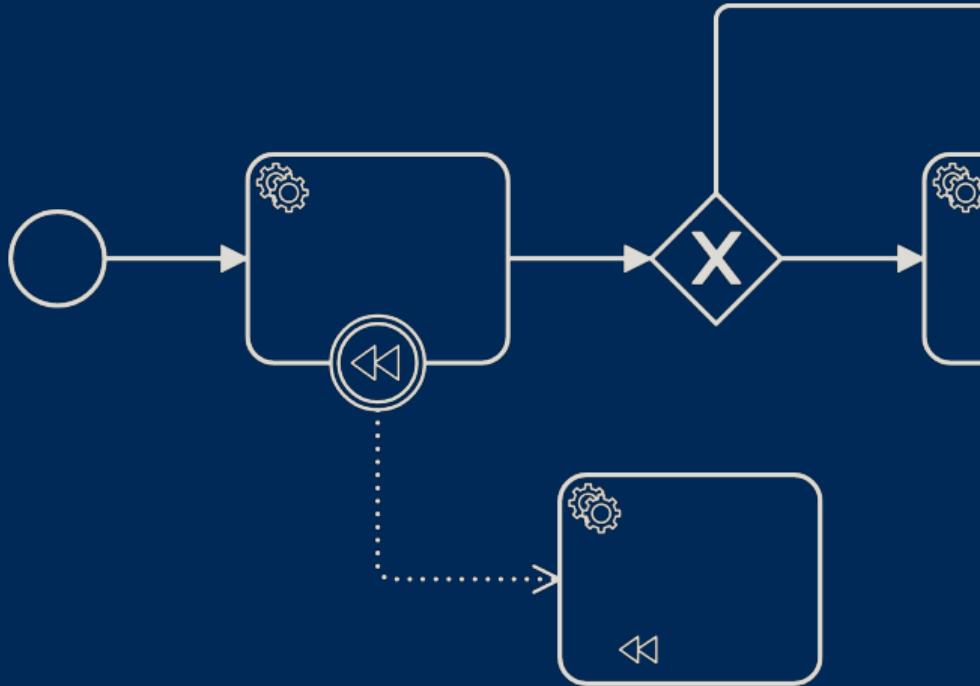




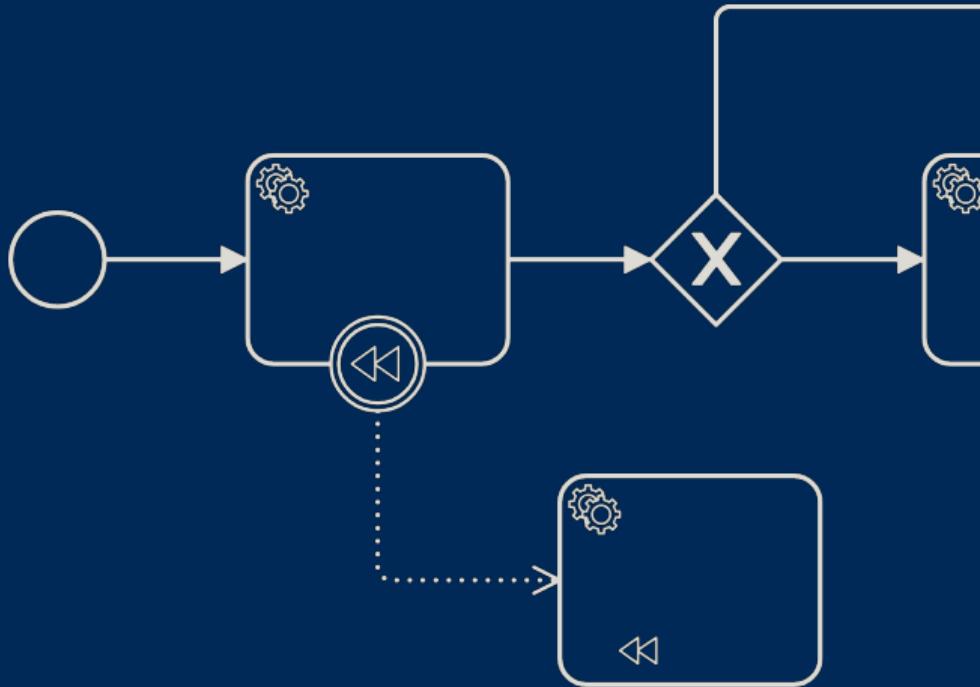
# Camunda Chapter: Finland

Joined

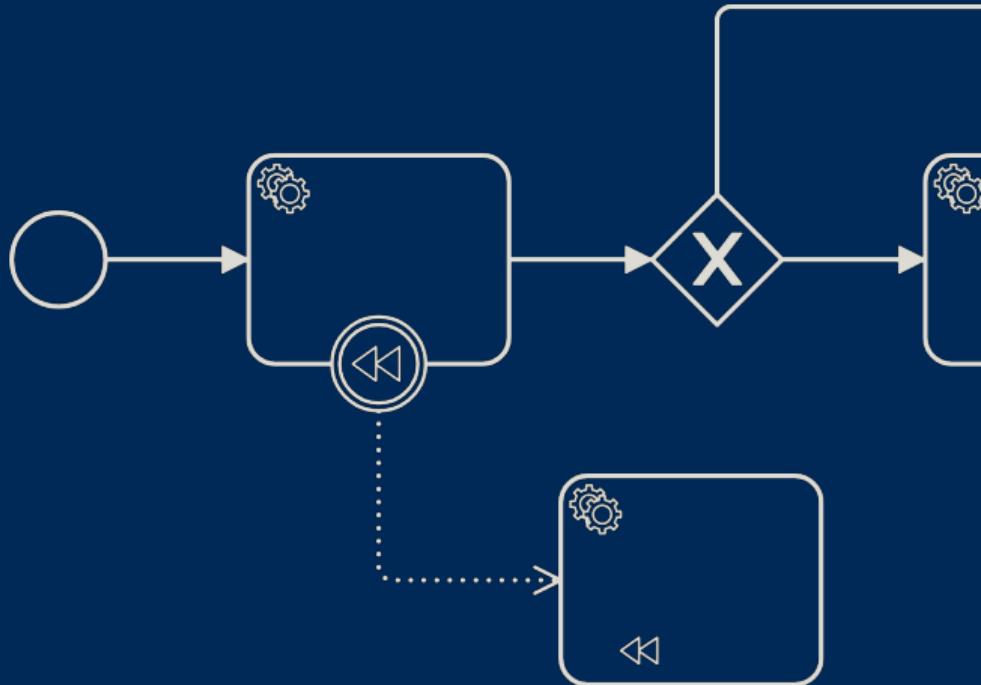
# Camunda



# Camunda Chapter:



# Camunda Chapter: Finland



# Camundan BPMN-teknologia osana avoimen lähdekoodin ratkaisuja

---

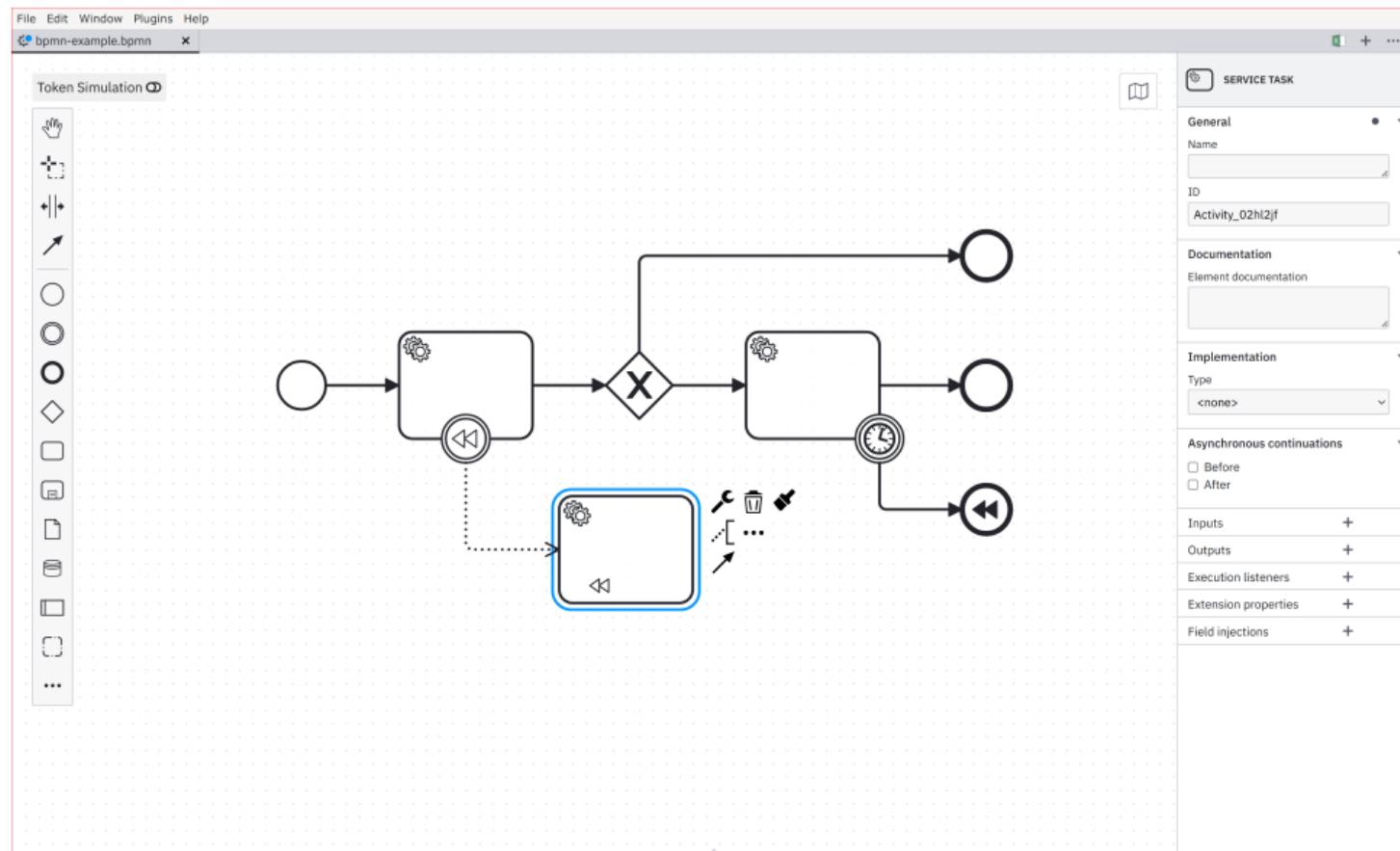
Asko Soukka

20.03.2024

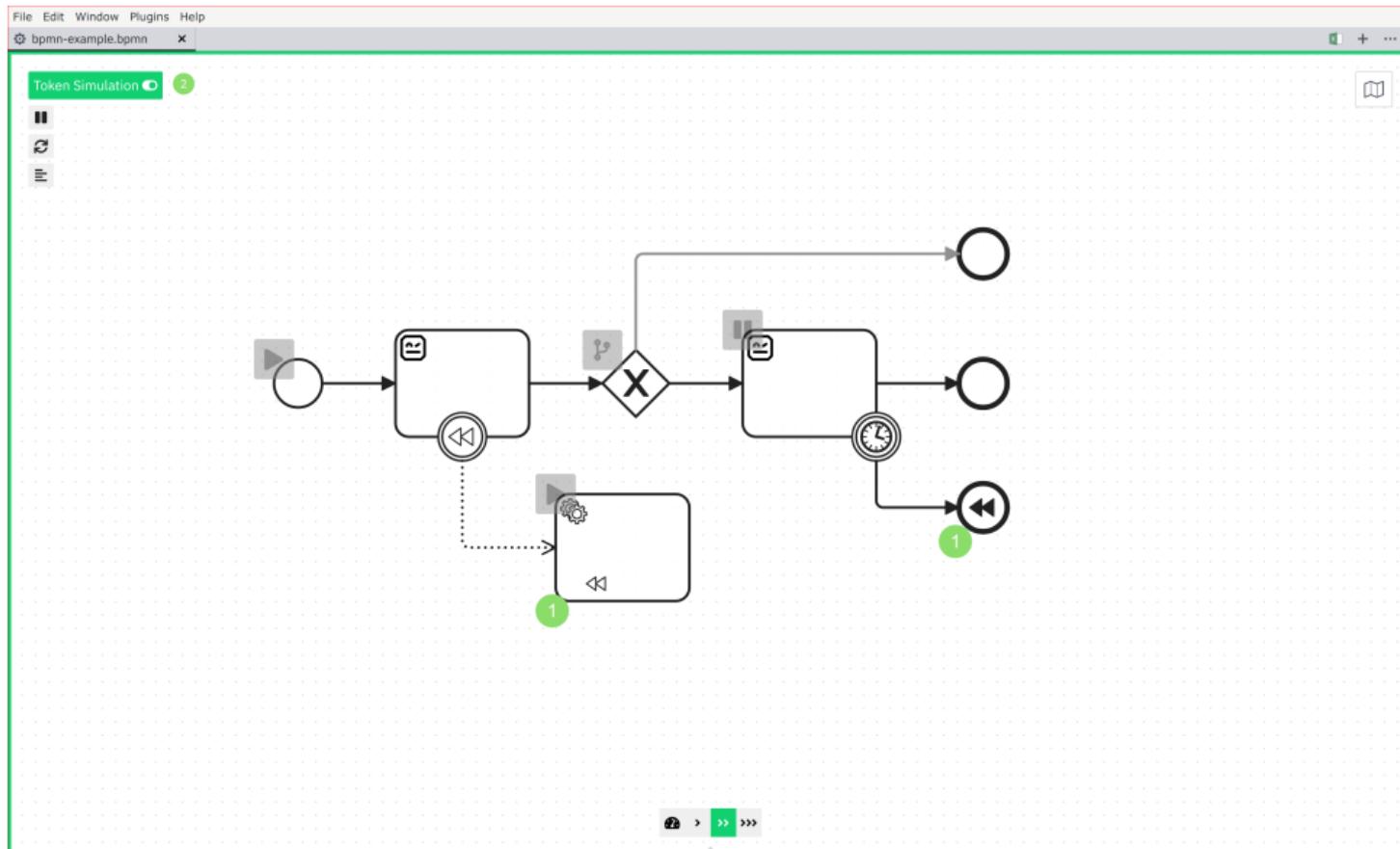
