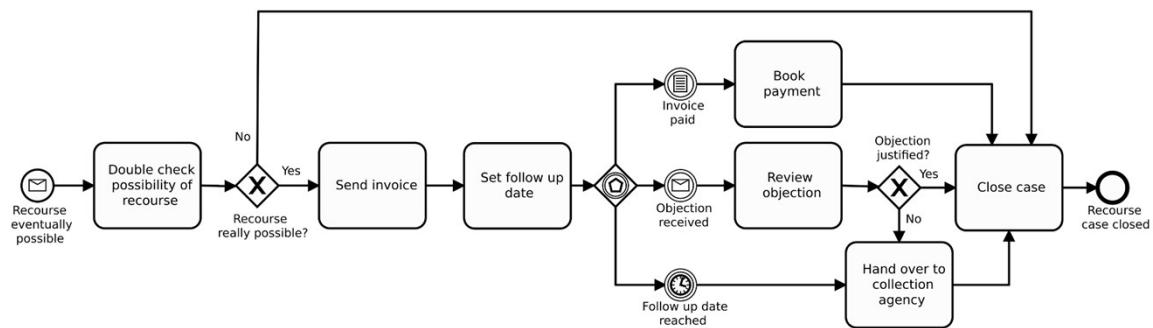
If the deadline for disagreement is reached and we haven't received any money, I forward the case to the collection agency as well.

### Background information:

Insurants can be forced to pay back money they received from the insurance company for different reasons. This is called recourse. Here the clerk describes how this process works.



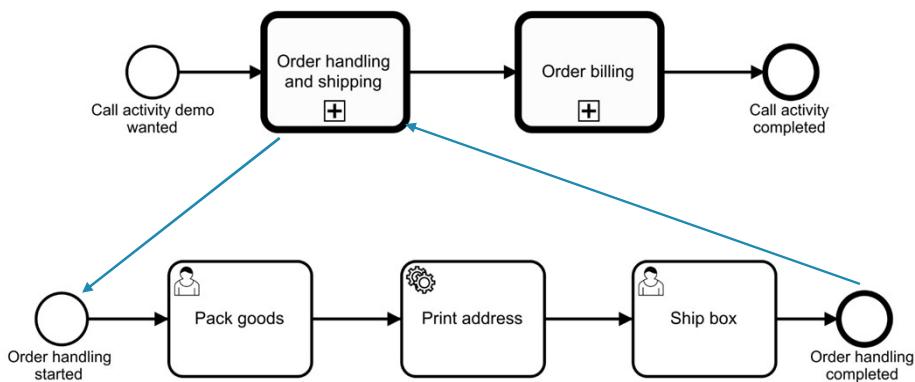
## Event-Based Gateway – Real World Example





