intra-mart Accel Platform

IM-Workflow 编程指南

2013/07/01 第 4 版

<<　变更历史　>>

变更年月日	变更内容
2012/10/01	第 1 版
2012/12/21	第 2 版 ● 2.2 添加并修正了“请求参数”的 “imwSerialProcParams”相关说明。 ● 7.4.1 “实现例”中添加了有关回调函数可接受的信息的说明。 ● 7.4.3 添加了“特别记载事项”、“7.4.3.1 IM-Workflow 8.0.2 版的改进”。 ● 变更了“7.5”章的标题。另外对“7.5”章以下的章节构成进行了调整，添加了说明。 ➢ 变更前： 从处理画面接收到的请求参数 ➢ 变更后： 处理完成后的画面迁移 ● 7.6 添加了“用户内容与连续处理/连续确认的联动方法 ”。
2013/04/01	第 3 版 ● 1.2 修正了“前提条件”。 ● 2.2 在“请求参数”中添加了智能手机用画面的说明。 ● 3 在“画面的生成”中添加了智能手机用画面的说明。 ● 3.5 在“限制事项”添加了章的说明。 ● 6.2.1 在“画面”中添加了智能手机用画面的说明。 ● 7.1 在“调出画面的初始显示值的指定”中添加了智能手机用画面的说明。 ● 7.3 在“画面输入信息的保持”中添加了智能手机用画面的说明。 ● 7.4 在“指定从调出画面调用的回调函数”中添加了智能手机用画面的说明。 ● 7.5 在“处理完成后的画面迁移”中添加了智能手机用画面的说明。 ● 7.6 在“用户内容与连续处理/连续确认的联动方法”中添加了智能手机用画面的说明。 ● 7.7 添加了“也可将PC版用户内容用于智能手机用画面 ”。 ● 除上述以外，修正了错字、缺字等错误。
2013/07/01	第 4 版 ● 修正了“ 7.1.2 实现例 ”的实现示例记载。 ● 添加了“7.4.2 非同步执行标准画面时的注意点”。

<< 目录 >>

1 序言.....1
1.1 目的.....1
1.2 前提条件.....1
1.3 准备.....1
2 概要.....2
2.1 用户应用程序数据和IM-Workflow的关系.....2
2.1.1 系统案件ID.....2
2.1.2 用户数据ID.....3
2.1.3 案件属性.....3
2.2 请求参数.....4
3 画面的生成.....7
3.1 申请画面的调用.....8
3.1.1 脚本开发模式.....9
3.1.2 JavaEE开发模式.....11
3.2 临时保存画面的调用.....13
3.2.1 脚本开发模式.....14
3.2.2 JavaEE开发模式.....16
3.3 申请（申请案件）/再申请/处理画面的调用.....18
3.3.1 脚本开发模式.....19
3.3.2 JavaEE开发模式.....21
3.4 确认画面的调用.....23
3.4.1 脚本开发模式.....24
3.4.2 JavaEE开发模式.....26
3.5 限制事项.....28
3.5.1 关于带有imw前缀的参数.....28
4 生成用户程序.....29
4.1 案件开始处理.....29
4.2 案件结束处理.....29
4.3 动作处理.....30
4.4 到达处理.....30
4.5 分支开始处理.....31
4.6 分支结束处理.....31
5 生成其他程序.....32
5.1 未完成案件删除处理监听器.....32
5.2 已完成案件删除处理监听器.....33
5.3 过去案件删除处理监听器.....34
5.4 案件存档处理监听器.....35
6 附录.....36
6.1 模板.....36
6.2 示例程序.....37
6.2.1 画面.....38
6.2.2 用户程序.....47
6.2.3 监听器.....49
7 定制.....51
7.1 调出画面的初始显示值的指定.....51
7.1.1 可指定的参数.....51

7.1.2 实现例 52
7.2 生成处理对象者插件 55
7.2.1 对象节点 55
7.2.2 示例的说明 56
7.2.3 示例的执行准备 57
7.2.4 执行示例程序 59
7.2.5 关于处理对象者插件 62
7.3 画面输入信息的保持 67
7.4 指定从调出画面调用的回调函数 69
7.4.1 实现例 69
7.4.2 非同步执行标准画面时的注意点 71
7.4.3 特别记载事项 71
7.5 处理完成后的画面迁移 72
7.5.1 用于指定要迁移到的画面的参数 72
7.5.2 要迁移到的画面可接受的请求参数 73
7.5.3 特别记载事项 73
7.6 用户内容与连续处理/连续确认的联动方法 74
7.6.1 继续执行连续处理/连续确认 74
7.6.2 中断连续处理/连续确认 74
7.7 也可将PC版用户内容用于智能手机用画面 76
7.7.1 必要工作 76

