

# Flowable Activities

## Flowable Activities

activiti

multiInstanceLoopCharacteristics

```
<multiInstanceLoopCharacteristics isSequential="false|true">
  ...
</multiInstanceLoopCharacteristics>
```

**isSequential**属性表示节点是进行 顺序执行还是并行执行。

```
isSequential=true
```

```
isSequential=false
```

```
<userTask id="B001" name="会签环节" activiti:assignee="${per}">
  <multiInstanceLoopCharacteristics isSequential="false" activiti:collection="pers" activiti:elementVariable="per">
    </multiInstanceLoopCharacteristics>
  </userTask>
```

|activiti:collection| pers

|activiti:elementVariable|      |activiti:assignee|

```
<multiInstanceLoopCharacteristics isSequential="false" activiti:collection="pers" activiti:elementVariable="per">
  <completionCondition>${mulitiInstance.completeTask(execution)}</completionCondition>
</multiInstanceLoopCharacteristics>
```

true

```
completionCondition    juel        multilInstance    spring    spring    bean id    key.
```

1.nrofInstances

2.nrofActiveInstances

3.nrofCompletedInstances

1.

```

/**
 *
 * @author huan
 */
public class AssgineeMultiInstancePer implements JavaDelegate {
    @Override
    public void execute(DelegateExecution execution) throws Exception {
        System.out.println("      .");
        execution.setVariable("pers", Arrays.asList(" ", " ", " ", " "));
    }
}

```

## 2. **spring** **Serializable**

```

/**
 *
 * @author huan
 */
public class MulitiInstanceCompleteTask implements Serializable {
    private static final long serialVersionUID = 1L;
    public boolean completeTask(DelegateExecution execution) {
        System.out.println("      " + execution.getVariable("nrOfInstances") + "      " +
execution.getVariable("nrOfActiveInstances") + " - " + "      " +
execution.getVariable("nrOfCompletedInstances"));
        System.out.println("I am invoked.");
        return false;
    }
}

```

## 3.

```

/**
 *
 * @author huan
 */
public class TestLinstener implements TaskListener {
    private static final long serialVersionUID = -5754522101489239675L;
    @Override
    public void notify(DelegateTask delegateTask) {
        System.out.print(delegateTask.getId() + " - " + delegateTask.getProcessInstanceId() +

```

```

" - " + delegateTask.getEventName() + " - " + delegateTask.getTaskDefinitionKey());
    }
}

```

4.

```

/**
 *
 * @author huan
 */
public class TestMultiInstance {
    @Test
    public void testProcess() {
        ProcessEngine processEngine = ProcessEngines.getDefaultProcessEngine();
        RepositoryService repositoryService = processEngine.getRepositoryService();
        RuntimeService runtimeService = processEngine.getRuntimeService();
        TaskService taskService = processEngine.getTaskService();
        Deployment deploy = repositoryService.createDeployment()//
            .name(" ")//
            .addInputStream("multiInstances.bpmn",
this.getClass().getResourceAsStream("multiInstances.bpmn"))//
            .addInputStream("multiInstances.png",
this.getClass().getResourceAsStream("multiInstances.png"))//
            .deploy();

        System.out.println(deploy.getId() + " " + deploy.getName());
        Map<String, Object> variables = new HashMap<String, Object>();
        variables.put("mulitiInstance", new MulitiInstanceCompleteTask());
        ProcessInstance pi =
runtimeService.startProcessInstanceByKey("multiInstances", variables);
        System.out.println(pi.getId() + " " + pi.getActivityId());

        Task task1 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
        System.out.println(task1.getId() + " - " + task1.getAssignee() + " - " +
task1.getProcessInstanceId() + " - " + task1.getProcessDefinitionId());

        Task task2 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
        System.out.println(task2.getId() + " - " + task2.getAssignee() + " - " +
task2.getProcessInstanceId() + " - " + task2.getProcessDefinitionId());

        Task task3 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();

```

```

System.out.println(task3.getId() + " - " + task3.getAssignee() + " - " +
task3.getProcessInstanceId() + " - " + task3.getProcessDefinitionId());

    Task task4 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
    if (task4 != null) {
        System.out.println(task4.getId() + " - " + task4.getAssignee() + " - " +
task4.getProcessInstanceId() + " - " + task4.getProcessDefinitionId());
    }
    Task task5 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
    System.out.println(task5);
    taskService.complete(task1.getId());
    taskService.complete(task2.getId());
    taskService.complete(task3.getId());
    Task task6 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
    System.out.println(task6);
    taskService.complete(task4.getId());
    Task task7 =
taskService.createTaskQuery().processInstanceId(pi.getId()).taskAssignee(" ").singleResult();
    System.out.println(task7);
    taskService.complete(task7.getId());
    ProcessInstance processInstance =
runtimeService.createProcessInstanceQuery().processInstanceId(pi.getId()).singleResult();
    if (null == processInstance) {
        System.out.println("    .");
    }
}
}

```

5.