245

## Reusable Sub-Processes: Call Activities

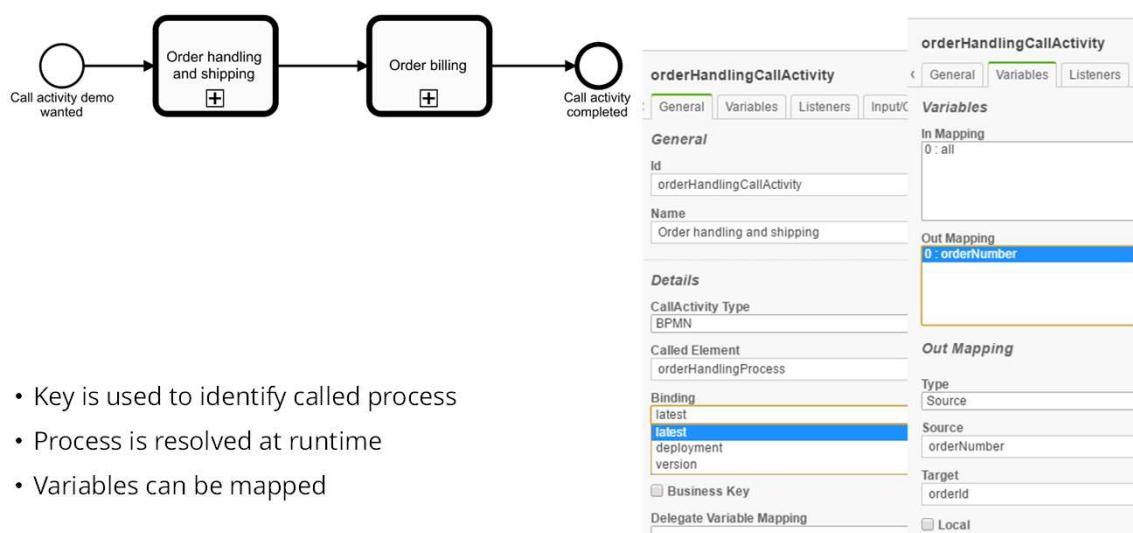


245



246

## Reusable Sub-Processes: Call Activities

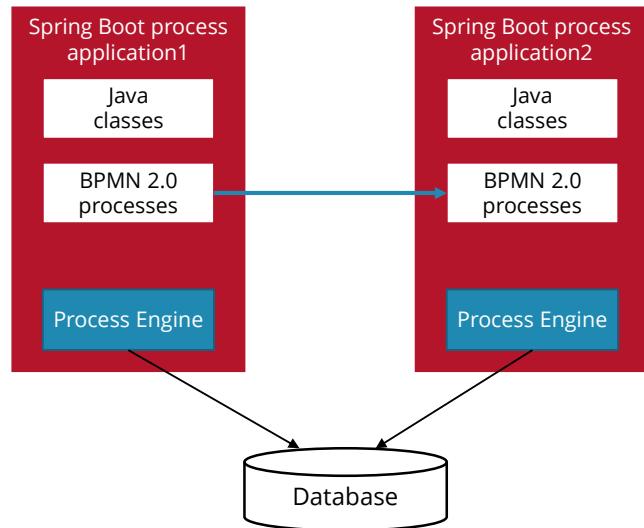


246



247

## Calling Processes From Other Process Applications

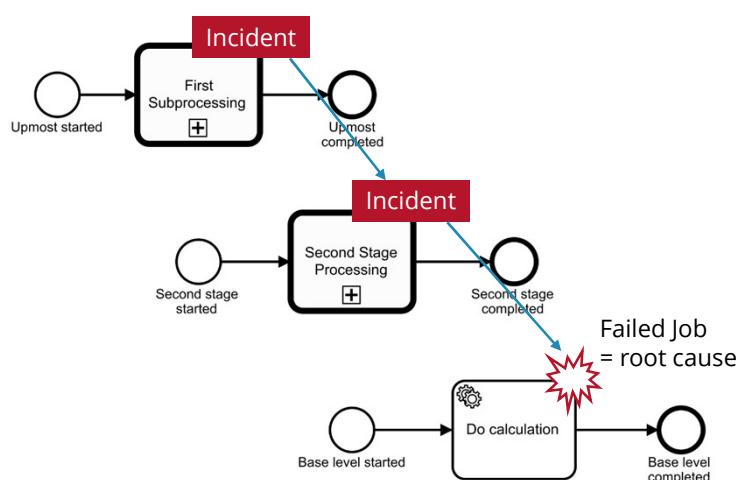


247



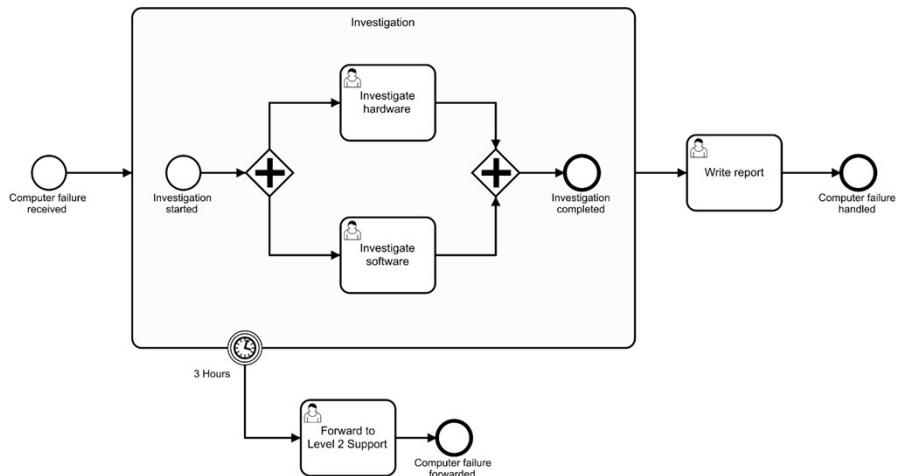
248

## Drill Down



248

## Embedded Sub-Processes



249

## The Sub-Process Is Part of the Main Process

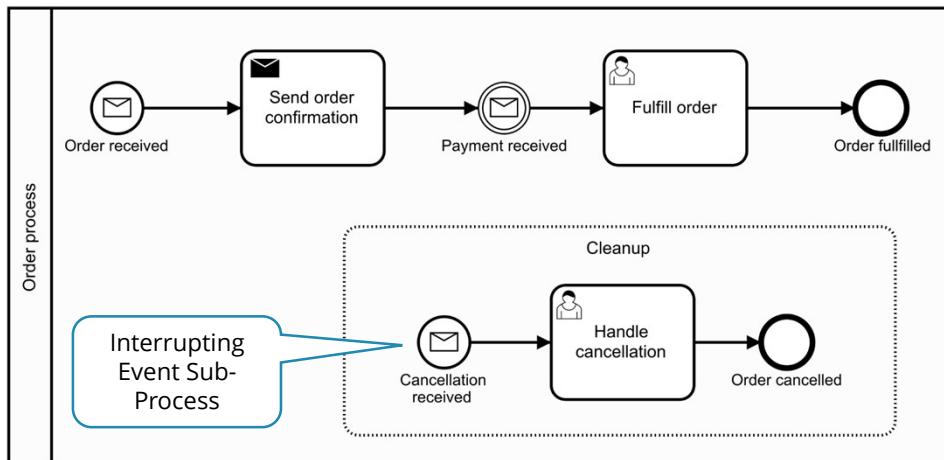
```

<bpmn:process id="Process_1" isExecutable="false">
  <bpmn:startEvent id="StartEvent_1" name="Computer failure received"></bpmn:startEvent>
  <bpmn:sequenceFlow id="SequenceFlow_1ilu0mr" sourceRef="StartEvent_1"
  targetRef="investigationSubprocess" />
  <bpmn:subProcess id="investigationSubprocess" name="Investigation">
    <bpmn:startEvent id="investigationStartedStartEvent" name="Investigation
    started"></bpmn:startEvent>
    <bpmn:parallelGateway id="ParallelGateway_1"></bpmn:parallelGateway>
    <bpmn:userTask id="investigateHardwareUserTask" name="Investigate hardware"></bpmn:userTask>
    <bpmn:userTask id="investigateSoftwareUserTask" name="Investigate software"></bpmn:userTask>
    <bpmn:parallelGateway id="ParallelGateway_2"></bpmn:parallelGateway>
    <bpmn:endEvent id="investigationCompletedEndEvent" name="Investigation
    completed"></bpmn:endEvent>
    <bpmn:sequenceFlow id="SequenceFlow_01s0ans" sourceRef="investigationStartedStartEvent"
    targetRef="ParallelGateway_1" />
    ...
  </bpmn:subProcess>
  ...