1 序言

1.1 目的

本书将对在 IM-Workflow 中可使用的画面和模块生成的方法进行说明。

本书记载了 IM-Workflow 功能的使用方法。

本书中使用的示例程序主要着眼于帮助读者理解 IM-Workflow 的功能和 API 等的使用方法。因此，有些地方可能并不是最好的编码方法。请把这些代码仅仅当成示例。

1.2 前提条件

- 本书中记述的示例程序是基于 JavaEE 开发模式和脚本开发模式编写的。因此必须对 JavaEE 开发模式和脚本开发模式有所了解。关于各开发模式，请参照附属的各种手册和 API 列表。
- 要理解本书，需要对基本的 IM-Workflow 有所了解。请参照附属的各种手册、API 列表及限制事项。
- 本书中记载的示例程序的路径位于下述目录下。
  〈 （展开的 war）/WEB-INF/ 〉

1.3 准备

为执行 IM-Workflow 的示例程序做准备。
参考“intra-mart Accel Platform / 安装指南”，并构筑 IM-Workflow 的动作环境。

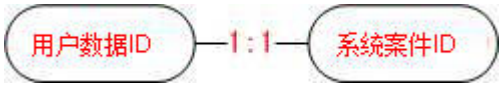
完成产品安装后，请一定要用系统管理员的身份登录，从【Tenent 环境安装】菜单进行 Tenent 环境安装，并执行示例数据安装。

对本书中记载的 JavaEE 开发模式的[java 文件]的配置场所进行说明。
实际配置的文件是[class 文件]。
JavaEE 开发模式[java 文件]的示例程序保存在产品媒介中。
另外，也可从产品最新信息下载页面（http://www.intra-mart.jp/download/product/index.html）取得。