# Camunda Modeler

---

# Camunda Modeler



# Camunda Modeler Plugins



# Camunda Modeler

- License: MIT
- Plugins:
  - Token Simulation Plugin
  - Transaction Boundaries Plugin
  - Embedded Comments Plugin
  - Property Info Plugin
  - Resize Tasks Plugin

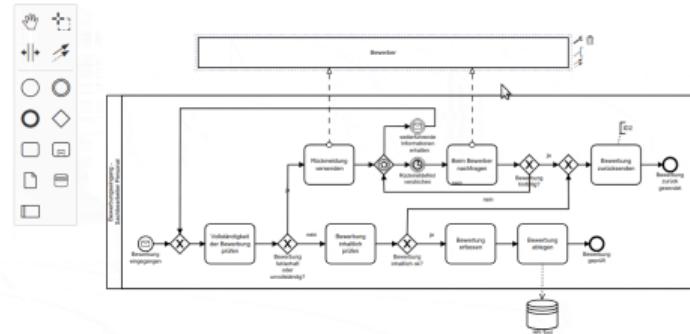
<https://github.com/camunda/camunda-modeler-plugins>

**bpmn.io**

---

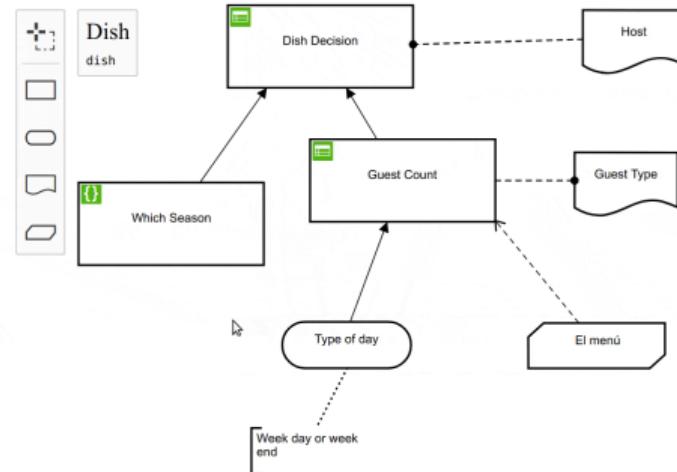
## Web-based tooling for BPMN, DMN and Forms

- bpmn-js



## Web-based tooling for BPMN, DMN and Forms

- bpmn-js
- dmn-js



## Web-based tooling for BPMN, DMN and Forms

- bpmn-js
- dmn-js
- form-js

The screenshot illustrates the bpmn.io web-based tooling interface. On the left, a sidebar titled "FORM ELEMENTS LIBRARY" lists various input types: Text Field, Number, Checkbox, Radio, Select, Text, and Button. In the center, a form titled "File an Invoice" is displayed with fields for "Creditor\*", "Invoice Number\*", and "Amount\*". Below the form is a "Submit" button. To the right, a preview panel shows the form's visual representation: a header "ABC TEXT # File an ...", a "General" section with the form fields, and a note "Use HTML or Markdown to format." At the bottom right of the preview panel is the "BPMN.io" logo.