```

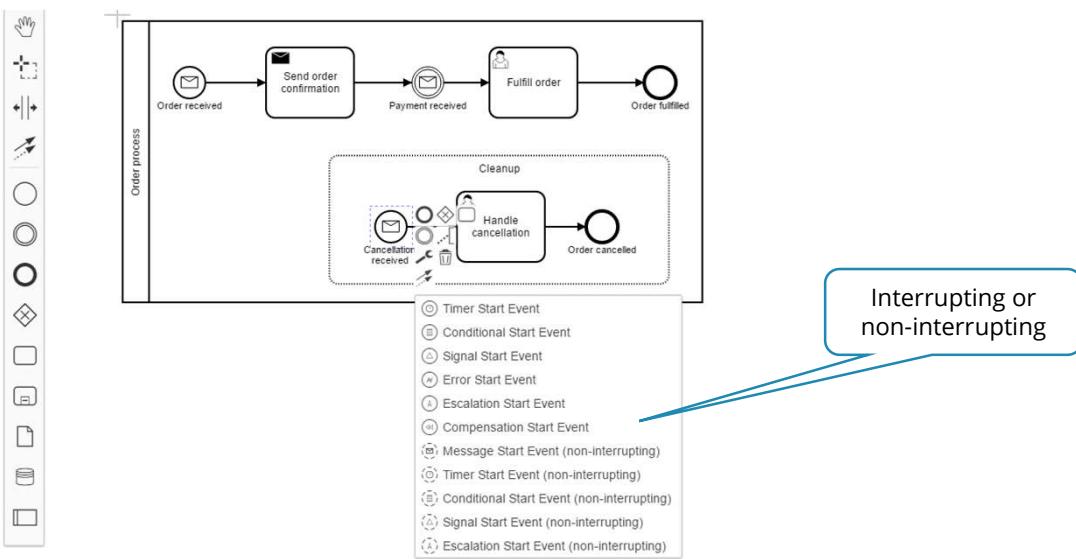
250

## Event Sub-Process



251

## Possible Events to Start an Event Sub-Process

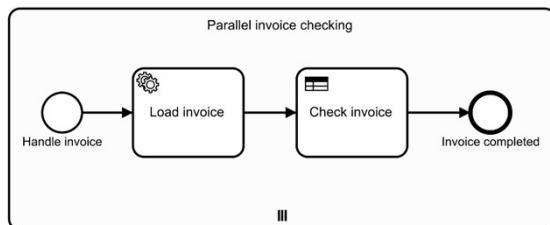


252

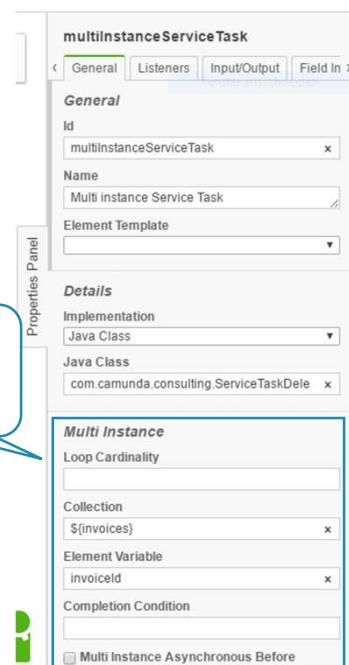


253

## Multiple Instance Marker



Multi  
Instance  
Control

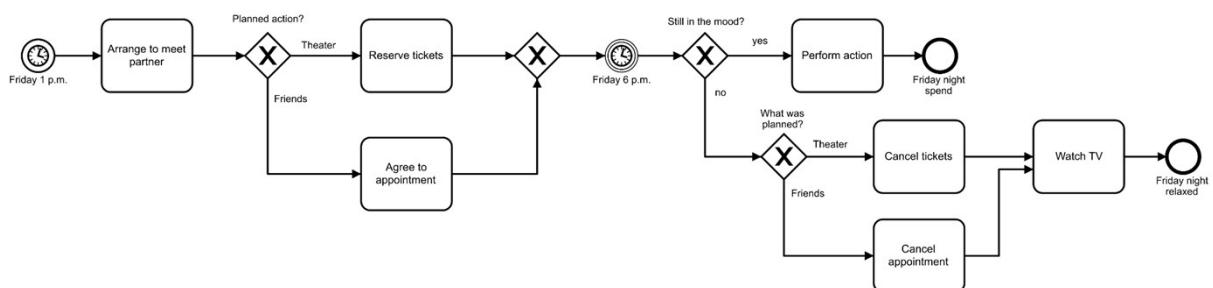


253



254

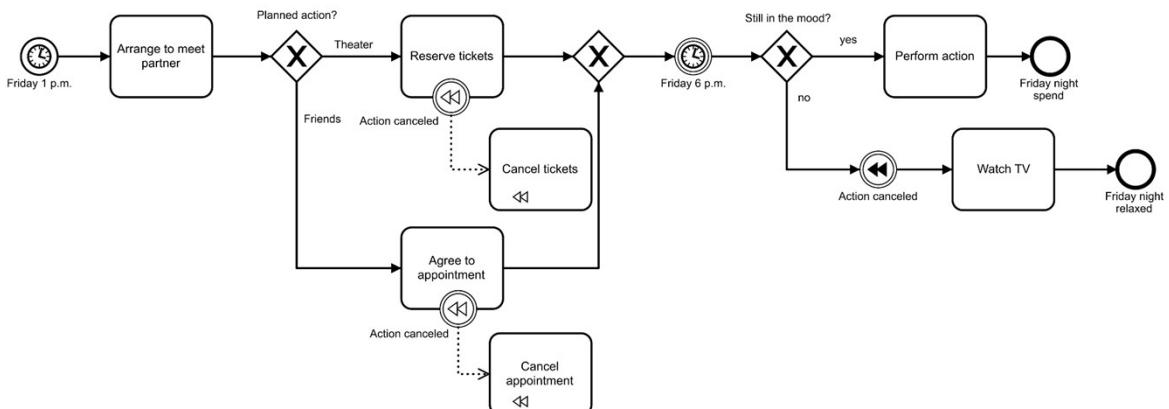
## Compensation



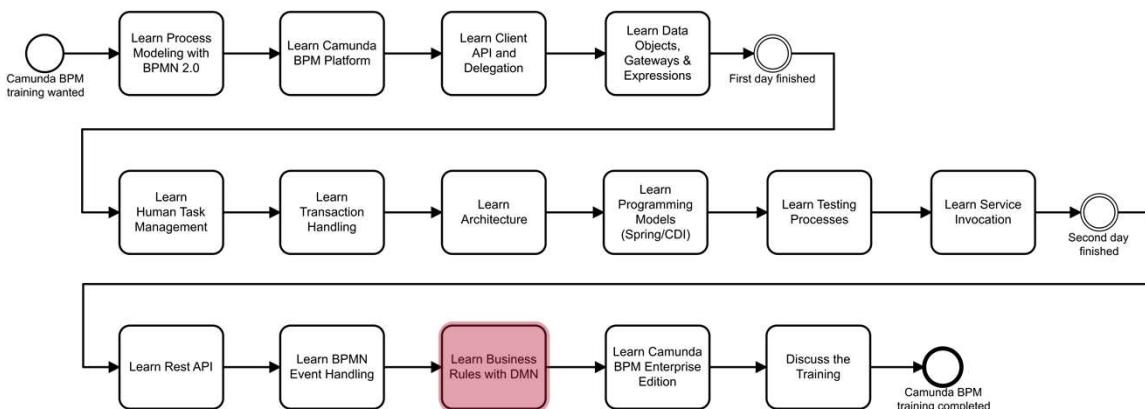
254



## Compensation Events



255

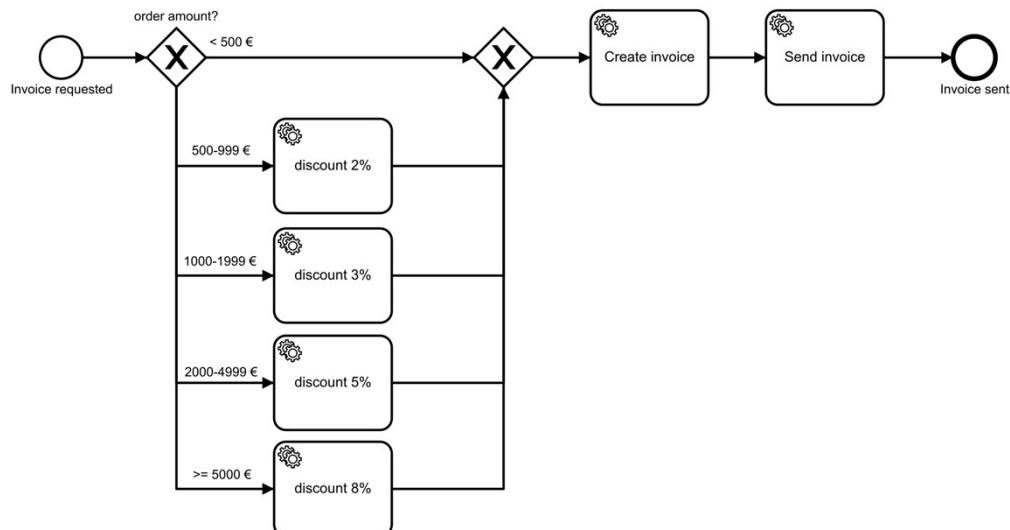


256



257

## Is This a Good Process Model?

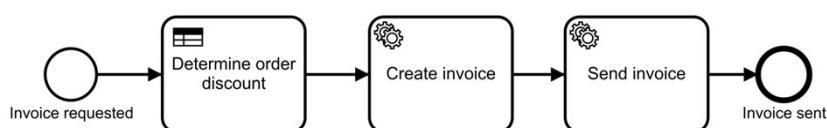


257



258

## An Improved Version



258



## Discount Rules as DMN Decision Table

The screenshot shows the Camunda Modeler application window. The title bar says "Camunda Modeler". The main area displays a "Decision Table" titled "Order Discount" for a rule named "orderDiscount". The table has two columns: "Input" (with header "U") and "Output" (with header "Discount"). The "Input" column contains values 1 through 5 corresponding to ranges of "Order Amount" (long type). The "Output" column contains values 0 through 8. An "Annotation" column is present but empty. At the bottom of the table, there are buttons for "Decision Table" and "XML". A "View DRD" button and a "Log" button are also visible.

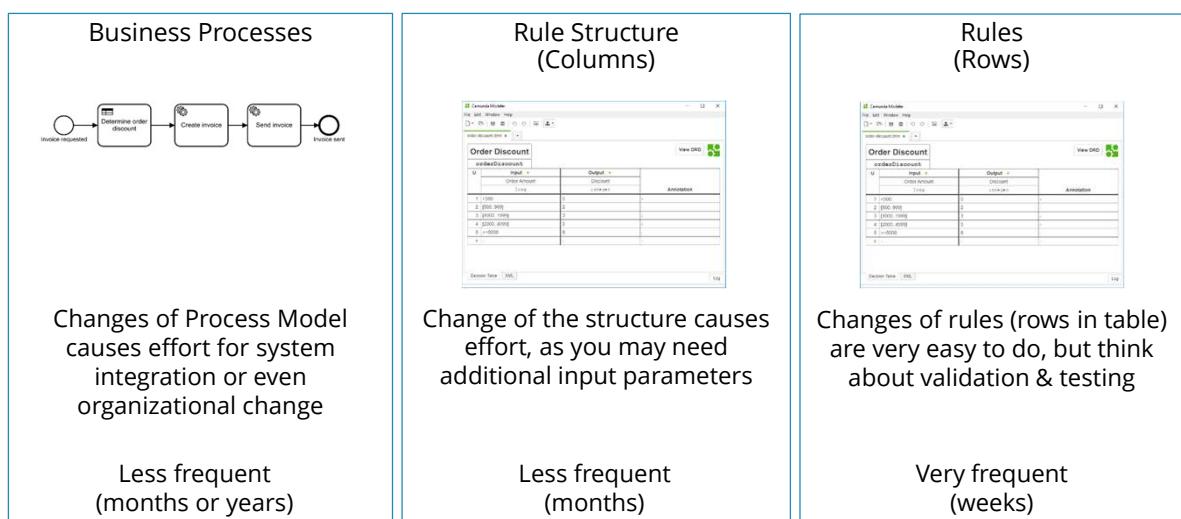
U	Input +	Output +	Annotation
1	<500	0	-
2	[500..999]	2	-
3	[1000..1999]	3	-
4	[2000..4999]	5	-
5	>=5000	8	-
	+	-	-



## The Standard DMN

- Decision Model and Notation
- Is currently in publications by OMG for Decisions/Business Rules