2 概要

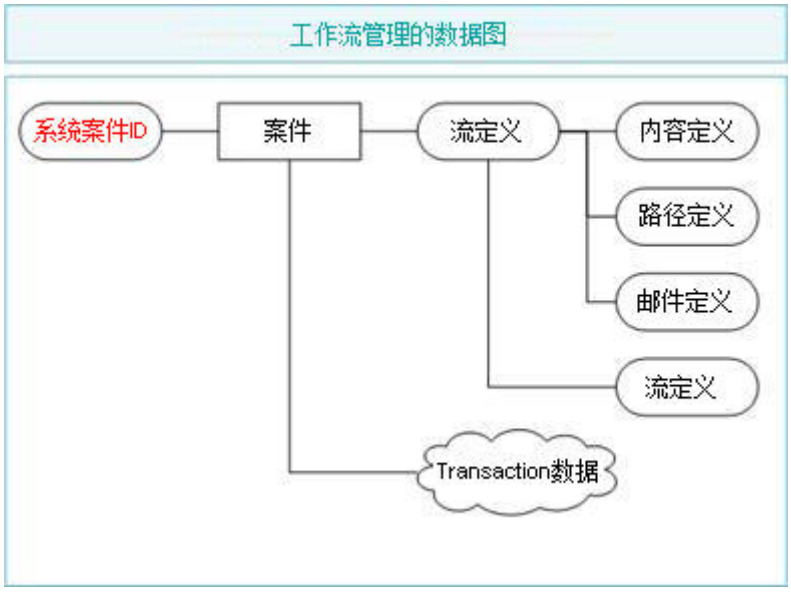
2.1 用户应用程序数据和IM-Workflow的关系

用户应用程序数据和 IM-Workflow 的数据由各自的“用户数据 ID”和“系统案件”这两个键指定。两个键以 1 对 1 的关系互相关联。



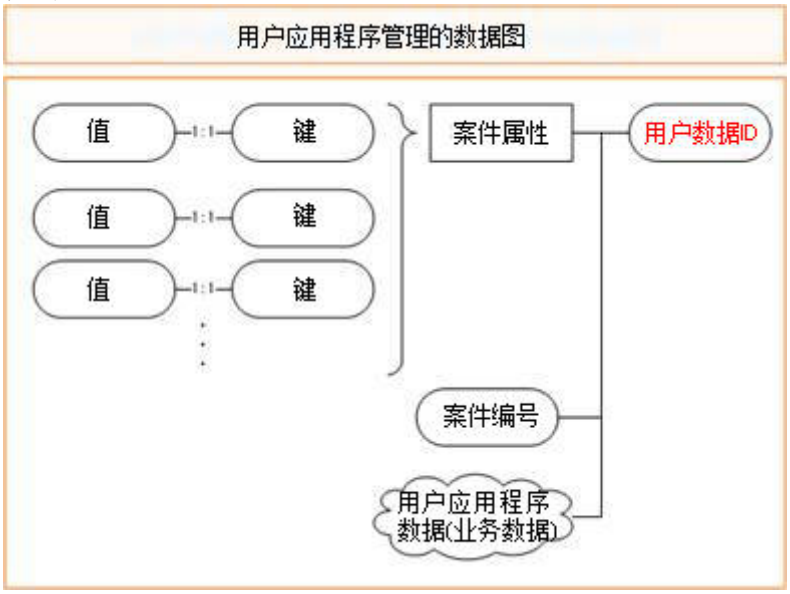
2.1.1 系统案件ID

系统案件 ID 是 IM-Workflow 中唯一的键。
是在 IM-Workflow 模块中通过编号生成确定的，不可从外部指定。
系统案件 ID 是为了在 IM-Workflow 的 API 和标签库中标识案件而使用的，不会显示在画面上。



2.1.2 用户数据ID

用户数据 ID 是用户应用程序侧唯一的键，是在用户应用程序中通过编号生成确定的键。
提出申请时，被作为参数传递给 IM-Workflow 提供的 API 及标签库。
用户数据 ID 与系统案件 ID 相同，是为了在 IM-Workflow 的 API 和标签库中标识案件而使用的，不会显示在画面上。



2.1.3 案件属性

案件属性就是通常说的 Key-Value Store。由一对“Key（键）”和“Value（值）”构成的数据模型，作为案件单位保存在 IM-Workflow 中。

通过 IM-Workflow 提供的 API 可在任意时刻登记、更新、删除和取得案件属性。
另外，可显示于 IM-Workflow 提供的各种一览画面中，可作为在分支条件的规则定义中参照的值。

2.2 请求参数

在从各种一览画面调出的申请和处理等画面中，可接受必要信息作为请求参数。

No	参数（物理名）	参数（逻辑名）	详细信息
1	imwGroupId	组别 ID	
2	imwUserCode	处理者 CD	
3	imwPageType	画面种别	所显示画面的种别 ・申请画面 ・临时保存画面 ・申请（申请案件）画面 ・再申请画面 ・处理画面 ・确认画面 ・处理详细信息 ・参照详细信息 ・确认详细信息 ・过去案件详细信息 ・申请画面（智能手机用） ・临时保存画面（智能手机用） ・申请（申请案件）画面（智能手机用） ・再申请画面（智能手机用） ・处理画面（智能手机用） ・确认画面（智能手机用）
4	imwUserDataId	用户数据 ID	
5	imwSystemMatterId	系统案件 ID	
6	imwNodeId	节点 ID	
7	imwArriveType	到达种别	
8	imwAuthUserCode	权限者 CD	登录用户在处理案件时可选择的权限者 CD。具体就是，登录用户本人和登录用户被设定为代理人时，権限者 CD 就是被代理用户 CD。 存在多个权限者时，用数组来传递该参数。※※ 但是，即便存在多个权限者，申请/临时保存画面显示时，由于一览中的权限者已确定，因此只传递已特定的权限者 CD。
9	imwApplyBaseDate	申请基准日	以“yyy/mm/dd”形式
10	imwContentsId	内容 ID	
11	imwContentsVersionId	内容版本 ID	
12	imwRouteId	路径 ID	

13	imwRouteVersionId	路径版本 ID	
14	imwFlowId	流程 ID	
15	imwFlowVersionId	流程版本 ID	
16	imwSerialProcParams	连续处理参数	连续处理用的参数 从 IM-Workflow 8.0.2 版开始，该参数变为无效。 传递的一定是空字符(""),因此不需要在用户内容间传递该参数。 连续处理用的信息包含于“imwCallOriginalParams”中。
17	imwCallOriginalParams	调用方参数	调用方页面的参数
18	imwCallOriginalPagePath	调用方页面路径	调用方的页面路径

※以下记载了在各开发模式中取得 imwAuthUserCode（权限者 CD）的例子。

此处记载的内容对于下述观点是通用的。

■ 客户端类型

脚本开发模式

```
function init(request) {
    var imwAuthUserCodeList = request.getParameterValues("imwAuthUserCode"); //权限者 CD 的数组
}
```

javaEE 开发模式

```
    HttpServletRequest request = getRequest();
    String[] imwAuthUserCodeList = request.getParameterValues("imwAuthUserCode"); //权限者 CD 的数组
```