## Web-based tooling for BPMN, DMN and Forms

- bpmn-js
- dmn-js
- form-js
- nikku/feelin

FEEL Playground

Code Expression ↗

```
1 for
2   fruit in [ "apple", "banana" ], vegetable in vegetables
3 return
4   { ingredients: [ fruit, vegetable ] }
```

Syntax Tree

```
Expression [0, 109]
  ForExpression [0, 109]
    InExpression [0, 62]
      InExpression [0, 37]
        Name [0, 11]
        IterationContext [19, 37]
          Like [19, 24]
            StringLiteral [17, 24]
            StringLiteral [20, 35]
        InExpression [39, 62]
        Name [51, 62]
        IterationContext [52, 62]
        VariableName [52, 62]
      Context [72, 109]
      ContextEntry [74, 107]
        Key [74, 63]
        Name [74, 85]
      List [87, 107]
      VariableName [89, 94]
      VariableName [90, 105]
```

Input

```
{ "vegetables": [ "garlic", "tomato" ],
  "Mike's age": 35
}
```

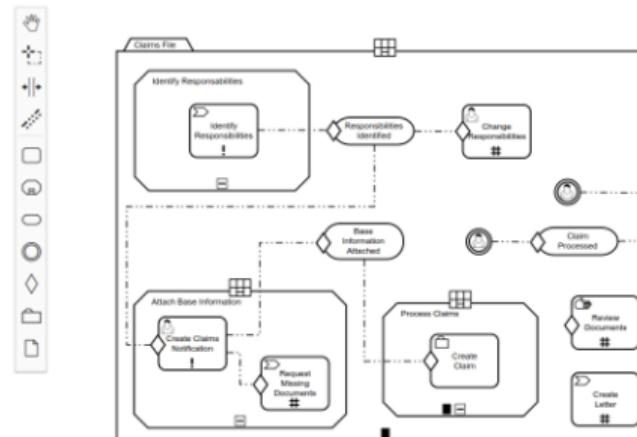
Output

```
[
  {
    "ingredients": [
      "apple",
      "garlic"
    ]
  },
  {
    "ingredients": [
      ...
    ]
  }
]
```

Define your input variables as a JSON object literal.  
Re-computes once you change code or input.

## Web-based tooling for BPMN, DMN and Forms

- bpmn-js
- dmn-js
- form-js
- nikku/feelin
- cmmn-js



# bpmn.io + JupyterLab

## BPMN, DMN and Forms in Jupyter notebooks

- jupyterlab-bpmn

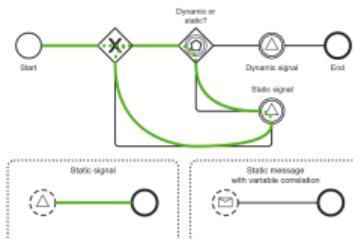
### Signal broadcast

Signal is the recommended way to broadcast event to multiple processes.

```
In [10]: signal = {
    "name": "Static signal",
    "variables": [
        "notId": {
            "type": "String",
            "value": "Hello World!"
        }
    ]
}

In [11]: _ = post(f"{base_url}/signal", json=signal)

In [12]: render(dict(activities=get(f"{base_url}/history/activity-instance", params={"processDefinitionId": definition_id}).json()))
```



# bpmn.io + JupyterLab

## **BPMN, DMN and Forms in Jupyter notebooks**

- jupyterlab-bpmn
- jupyterlab-dmn

# bpmn.io + JupyterLab

## BPMN, DMN and Forms in Jupyter notebooks

- jupyterlab-bpmn
- jupyterlab-dmn
- jupyterlab-form-js

```
},
  "schemaVersion": 5
}

[3]: form = jupyterlab_form_js.FormJSWidget(
    schemaVersion,
    data={"field_0jgi51a": 10}
)
out = widgets.Output()
def handle_submit(f, data, errors):
    with out:
        clear_output()
        display(HTML("<div>And the data being submitted is...</div>"))
        display(JSON(data, expanded=True))
form.on_submit(handle_submit)
display(form, out)
```

Number  
10

Checkbox

Button

[BPMN.IO](#)

And the data being submitted is...

```
▼ root:
  field_0jgi51a: 10
  field_0i78dhv: true
```

Filter...

# Sphinx documentation

[Open Automation Playground](#)

Search docs

- Robocon 2023 Workshop
- Playground introduction
- BPMN basics by examples
- BPMN modeling exercises
- Camunda basics for execution

Camunda execution exercises

- Sole service task
- Add exclusive gateway
- Handle BPMN error
- Embedded sub-process wrapping