- Specification: „The primary goal of DMN is to provide a **common notation** that is readily understandable by all **business users**, from the business analysts needing to create initial decision requirements and then more detailed decision models, to the **technical developers** responsible for automating the decisions in processes, and finally, to the business people who will manage and monitor those decisions. DMN creates a standardized **bridge for the gap between the business decision design and decision implementation**. DMN notation is designed to be **useable alongside the standard BPMN** business process notation.”
- Camunda shipped DMN 1.1 support with 7.4

## Typical Frequency of Changes



261

## Rules Expressed as Decision Table

 Decision name		 Input expression	
 Hit Policy "Unique"		 Input entry	
			 Output entry = Result of rule
			 Optional remarks. Typically used for motivation of rule

**Dish**

decision

Input +      Output +

Season      Dish

string      string

Annotation

1 "Fall"      "Spareribs"

2 "Winter"      "Roastbeef"

3 "Spring"      "Steak"

4 "Summer"      "Light Salad and a nice Steak"

Hey, why not?

262



## Multiple Inputs

Input entries can have various data formats

Multiple inputs always follow „AND“ logic

Dish		Output		Annotation
U	Input	Output	+	
	Season string	Vegetarian Guest boolean	Dish string	
1	"Fall"	false	"Spareribs"	-
2	"Winter"	false	"Roastbeef"	-
3	"Spring"	false	"Steak"	-
4	"Summer"	false	"Light Salad and a nice Steak"	Hey, why not?!
5	-	true	"Pasta"	-
+ -			-	-



## Technical Details

Decision Id

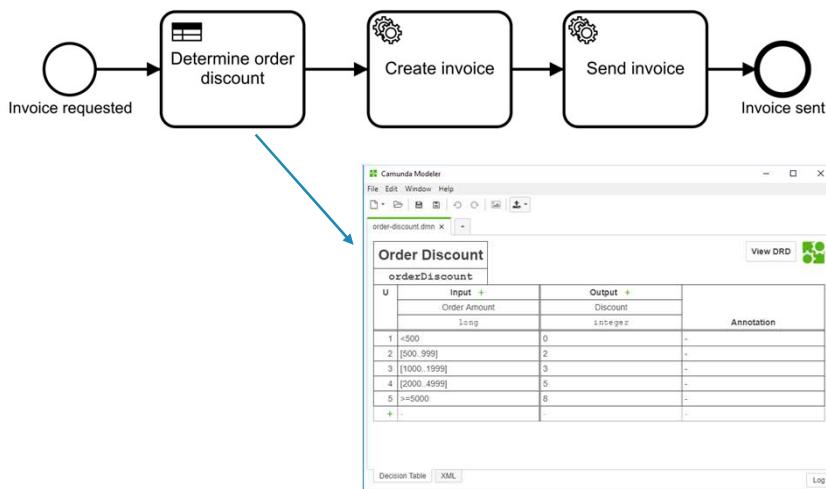
Type definition

Input parameter

Dish		Output		Annotation
U	Input	Output	+	
	Season string	How many guests integer		
1	"Fall"	<=8	guestCount	Enter simple FEEL expression or change to script.
2	"Winter"	<=8		
3	"Spring"	<=4		
4	"Spring"	[5..8]		
5	"Fall", "Winter", "Spring"	> 8		
6	"Summer"	-	"Light Salad and a nice Steak"	Hey, why not?!
+ -		-	-	-