No	参数	申请※	临时保存※	申请※	再申请※	处理※	确认※	处理详细	参照详细	确认详细	过去案件详细
1	imwGroupId	○	○	○	○	○	○	○	○	○	○
2	imwUserCode	○	○	○	○	○	○	○	○	○	○
3	imwPageType	○	○	○	○	○	○	○	○	○	○
4	imwUserDataId	-	○	○	○	○	○	○	○	○	○
5	imwSystemMatterId	-	-	○	○	○	○	○	○	○	○
6	imwNodeId	○	○	○	○	○	○	-	-	-	-
7	imwArriveType	○	○	○	○	○	-	-	-	-	-
8	imwAuthUserCode	○	○	○	○	○	-	-	-	-	-
9	imwApplyBaseDate	○	○	○	○	○	○	○	○	○	○
10	imwFlowId	○	○	○	○	○	○	○	○	○	○
11	imwFlowVersionId	○	○	○	○	○	○	○	○	○	○
12	imwContentsId	○	○	○	○	○	○	○	○	○	○
13	imwContentsVersionId	○	○	○	○	○	○	○	○	○	○
14	imwRouteId	○	○	○	○	○	○	○	○	○	○
15	imwRouteVersionId	○	○	○	○	○	○	○	○	○	○
16	imwSerialProcParams	-	-	~~○~~ -	~~○~~ -	~~○~~ -	~~○~~ -	-	-	-	-
17	imwCallOriginalParams	○	○	○	○	○	○	-	-	-	-
18	imwCallOriginalPagePath	○	○	○	○	○	○	-	-	-	-

＜「○」： 可取得 / -: 不可取得＞

※ 使用智能手机时也相同。
但是，只有“imwAuthUserCode”在智能手机用的申请、再申请、处理画面中无法取得。

3 画面的生成

本章中，将从“画面种别”、“开发模式”、“客户端类型”的观点出发，对如何实现画面以便与IM-Workflow提供的案件的各处理画面联动的基本部分进行说明。

上述观点的内容如下。

- 画面种别
  - 申请画面
  - 临时保存画面
  - 申请（申请案件）画面
  - 再申请画面
  - 处理画面
  - 确认画面
  - 处理详细
  - 参照详细
  - 确认详细
  - 过去案件详细
- 开发模式
  - 脚本开发模式
  - javaEE开发模式
- 客户端类型
  - PC
  - 智能手机

另外，在“ 7 定制”中对生成画面的应用实现进行了说明。
必要时请参照。

3.1 申请画面的调用

对与 IM-Workflow 中提供的用于申请的画面（以下，简称为申请画面）的联动方法进行说明。

使用 IM-Workflow 提供的标签库和 Client-side JavaScript API 来显示申请画面。

3.1.1 脚本开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.1.1.1 PC用画面

在与申请画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="workflowOpenPageCsjs" />

</imart>
```

在与申请画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常申请一览画面中取得的请求参数。
来自申请一览画面的请求参数中不包含“imwUserDataId”。
需要在 Function Container 中编号生成。

```
<imart type="workflowOpenPage"
        name="applyForm"
        id="applyForm"
        method="POST"
        target="_top "
        imwUserDataId=oRequest.imwUserDataId
        imwAuthUserCode=oRequest.imwAuthUserCode
        imwApplyBaseDate=oRequest.imwApplyBaseDate
        imwNodeId=oRequest.imwNodeId
        imwFlowId=oRequest.imwFlowId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示申请画面。

```
<script type="text/javascript">

workflowOpenPage('0');

</script>
```

3.1.1.2 智能手机用画面

在与申请画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="spWorkflowOpenPageCsjs" />

</imart>
```

在与申请画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常申请一览画面中取得的请求参数。
来自申请一览画面的请求参数中不包含“imwUserDataId”。
需要在 Function Container 中编号生成。

```
<imart type="spWorkflowOpenPage"
        name="applyForm"
        id="workflowOpenPageForm"
        method="POST"
        target="_top"
        imwUserDataId=$data.imwUserDataId
        imwAuthUserCode=$data.imwAuthUserCode
        imwApplyBaseDate=$data.imwApplyBaseDate
        imwNodeId=$data.imwNodeId
        imwFlowId=$data.imwFlowId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示申请画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('10');

</script>
```

3.1.2 JavaEE开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.1.2.1 PC用画面

在与申请画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:workflowOpenPageCsjs />

</imui:head>
```