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:activiti="http://activiti.org/bpmn"
xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
xmlns:omgdc="http://www.omg.org/spec/DD/20100524/DC"
xmlns:omgdi="http://www.omg.org/spec/DD/20100524/DI"
typeLanguage="http://www.w3.org/2001/XMLSchema"

```

```

expressionLanguage="http://www.w3.org/1999/XPath"
targetNamespace="http://www.activiti.org/test">
  <process id="multiInstances" name="      " isExecutable="true">
    <startEvent id="startevent1" name="Start"></startEvent>
    <sequenceFlow id="flow1" sourceRef="startevent1" targetRef="A001"></sequenceFlow>
    <serviceTask id="A001" name="      "
activiti:class="com.huan.activiti.liuyang. .AssgineeMultiInstancePer"></serviceTask>
    <userTask id="B001" name="      " activiti:assignee="{per}">
      <extensionElements>
        <activiti:taskListener event="complete"
class="com.huan.activiti.liuyang. .TestLinstener"></activiti:taskListener>
      </extensionElements>
      <multiInstanceLoopCharacteristics isSequential="false" activiti:collection="pers"
activiti:elementVariable="per">

<completionCondition>${mulitiInstance.completeTask(execution)}</completionCondition>
      </multiInstanceLoopCharacteristics>
    </userTask>
    <sequenceFlow id="flow2" sourceRef="A001" targetRef="B001"></sequenceFlow>
    <userTask id="C001" name="      " activiti:assignee="      "></userTask>
    <sequenceFlow id="flow3" sourceRef="B001" targetRef="C001"></sequenceFlow>
    <endEvent id="endevent1" name="End"></endEvent>
    <sequenceFlow id="flow4" sourceRef="C001" targetRef="endevent1"></sequenceFlow>
  </process>
  <bpmndi:BPMNDiagram id="BPMNDiagram_multiInstances">
    <bpmndi:BPMNPlane bpmnElement="multiInstances" id="BPMNPlane_multiInstances">
      <bpmndi:BPMNShape bpmnElement="startevent1" id="BPMNShape_startevent1">
        <omgdc:Bounds height="35.0" width="35.0" x="100.0" y="240.0"></omgdc:Bounds>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="A001" id="BPMNShape_A001">
        <omgdc:Bounds height="71.0" width="117.0" x="190.0" y="222.0"></omgdc:Bounds>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="B001" id="BPMNShape_B001">
        <omgdc:Bounds height="55.0" width="105.0" x="380.0" y="230.0"></omgdc:Bounds>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="C001" id="BPMNShape_C001">
        <omgdc:Bounds height="55.0" width="105.0" x="561.0" y="230.0"></omgdc:Bounds>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape bpmnElement="endevent1" id="BPMNShape_endevent1">
        <omgdc:Bounds height="35.0" width="35.0" x="740.0" y="240.0"></omgdc:Bounds>

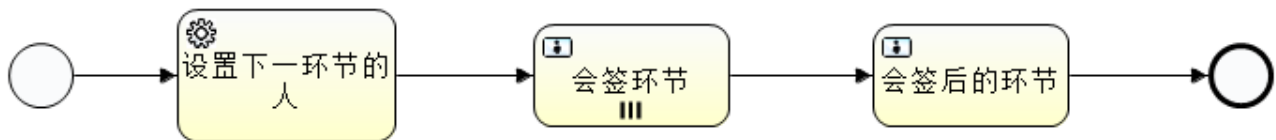
```

```

</bpmndi: BPMNShape>
<bpmndi: BPMNEdge bpmnElement="flow1" id="BPMNEdge_flow1">
  <omgdi: waypoint x="135.0" y="257.0"></omgdi: waypoint>
  <omgdi: waypoint x="190.0" y="257.0"></omgdi: waypoint>
</bpmndi: BPMNEdge>
<bpmndi: BPMNEdge bpmnElement="flow2" id="BPMNEdge_flow2">
  <omgdi: waypoint x="307.0" y="257.0"></omgdi: waypoint>
  <omgdi: waypoint x="380.0" y="257.0"></omgdi: waypoint>
</bpmndi: BPMNEdge>
<bpmndi: BPMNEdge bpmnElement="flow3" id="BPMNEdge_flow3">
  <omgdi: waypoint x="485.0" y="257.0"></omgdi: waypoint>
  <omgdi: waypoint x="561.0" y="257.0"></omgdi: waypoint>
</bpmndi: BPMNEdge>
<bpmndi: BPMNEdge bpmnElement="flow4" id="BPMNEdge_flow4">
  <omgdi: waypoint x="666.0" y="257.0"></omgdi: waypoint>
  <omgdi: waypoint x="740.0" y="257.0"></omgdi: waypoint>
</bpmndi: BPMNEdge>
</bpmndi: BPMNPlane>
</bpmndi: BPMNDiagram>
</definitions>

```

7.



Revision #1

Created 30 July 2020 08:53:48 by

Updated 30 July 2020 11:14:20 by