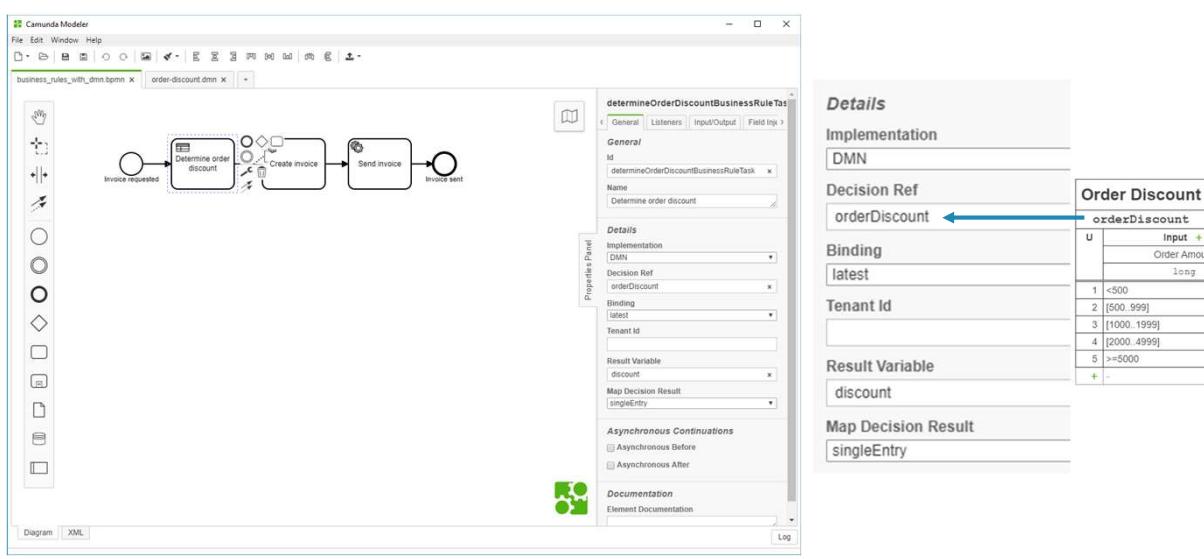
## Business Rule Task Reference a DMN Decision Table



265



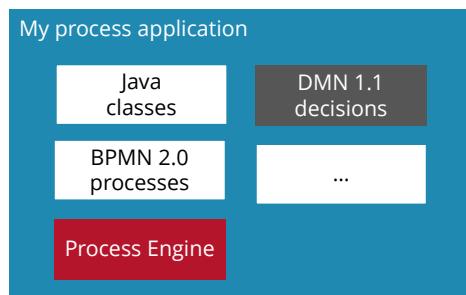
## Reference in Camunda Modeler



266



## DMN Is Part of the BPM Platform



267



## Decision Service in the Platform

```
DmnDecisionTableResult result = processEngine  
    .getDecisionService()  
    .evaluateDecisionTableByKey(decisionDefinitionKey, variables);
```

Also available as REST-Service

```
POST /decision-definition/key/aDecisionDefinitionKey/evaluate
```

Request body:

```
{  
    "variables" : {  
        "amount" : { "value" : 600, "type" : "Double" },  
        "invoiceCategory" : { "value" : "Misc", "type" : "String" }  
    }  
}
```

Response:

```
[  
    {  
        "result": { "value" : "management", "type" : "String", "valueInfo" : null }  
    }  
]
```

268



269

## JUnit for Decisions

```
@Test  
@Deployment(resources = {"myRule.dmn"})  
public void testWithProcessEngine() {  
    Map<String, Object> variables = withVariables("orderAmount", 1999.90);  
    DmnDecisionTableResult result = processEngine()  
        .getDecisionService()  
        .evaluateDecisionTableByKey("my_decision", variables);  
    assertThat(result.getFirstResult()).containsEntry("discount", 5);  
}
```

For decisions in the process engine

269



270

## Hit Policies

- Single
  - **U(nique)**: Only one rule can match
  - **A(ny)**: Multiple rules may match – must have same output
  - **P(riority)**: Rule with highest priority (output field) is selected
  - **F(irst)**: First matching rule (order in table!) is selected
- Multiple
  - **O(utput Order)**: List of rules in order of priority (output field)
  - **R(ule Order)**: List of rules in order of table
  - **Collect**: List of all hits without order, might be combined with operator + (sum), < (min), > (max), # (count)

*Italic hit policies are not yet supported by the Camunda DMN engine*

270



## Examples From the Spec

Applicant Risk Rating			
U	Applicant Age	Medical History	Applicant Risk Rating
1	> 60	good	Medium
2		bad	High
3	[25..60]	-	Medium
4	< 25	good	Low
5		bad	Medium

Student Financial Package Eligibility				
R	Student GPA	Student Extra-Curricular Activities Count	Student National Honor Society Membership	Student Financial Package Eligibility List
1	> 3.5	>= 4	Yes	20% Scholarship
2	> 3.0	-	Yes	30% Loan
3	> 3.0	>= 2	No	20% Work-On-Campus
4	<= 3.0	-	-	5% Work-On-Campus



## Map the Result to the Process

Mapper	Result	Is suitable for decision tables with
singleEntry	TypedValue	No more than one matching rule and only one output
singleResult	Map<String, Object>	No more than one matching rule and multiple outputs
collectEntries	List<Object>	Multiple matching rules and only one output
resultList	List<Map<String, Object>>	Multiple matching rules and multiple outputs





## Friendly Enough Expression Language (FEEL)

The screenshot shows the Camunda FEEL editor interface. On the left, there's a code editor with two examples:

```
[date and time(„2016-12-24T00:00:00“)  
..  
date and time(„2016-12-26T23:59:99“)]
```

To the right of the editor is a table titled "Simple Expressions" with columns for F, Season, Date, Number of guests, Children, Output, and Annotation. The table contains 9 rows of examples. A callout box points to the first row with the text: "Support for different „endpoint“ data types: number, string, boolean, time, date, date-time, time-duration." Another callout box points to the 7th row with the text: "Support for Comparison (<, <=, >, >=) and Ranges ([x..y]):". A third callout box points to the 9th row with the text: "Negation".