在与申请画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常申请一览画面中取得的请求参数。
来自申请一览画面的请求参数中不包含“imwUserDataId”。
需要在 ServiceController 等中编号生成。

```
<workflow:workflowOpenPage
            name="applyForm"
            id="applyForm"
            method="POST"
            target="_top "
            imwUserDataId='<%=(String)request.getAttribute("imwUserDataId")%>'
            imwAuthUserCode='<%=(String)request.getAttribute("imwAuthUserCode")%>'
            imwApplyBaseDate='<%=(String)request.getAttribute("imwApplyBaseDate")%>'
            imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'
            imwFlowId='<%=(String)request.getAttribute("imwFlowId")%>'>
</workflow:workflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示申请画面。

```
<script type="text/javascript">

workflowOpenPage('0');

</script>
```

3.1.2.2 智能手机用画面

在与申请画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:spWorkflowOpenPageCsjs />

</imui:head>
```

在与申请画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常申请一览画面中取得的请求参数。
来自申请一览画面的请求参数中不包含“imwUserDataId”。
需要在 ServiceController 等中编号生成。

```
<workflow:spWorkflowOpenPage
          name="applyForm"
          id="applyForm"
          method="POST"
          target="_top "
          imwUserDataId='<%=(String)request.getAttribute("imwUserDataId")%>'
          imwAuthUserCode='<%=(String)request.getAttribute("imwAuthUserCode")%>'
          imwApplyBaseDate='<%=(String)request.getAttribute("imwApplyBaseDate")%>'
          imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'
          imwFlowId='<%=(String)request.getAttribute("imwFlowId")%>'>
</workflow:spWorkflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示申请画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('10');

</script>
```

3.2 临时保存画面的调用

对与 IM-Workflow 中提供的用于进行临时保存的画面（以下，简称为临时保存画面）的联动方法进行说明。

使用 IM-Workflow 提供的标签库和 Client-side JavaScript API 来显示临时保存画面。

3.2.1 脚本开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.2.1.1 PC用画面

在与临时保存画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="workflowOpenPageCsjs" />

</imart>
```

在与临时保存画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常申请一览画面中取得的请求参数。

```
<imart type="workflowOpenPage"
        name="tempForm"
        id="tempForm"
        method="POST"
        target="_top "
        imwUserDataId=oRequest.imwUserDataId
        imwAuthUserCode=oRequest.imwAuthUserCode
        imwApplyBaseDate=oRequest.imwApplyBaseDate
        imwNodeId=oRequest.imwNodeId
        imwFlowId=oRequest.imwFlowId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示临时保存画面。

```
<script type="text/javascript">

workflowOpenPage('1');

</script>
```

3.2.1.2 智能手机用画面

在与临时保存画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="spWorkflowOpenPageCsjs" />

</imart>
```

在与临时保存画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常申请一览画面中取得的请求参数。

```
<imart type="spWorkflowOpenPage"
       name="tempForm"
       id="tempForm"
       method="POST"
       target="_top "
       imwUserDataId=$data.imwUserDataId
       imwAuthUserCode=$data.imwAuthUserCode
       imwApplyBaseDate=$data.imwApplyBaseDate
       imwNodeId=$data.imwNodeId
       imwFlowId=$data.imwFlowId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示临时保存画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('11');

</script>
```

3.2.2 JavaEE开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.2.2.1 PC用画面

在与临时保存画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:workflowOpenPageCsjs />

</imui:head>
```

在与临时保存画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常申请一览画面中取得的请求参数。

```
<workflow:workflowOpenPage
            name="tempForm"
            id="tempForm"
            method="POST"
            target="_top "
            imwUserDataId='<%=(String)request.getAttribute("imwUserDataId")%>'
            imwAuthUserCode='<%=(String)request.getAttribute("imwAuthUserCode")%>'
            imwApplyBaseDate='<%=(String)request.getAttribute("imwApplyBaseDate")%>'
            imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'
            imwFlowId='<%=(String)request.getAttribute("imwFlowId")%>'>
</workflow:workflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示临时保存画面。

```
<script type="text/javascript">

workflowOpenPage('1');

</script>
```

3.2.2.2 智能手机用画面

在与临时保存画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:spWorkflowOpenPageCsjs />

</imui:head>
```

在与临时保存画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常申请一览画面中取得的请求参数。

```
<workflow:spWorkflowOpenPage
                name="tempForm"
                id="tempForm"
                method="POST"
                target="_top "
                imwUserDataId='<%=(String)request.getAttribute("imwUserDataId")%>'
                imwAuthUserCode='<%=(String)request.getAttribute("imwAuthUserCode")%>'
                imwApplyBaseDate='<%=(String)request.getAttribute("imwApplyBaseDate")%>'
                imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'
                imwFlowId='<%=(String)request.getAttribute("imwFlowId")%>'>
</ workflow:spWorkflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示临时保存画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('11');

</script>
```