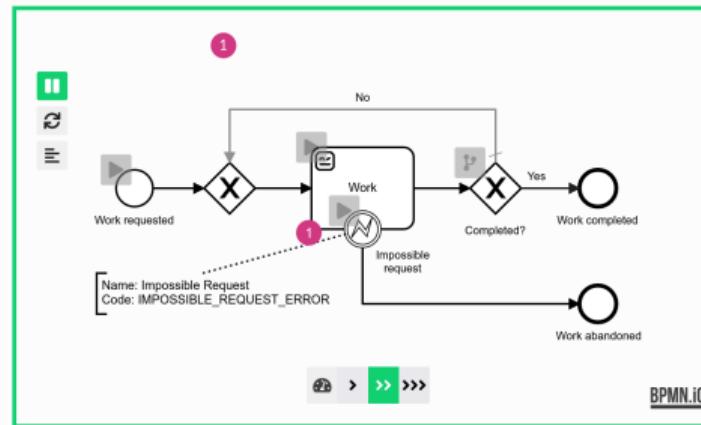
Multi-instance sub-process

- DMN, user tasks and forms
- DMN exercise with forms
- Service tasks for Robots
- Robocorp Code in action
- Exercise: PDF creation robot
- Exercise: Email sending robot
- Exercise: API consuming robot
- Exercise: Spreadsheet robot
- Exercise: Exception handling
- Extra: Classification training
- About Camunda Platforms
- Summary: Hello World

## Handle BPMN error

Let's then refactor a little and add a new feature:

1. Re-use the model from the previous exercise.
2. Remove end event *Work not completed* and route the path back to the service task instead.
3. Add a error boundary event to service task with a path to a new *Work abandoned*.
4. Configure the error boundary event to expect error with some specific error code.
5. Deploy process to Zeebe, start an instance of it and **Throw Error** with the code you just configured on its service task to reach the new *Work abandoned*.



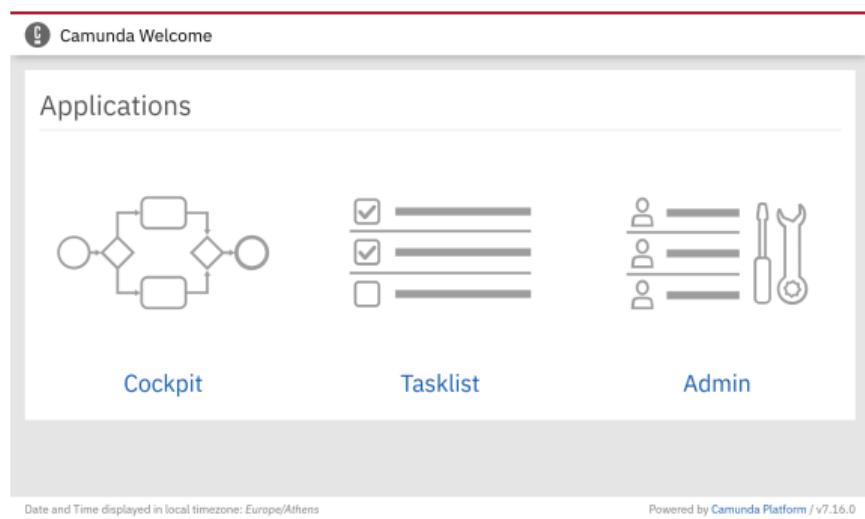
# Camunda 7

---

# Camunda 7

## Camunda 7 Community Edition

- License: Apache 2.0
- "Open Core"
- Distributions:
  - Camunda Run
  - Tomcat
  - Docker
  - SpringBoot
  - Micronaut (unofficial)



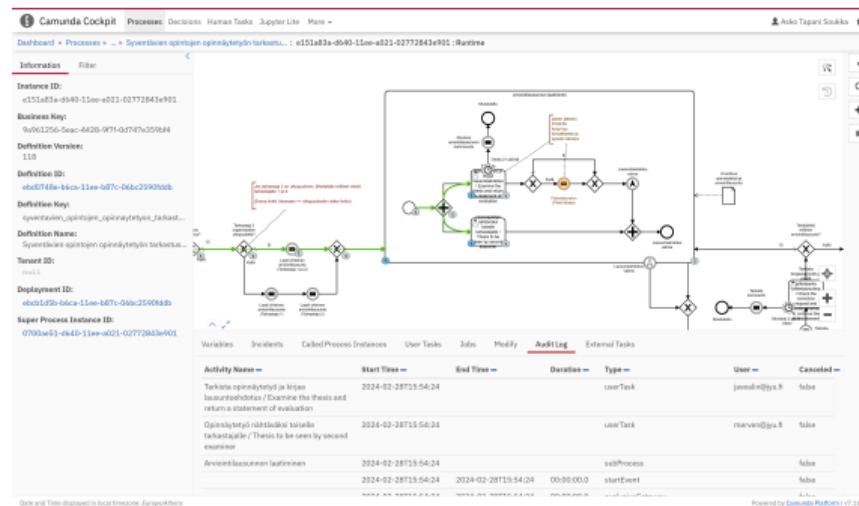
# Camunda 7 Plugins

## Camunda 7 Webapps Plugins

- Minimal History
- JupyterLite
- form.io-forms

## Other

- Command



# Camunda 7 Plugins

## Camunda 7 Backend Plugins

- Keycloak Identity Provider
- GraphQL API

## Other

- External Task Clients
- Vasara "Message Gateway"
- Vasara "Carrot RCC"