F	Season	Date	Number of guests	Children	Output	Annotation
1 -		[date and time("2016-12-24T00:00:00"), date and time("2016-12-26T23:59:99")]	-	-	"Christmas Dinner"	-
2 "Winter"	-	-	-	true	"Vegetables"	-
3 "Winter"	-	-	<=6	-	"Steak"	-
4 "Winter"	-	-	]6..10[	-	"Pizza"	Same as (6..10)
5 "Winter"	-	-	[10..12]	-	"Huge Pizza"	-
6 "Winter"	-	-	]12..[13..14]>14	-	"Family Deluxe Pizza"	-
7 "Spring", "Summer", "Fall"	-	-	-	true	"Salad"	-
8 not("Winter")	-	-	-	false	"Salad with Goat Cheese"	-
9 not("Spring", "Summer", "Fall")	-	-	-	-	"Vegetables"	It is Winter again :-)



## Community Extension FEEL-Scala

- Full support for FEEL
- Built-in functions (see <https://github.com/camunda/feel-scala/wiki>):
- Example expressions and functions:

```
applicant.monthly.income * 12  
  
if applicant.maritalStatus in ("M", "S") then "valid" else "not valid"  
  
sum( [applicant.monthly.repayments, applicant.monthly.expenses] )  
  
sum( credit_history[record_date > date("2011-01-01")].weight )  
  
some ch in credit_history satisfies ch.event = "bankruptcy"  
  
contains("foobar", "of")
```

<https://github.com/camunda/feel-scala>



275

## FEEL-Scala Usage

Maven dependency:

```
<dependency>
  <groupId>org.camunda.bpm.extension.feel.scala</groupId>
  <artifactId>feel-engine-plugin</artifactId>
  <version>1.6.2</version>
</dependency>
```

Spring Boot application:

```
@Configuration
public class BpmPlatformConfiguration {

    @Bean
    public static ProcessEnginePlugin feelScalaPlugin() {
        return new CamundaFeelEnginePlugin();
    }
}
```

JUnit engine configuration:

```
<property name="processEnginePlugins">
<list>
    <bean class="org.camunda.feel.CamundaFeelEnginePlugin"/>
</list>
</property>
```

275



276

## JUEL as Expression Language

[View DRD](#)

Tweet approval			
approveTweet			
F	Input + Tweet Conditions boolean	Output + -	Annotation
1	lastTweet.content == currentTweet.content	false	You cannot tweet the same content twice
2	currentTweet.content.contains('#ibm')	false	Please do not tweet about competition
3	currentTweet.content.matches('someRegEx')		
4	-		
+ -			

Rule

- ~ Add Above
- ~ Add Below
- ~ Remove
- ~ Copy
- ~ Cut
- ~ Paste Above
- ~ Paste Below

Input

- ~ Add Left
- ~ Add Right
- ~ Remove
- ~ Copy
- ~ Cut
- ~ Paste Left
- ~ Paste Right

Cell

- ~ Expression Language

juel

+ Add Description

276

## Cockpit for Decisions

The screenshot shows the Camunda Cockpit interface for managing decisions. The main title is "Cockpit for Decisions". Below it, the specific decision being managed is "Assign Approver Group". The interface displays the decision table structure with columns for Input, Output, and Annotation. The input column contains categories like "day-to-day expense", "sales", and "exceptional". The output column contains approver groups like "accounting", "sales", and "management". Annotations are present in the annotation column. On the left side, there is a sidebar with various metadata fields such as Instance ID, Definition Version, Definition ID, Definition Key, Definition Name, Tenant ID, Deployment ID, Process Instance ID, Case Instance ID, and Decision Requirements Definition.

277

## Live Editing of Decision Tables

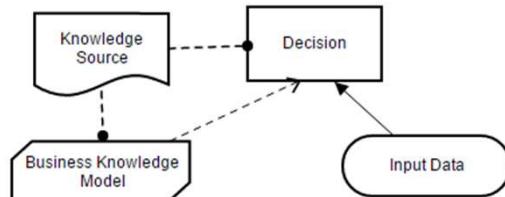
The screenshot shows the Camunda Cockpit interface for editing DMN files. The title bar indicates "Edit DMN". The main area displays the "Invoice Classification" decision table. The table has two columns: "Input" and "Output". The "Input" column includes "Invoice Amount" and "Invoice Category". The "Output" column includes "Classification". The table rows define classification rules based on invoice amount ranges. A blue banner labeled "Enterprise Feature" is overlaid on the top right of the interface. At the bottom, there are buttons for "Download changed version", "Cancel", and "Proceed". A status bar at the bottom shows "Powered by camunda BPM / v7.9.0-ee".