3.3 申请（申请案件）/再申请/处理画面的调用

对与 IM-Workflow 中提供的用于进行申请（申请案件）/再申请/处理画面的画面（以下，简称为处理画面）的联动方法进行说明。

使用 IM-Workflow 提供的标签库和 Client-side JavaScript API 来显示处理画面。

3.3.1 脚本开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.3.1.1 PC用画面

在与处理画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="workflowOpenPageCsjs" />

</imart>
```

在与处理画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常未申请一览画面中取得的请求参数。

```
<imart type="workflowOpenPage"
        name="approveForm"
        id="approveForm"
        method="POST"
        target="_top "
        imwSystemMatterId=$data.imwSystemMatterId
        imwNodeId=$data.imwNodeId >
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示处理画面。

■ 申请（申请案件）

```
<script type="text/javascript">

workflowOpenPage(‘2’);

</script>
```

■ 再申请

```
<script type="text/javascript">

workflowOpenPage('3');

</script>
```

■ 处理

```
<script type="text/javascript">

workflowOpenPage('4');

</script>
```

3.3.1.2 智能手机用画面

在与处理画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="spWorkflowOpenPageCsjs" />

</imart>
```

在与处理画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常未申请一览画面中取得的请求参数。

```
<imart type="spWorkflowOpenPage"
        name="approveForm"
        id="approveForm"
        method="POST"
        target="_top "
        imwSystemMatterId=$data.imwSystemMatterId
        imwNodeId=$data.imwNodeId >
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示处理画面。

■ 申请（申请案件）

```
<script type="text/javascript">

workflowOpenPage4Sp('12');

</script>
```

■ 再申请

```
<script type="text/javascript">

workflowOpenPage4Sp('13');

</script>
```

■ 处理

```
<script type="text/javascript">

workflowOpenPage4Sp('14');

</script>
```

3.3.2 JavaEE开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.3.2.1 PC用画面

在与处理画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:workflowOpenPageCsjs />

</imui:head>
```

在与处理画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常未处理一览画面中取得的请求参数。

```
<workflow:workflowOpenPage
          name="approveForm"
          id="approveForm"
          method="POST"
          target="_top "
          imwSystemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>'
          imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'>
</workflow:workflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示处理画面。

■ 申请（申请案件）

```
<script type="text/javascript">

workflowOpenPage('2');

</script>
```

■ 再申请

```
<script type="text/javascript">

workflowOpenPage('3');

</script>
```

■ 处理

```
<script type="text/javascript">

workflowOpenPage('4');

</script>
```

3.3.2.2 智能手机用画面

在与处理画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:spWorkflowOpenPageCsjs />

</imui:head>
```

在与处理画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常未处理一览画面中取得的请求参数。

```
<workflow:spWorkflowOpenPage
          name="approveForm"
          id="approveForm"
          method="POST"
          target="_top "
          imwSystemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>'
          imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'>
</workflow:spWorkflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示处理画面。

■ 申请（申请案件）

```
<script type="text/javascript">

workflowOpenPage4Sp('12');

</script>
```

■ 再申请

```
<script type="text/javascript">

workflowOpenPage4Sp('13');

</script>
```

■ 处理

```
<script type="text/javascript">

workflowOpenPage4Sp('14');

</script>
```

3.4 确认画面的调用

对与 IM-Workflow 中提供的用于进行确认的画面（以下，简称为确认画面）的联动方法进行说明。

使用 IM-Workflow 提供的标签库和 Client-side JavaScript API 来显示确认画面。

3.4.1 脚本开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.4.1.1 PC用画面

在与确认画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="workflowOpenPageCsjs" />

</imart>
```

在与确认画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常确认一览画面中取得的请求参数。

```
<imart type="workflowOpenPage"
       name="confirmForm"
       id="confirmForm"
       method="POST"
       target="_top "
       imwSystemMatterId=$data.imwSystemMatterId
       imwNodeId=$data.imwNodeId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示确认画面。

```
<script type="text/javascript">

workflowOpenPage('5');

</script>
```

3.4.1.2 智能手机用画面

在与确认画面联动的画面的 header 部（<imart type=“head”> ～ </imart>）记述下述的 IMART 标签。

```
<imart type=“head”>

<imart type="spWorkflowOpenPageCsjs" />

</imart>
```