# Camunda 7 Platforms

## Camunda 7 Based Platforms

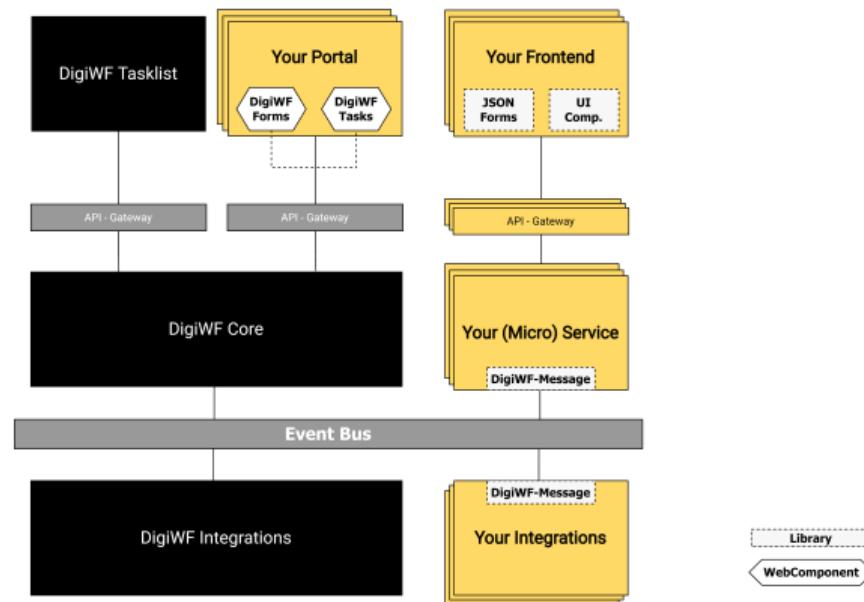
- WKS Platform

The screenshot displays the wksPower platform interface, specifically the 'Contractor On Boarding Case' section. The left sidebar includes links for Dashboard, Cases (selected), Generic Case, Motion Detected, Contractor On Boarding Case (selected), Tasks, Records, Management, System, and Case Life Cycle. The main content area shows a 'Create' button and a grid of cards. The grid has two columns and four rows. The first row contains cards for cases 87532 and 78897, both labeled 'Closed'. The second row contains cards for cases 29181 and 58719, both labeled 'Closed'. The third row contains cards for cases 29337 and 77022, both labeled 'Closed'. The fourth row contains cards for cases 49413 and 247, both labeled 'Closed'. The top right corner shows a user profile for 'demo'.

# Camunda 7 Platforms

## Camunda 7 Based Platforms

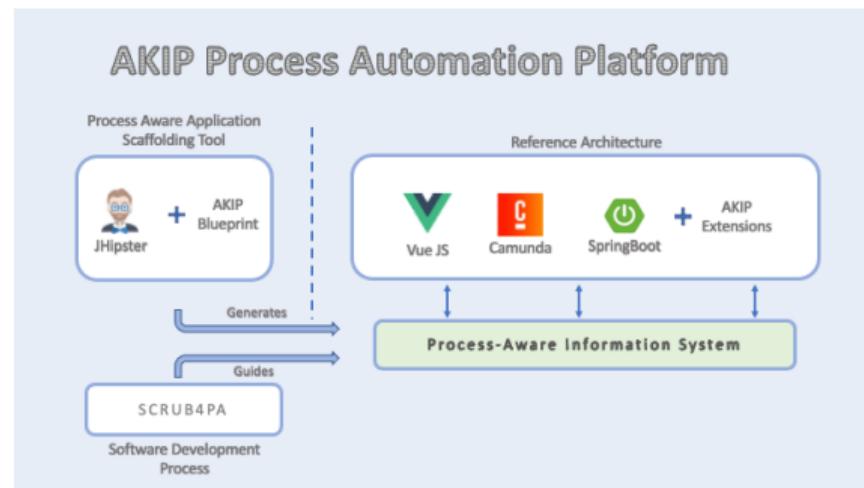
- WKS Platform
- digiWF



# Camunda 7 Platforms

## Camunda 7 Based Platforms

- WKS Platform
- digiWF
- AgileKIP (on-hold)



# Camunda 7 Platforms

## Camunda 7 Based Platforms

- WKS Platform
- digiWF
- AgileKIP (on-hold)
- Digital State (unknown)
- Vasara (in-progress)
- Plone-integration (on-hold)

The screenshot shows a web interface for managing interview invitations. At the top, there's a header with a search bar and navigation links for 'Tehdästä', 'Prosessit', and 'Tietosuoja-ilmoitus'. Below the header, a message box displays: 'Tutkintoon johdannaisiin opintojen hakemusten avointi: Kysymyksien avointi, haastattelujen makiautomaatioiden avointi: Kutsu haastatteluun valitut'.