278

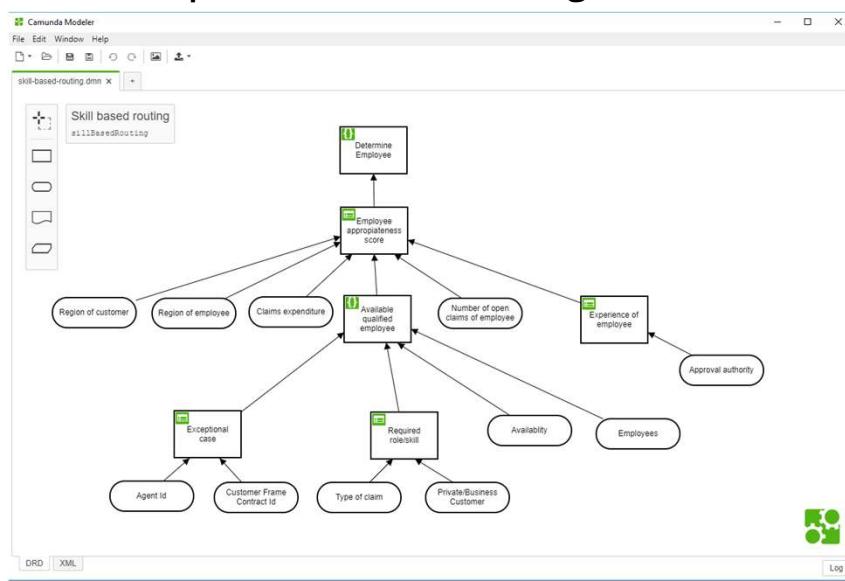


## Beyond (Or Before) Decision Tables

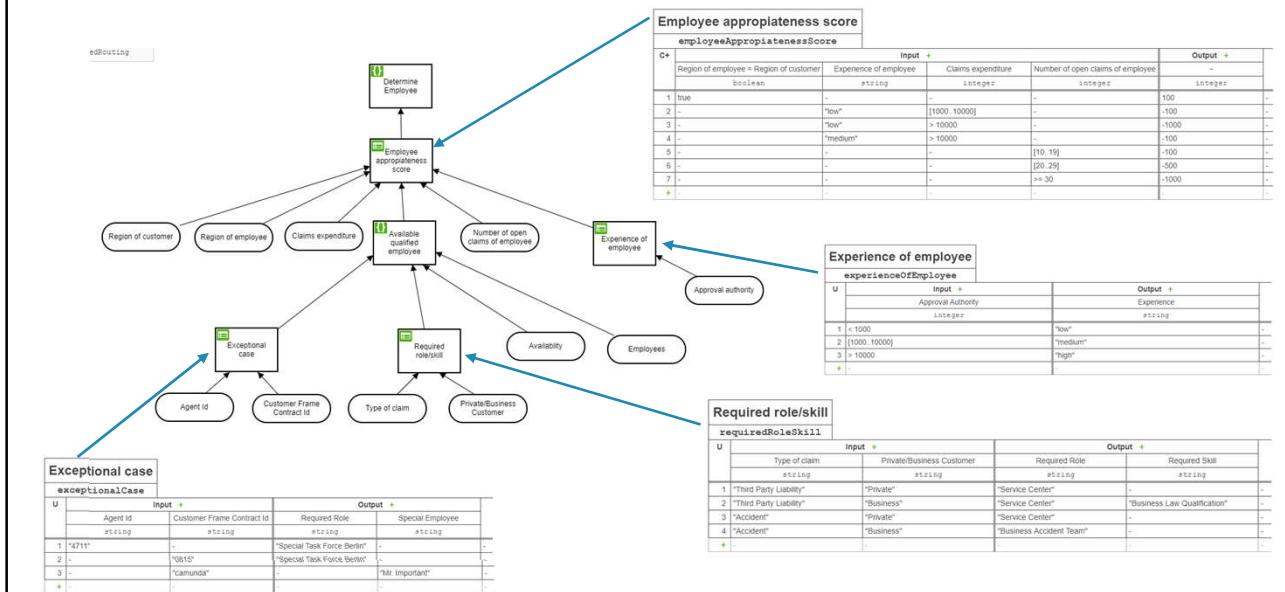
- DMN defines „Decision Requirements Diagram“
- Decision Requirements Diagrams will define the decisions [...], their interrelationships, and their requirements for decision logic
- Starting point for modeling complex decisions
- Elements of Decision Requirements Diagram



## DRD Example: Skill Based Routing



## Decisions Reference Decision Tables



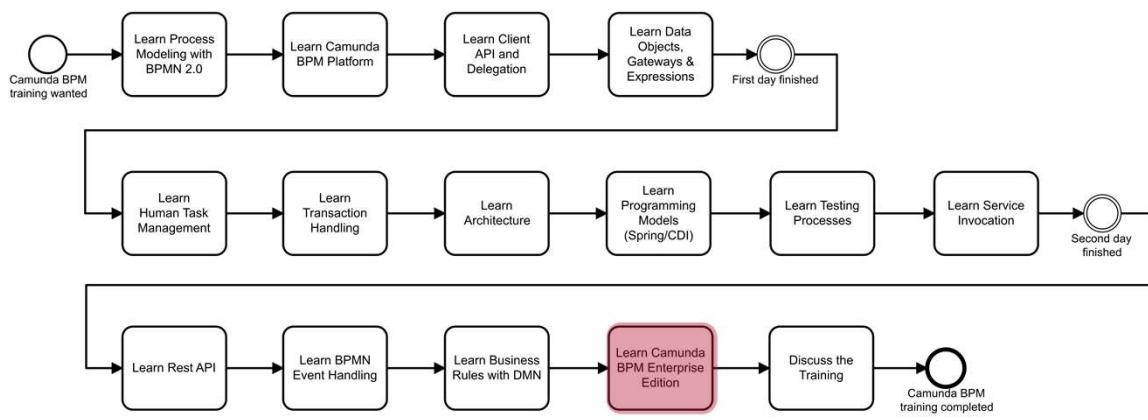
281

## Exercise 12



<https://training.camunda.com/java-dev/>  
user: java-dev password: camundarocks

282



283



284

## Enterprise Exclusive Cockpit Features

- Process Definition History
- Process Definition Heatmap
- Process Instance History
- Process Instance Migration
- Process Instance Modification
- Bulk Retry and Bulk Cancel
- Decision Table Live Editing
- Process Duration Report
- Redeploy Definitions

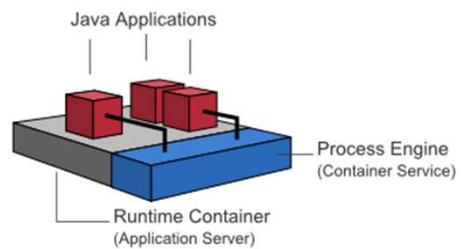
<https://camunda.com/bpm/enterprise/#cockpit>

284



## Runtime Support

- Integration with IBM WAS
- Integration with Oracle WLS



<https://camunda.com/bpm/enterprise/#as>



## SLA Based Support

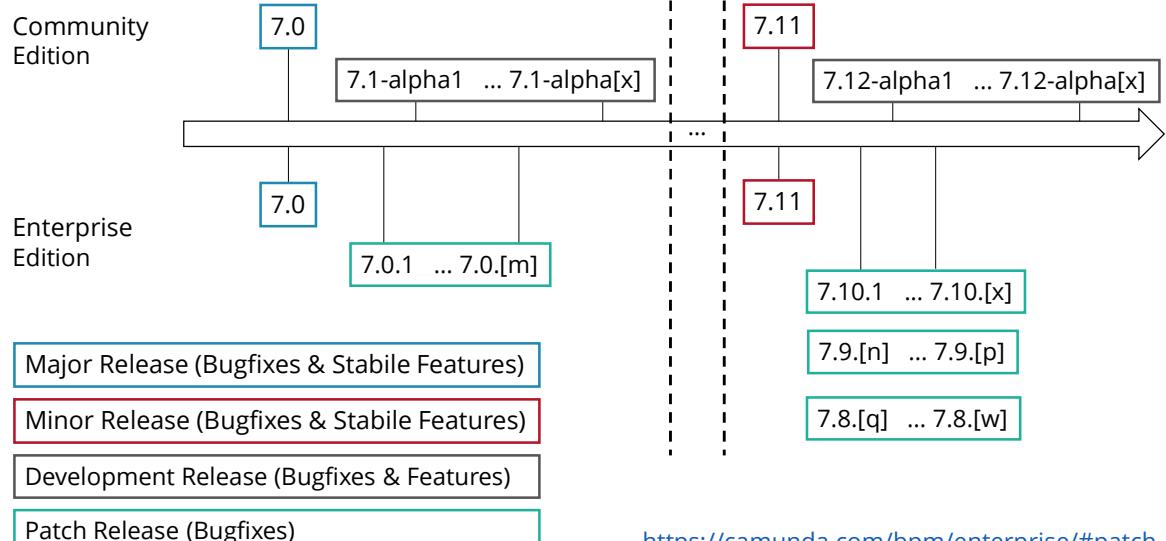
- Reliable support infrastructure with direct contact to core developers
- Service Level Agreements (SLA)

Level	Name	Description	Availability		Max. response time	
			Standard SLA	Advanced SLA	Standard SLA	Advanced SLA
L1	Blocker	Core components (i.e., process engine) of Camunda BPM do not work at all/produce critical errors that prevent usage in production mode.	8x5	24x7	8 business hours	2 hours
L2	Critical	Usage of Camunda BPM seriously affected, a workaround is urgently needed.	8x5		8 business hours	
L3	Help Request	Non-critical errors, Help Requests, Feature Requests.	8x5		16 business hours	

<https://camunda.com/bpm/enterprise/#support>



## Patch Releases



## Please Plan your Updates

- Check patch releases: Bugs that bother you?
- Minor release:
  - 6 Months before End of Support you will receive a mail
  - Check if you run the version mentioned here
  - If yes, start planning your update



289

## Optimize

The screenshot shows the Camunda Optimize interface with a 'Hiring Dashboard'. Key statistics displayed include:

- Total Applications 20...: 4,743
- New Hires 2018: 39
- New Applications per Day 2018: A bar chart showing daily application counts from Jan 1 to Dec 31.