在与确认画面联动的画面的 body 部记述下面的 IMART 标签。

IMART 标签中指定的属性指定从通常确认一览画面中取得的请求参数。

```
<imart type="spWorkflowOpenPage"
       name="confirmForm"
       id="confirmForm"
       method="POST"
       target="_top "
       imwSystemMatterId=$data.imwSystemMatterId
       imwNodeId=$data.imwNodeId>
</imart>
```

通过执行下述 Client-side JavaScript API，即可显示确认画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('15');

</script>
```

3.4.2 JavaEE开发模式

关于 IM-Workflow 用的标签库的使用方法请一并参照 API 列表。

3.4.2.1 PC用画面

在与确认画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:workflowOpenPageCsjs />

</imui:head>
```

在与确认画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常确认一览画面中取得的请求参数。

```
<workflow:workflowOpenPage
          name="confirmForm"
          id="confirmForm"
          method="POST"
          target="_top "
          imwSystemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>'
          imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'>
</workflow:workflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示确认画面。

```
<script type="text/javascript">

workflowOpenPage('5');

</script>
```

3.4.2.2 智能手机用画面

在与确认画面联动的画面的 header 部（<imui:head> ～ </imui:head>）记述下述的标签库。

```
<imui:head>

<workflow:spWorkflowOpenPageCsjs />

</imui:head>
```

在与确认画面联动的画面的 body 部记述下面的标签库。

标签库中指定的属性指定从通常确认一览画面中取得的请求参数。

```
<workflow: spWorkflowOpenPage
          name="confirmForm"
          id="confirmForm"
          method="POST"
          target="_top "
          imwSystemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>'
          imwNodeId='<%=(String)request.getAttribute("imwNodeId")%>'>
</workflow:spWorkflowOpenPage>
```

通过执行下述 Client-side JavaScript API，即可显示确认画面。

```
<script type="text/javascript">

workflowOpenPage4Sp('15');