The main content area is titled 'Kutsu haastatteluun valitut' and contains two tables:

Lähettä haastattelujen valitulle kutsut.					
Sukunimi	Etunimet	Sähköposti	Pisteet yhteenä	Kommennetti aikemuksesta	Hakija kutsutaan haastatteluun
Reijas	Risto	m@gmail.com	10	Henno hakija	Kyllä
Ruohonen	Matti Tapio	mtt@gmail.com	11		Kyllä
Ruohonen	Teppo Ilmari	teppi@gmail.com	8		Kyllä

Hakijat, joita ei kutsuta haastatteluun					
Sukunimi	Etunimet	Sähköposti	Pisteet yhteenä	Kommennetti aikemuksesta	Hakija kutsutaan haastatteluun
Mäkinen	Jarkko Antero	lahti@gmail.com	10	Elävät minni vuosimaksua	Ei

At the bottom of the page, there are buttons for 'TALLENNÄ MUUTOKSET', 'LAHETA ETEENPÄIN', and 'POISTU LOMAKKEELTA'. The footer features the Jyväskylä University logo and the text 'JYVÄSKYLÄN YLIOPISTO UNIVERSITY OF JYVÄSKYLÄ'.

# Camunda 7 EOL

- Camunda 7 CE will EOL (end of life) in October 2025
- final release (v7.24), happening on Oct 14, 2025
- GitHub repo will be archived, deprecated
- artifacts on Maven Central will no longer be maintained

*Camunda 7 Enterprise Edition (EE) will EOL in April 2030*

## Zeebe (Camunda 8)

---

# Camunda 8 Licensing

- Zeebe – Zeebe Community License 1.1
- Connector SDK – Apache 2.0
- Camunda Webapps – proprietary
- Camunda Connectors – proprietary

# Camunda 7 vs. Zeebe

## Camunda 7

- Apache 2.0
- Full stack
- Standalone
- Transactional
- Integratable
- REST API + long-poll
- Customizable

## Zeebe

- Custom
- Only engine
- Cluster
- Event-driven
- External service
- gRPC + exporters
- Opinionated

# Zeebe OSS ecosystem

<https://github.com/camunda-community-hub>

- interceptors
- exporters
- task clients
- "connectors"
- APIs
- webapps

# Zeebe OSS picks

- zeebe-keycloak-interceptor
- zeebe-jwt-interceptor
- zeebe-redis-exporter
- zeebe-simple-monitor
- zeeqs (GraphQL API)
- zeebe-play
- Parrot RCC
- datasette.io example
- Connector SDK Connectors

Zeebe Simple Monitor

New Instance

Key 2251799813685249

BPMN demo-process

process id

Version 1

# active 1

# ended 0

task 1 task 2 task 3

Instances Message Subscriptions Timers

1 instances

Workflow Instance Key	State	Start Time
2251799813687699	Active	2019-07-11T07:48:42.907Z

Previous Next

localhost:8080/views/.../2251799813687699

The screenshot shows the Zeebe Simple Monitor web application. At the top, there's a navigation bar with links for Workflows, Instances, Incidents, Jobs, and Messages. A 'New Instance' button is located in the top right. Below the navigation, there's a sidebar with monitoring metrics: Key (2251799813685249), BPMN (demo-process), process id, Version (1), # active (1), and # ended (0). To the right of the sidebar is a process diagram consisting of three sequential tasks labeled 'task 1', 'task 2', and 'task 3'. Task 1 has a green rounded rectangle above it with the text '0/1'. Task 2 has a green rounded rectangle above it with the text '1/1'. Task 3 has a green rounded rectangle above it with the text '0/0'. Below the process diagram is a table titled 'Instances' showing one row with the workflow instance key '2251799813687699', state 'Active', and start time '2019-07-11T07:48:42.907Z'. At the bottom of the table are 'Previous' and 'Next' buttons. The footer of the page shows the URL 'localhost:8080/views/.../2251799813687699'.

# Zeebe Playground (2022)

**Open Automation Playground**

<https://datakurre.github.io/automation-playground>