Below these are two BPMN process diagrams illustrating 'Most executed steps' and 'Average Duration per Step'.

289



290

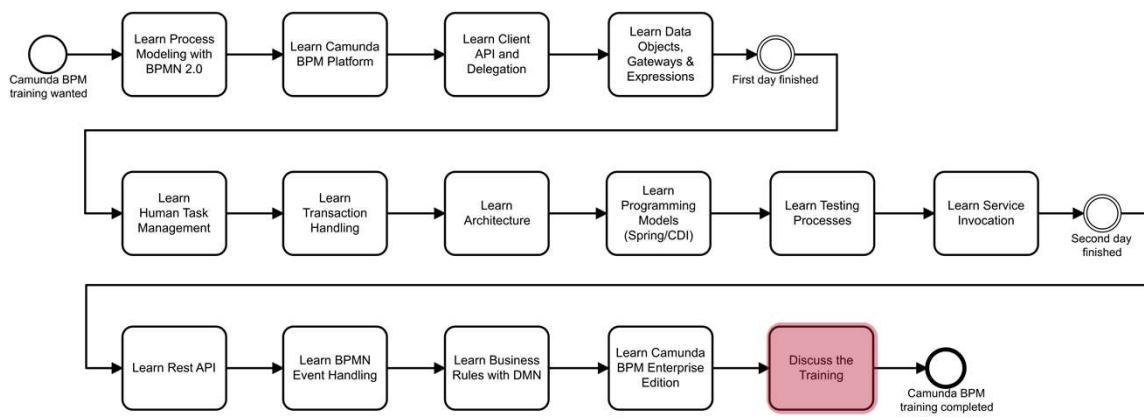
## References

The screenshot shows the Camunda Case Studies page. It includes a search bar and filters for 'All Industries' and 'All Regions'. Logos of organizations using Camunda are displayed:

- Haspa (Hamburger Savings Bank)
- Financial Industry Regulatory Authority (FINRA)
- Zalando
- total

<https://camunda.com/case-studies/>

290



291

## How to Get Updates From the Team?

Forum: <https://forum.camunda.org/>

Blog: <https://blog.camunda.org/>



Whitepapers: <https://camunda.com/learn/whitepapers/>

Events: <https://camunda.com/events/>

### Where to meet us

Whether it's at an exhibition booth, a local Meetup or our own event, we're always keen to have a chat with you so drop by and say hello!

#### DE Roadshow 2018

Meet our team in Germany, Switzerland and Austria.

[More Info and Dates](#)

#### CamundaCon

Camunda's annual 2-day user Conference in Berlin.

[CamundaCon](#)

#### Meetups

Local Meetups, organized by our amazing community.

[Meetups](#)

#### Conferences

Visiting a developer's conference? There's a good chance we're there.

[Conferences](#)

all categories > Latest New Unread (85) Top Categories

Topic	Category	Users
Process not found on Camunda platform after setup/deployment <span style="color:red;">• new</span>	Cockpit / Tasklist / Admin & Web	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Link to external documentation? <span style="color:red;">• new</span>	Modeler	G H I J K L M N O P Q R S T U V W X Y Z
Get the task instance from a execution listener <span style="color:red;">• new</span>	Process Engine	B C D E F G H I J K L M N O P Q R S T U V W X Y Z
REST API request to fill a form	Process Engine	B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Bug using Camunda Data	Process Engine	B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Minor typo in BPMN tutorial	Meta	P
JsonLogic in Camunda <span style="color:red;">• new</span>	Process Engine	E
Single-Sign-On in Camunda	Cockpit / Tasklist / Admin & Web	P Q R E J
Camunda - Getting certificate error while calling an https REST service through connector <span style="color:red;">• new</span>	Modeler	D G H I K L M N O P Q R S T U V W X Y Z

Follow us on Twitter: @camundabpm



292



## Camunda training offering

### Overview trainings

#### Camunda BPM Overview (1 day)

- Introduction into BPMN
- Hands on Camunda BPM
- Monitoring & Reporting
- How to start with Camunda

#### OCEB training (2 days)

- Prepare for the OCEB BPM certification

### Modeling trainings

#### BPMN (3 days)

- BPMN in detail
- Implementing BPM-projects with BPMN
- BPMN real world examples
- BPMN in context of DMN

#### DMN (1 day)

- DMN in detail
- Decision design
- Complex decisions with DRDs
- DMN in context of BPMN

### Development trainings

#### Camunda BPM for developers (3 days)

- Introduction BPMN
- Camunda BPM Architecture
- Automating processes with Camunda
- Testing, deployment, versioning

#### Camunda BPM DevOps (2 days)

- Camunda BPM installation
- Monitoring & alarming
- Process version migration
- Advanced DevOps topics

You'll find further information under  
[camunda.com/de/services/training](http://camunda.com/de/services/training)

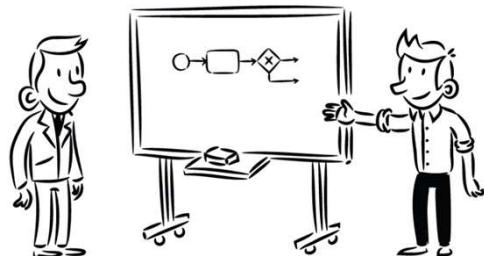
293

294



## Want to Learn More BPMN?

- 3-day training
- Learn in-depth BPMN 2.0
  - Understand BPMN
  - Apply BPMN
  - Introduce BPMN



294



## Feedback

Please fill out this form:

<https://camunda.com/services/training/feedback/>

The screenshot shows a web browser window with the URL <https://camunda.com/services/training/feedback/>. The page title is "Training Feedback". The form contains the following fields:

- \* 1. Trainer: A dropdown menu.
- \* 2. Training: A dropdown menu showing "Camunda BPM Basic".
- \* 3. Date of training: A date input field with a placeholder "Date / Time" and a separate section for "DD", "MM", and "YYYY".
- 4. What is your role?: An input field.
- \* 5. Training Management: A horizontal scale with four options: "Excellent", "Good", "Not Great", and "Terrible".