</script>
```

3.5 限制事项

对生成画面时的限制事项进行说明。
有关未在此处记载的限制事项，请参考《intra-mart Accel Platform /发行说明》。

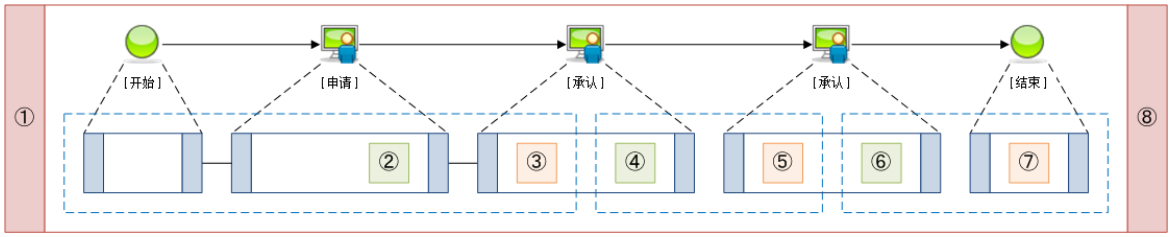
3.5.1 关于带有imw前缀的参数

“workflowOpenPage”标签和“spWorkflowOpenPage”中，会输出多个用于工作流处理时控制用的带有 imw 前缀的 hidden 标签。

由于参数名称重复有可能导致无法正常处理，除了下述已明确记载的可被使用的内容之外，请不要记述带有 imw 前缀的参数。

- 7.1 调出画面的初始显示值的指定
- 7.3 画面输入信息的保持

4 生成用户程序



No	处理名	项目号
1	案件开始处理	①
2	案件结束处理	⑧
3	动作处理	② ④ ⑥
4	到达处理	③ ⑤ ⑦

4.1 案件开始处理

案件开始处理指的是在案件开始时执行的一次处理。
在下述情况下执行。

- 申请者进行了申请时
- 生成了“申请”的案件时（仅限 API）

由于案件开始处理是在 IM-Workflow 模块的事务处理内执行的，因此无法在此程序中进行 DB 事务控制。

4.2 案件结束处理

案件结束处理指的是在案件开始时执行的一次处理。
在下述情况下执行。

- 最后的审批者进行了审批时
- 审批者进行了“审批结束”时
- 审批者进行了“否认”时
- 申请者进行了“中止”时
- 案件操作中到达结束节点时

案件结束处理与之前的动作处理和到达处理是独立的处理（事务）。因此，在案件结束处理中发生错误时，无法返回（回滚）到之前的处理。

由于案件结束处理是在 IM-Workflow 模块的事务内执行的，因此无法在此程序中进行 DB 事务控制。

4.3 动作处理

动作处理指的是在进行了下述行为时执行的处理。

No	动作	方法
1	申请	apply
2	再申请	reapply
3	申请（临时保存）	applyFromTempSave
4	申请（未处理）	applyFromUnapply
5	中止	discontinue
6	撤回	pullBack
7	退回后撤回	sendBackToPullBack
8	审批	approve
9	审批结束	approveEnd
10	否认	deny
11	退回	sendBack
12	搁置	reserve
13	搁置解除	reserveCancel
14	案件操作	matterHandle
15	临时保存(新建登记)	tempSaveCreate
16	临时保存(更新)	tempSaveUpdate
17	临时保存(删除)	tempSaveDelete

由于动作处理是在 IM-Workflow 模块的事务内执行的，因此无法在此程序中进行 DB 事务控制。

4.4 到达处理

到达处理指的是到达节点时执行的处理。
此处理被作为与动作处理和 IM-Workflow 的内部处理相独立的处理（thread）来执行。
因此，在到达处理中发生错误时，无法返回（回滚）到之前的处理。
（与之前的动作处理的事务也不相同。）

此程序中，在进行数据库的登记/更新/删除处理时，请独自进行 DB 事务控制。

在下述情况下执行。

- 前一个节点的处理者进行了“申请”或“审批”后到达本节点时
- 被“退回”，从其他节点到达本节点时
- 进行“撤回”并到达本节点时
- 通过案件操作到达本节点时

4.5 分支开始处理

分支开始处理指的是在分支开始节点选择了“在用户程序中分支”时执行的处理。
按照顺序执行每个分支地节点。

由于分支开始处理是在 IM-Workflow 模块的事务内执行的，因此无法在此程序中进行 DB 事务控制。

分支开始处理中，若返回的路径迁移可否的条件是迁移（true），则前往执行中设定了分支开始处理的分支地节点。
若全部分支开始处理的路径迁移可否条件是不迁移（false）时，案件会在分支开始节点处停止。
这种情况下，请通过案件操作处理来继续推进案件。

4.6 分支结束处理

分支结束处理指的是在分支结束节点选择了“在用户程序中分支”时执行的处理。
在案件到达分支结束节点时执行。

由于分支结束处理是在 IM-Workflow 模块的事务内执行的，因此无法在此程序中进行 DB 事务控制。

分支结束处理中，若返回的路径迁移可否的条件是结合（true），则不等待未到达的节点前往下一个分支节点。
即便全部节点都已到达，但结果全部是不结合（false）时，案件也会在分支结束节点停止。
这种情况下，请通过案件操作处理来继续推进案件。

5 生成其他程序

5.1 未完成案件删除处理监听器

未完成案件删除处理监听器指的是删除了未完成案件时执行的程序。
通常，通过“案件操作”画面进行了“案件删除”时，或在执行了删除未完成案件的 API 时被调用。

通常在“**内容定义**”中设定未完成案件删除处理监听器。

另外，以 Tenant 为单位进行处理时，需设定下述文件。

%PUBLIC_STORAGE%/im_workflow/conf/param/param_group_%Tenant ID%.xml

```
<!--
    未完成案件删除监听器的种类
        [java] 或 [script] 或 []（无指定）
        []设定了（无指定）时监听器不启动
-->
<param>
    <param-name>delete-active-matter-type</param-name>
    <param-value></param-value>
</param>
<!--
    未完成案件删除监听器的路径
    1.案件删除监听器的种类为 java：package 名
    2.案件删除监听器的种类为 script：WEB-INF/jssp 的路径
-->
<param>
    <param-name>delete-active-matter-listener-path</param-name>
    <param-value></param-value>
</param>
```

※也可从工作流参数画面进行设定。

有关设定方法的详细信息，请参照“IM-Workflow 管理员操作指南”或“IM-Workflow 式样书”。

5.2 已完成案件删除处理监听器

已完成案件删除处理监听器指的是删除了已完成案件时执行的程序。
通常，通过“参照”画面的已完成案件标签页进行了案件的“删除”时，或在执行了删除已完成案件的 API 时被调用。

通常在“**内容定义**”中设定已完成案件删除处理监听器。

另外，以 Tenant 为单位进行处理时，需设定下述文件。

%PUBLIC_STORAGE%/im_workflow/conf/param/param_group_%Tenant ID%.xml

```
<!--
    已完成案件删除监听器的种类
        [java] 或 [script] 或 []（无指定）
        []设定了（无指定）时监听器不启动
-->
<param>
    <param-name>delete-complete-matter-listener-type</param-name>
    <param-value></param-value>
</param>
<!--
    已完成案件删除监听器的路径
    1.案件删除监听器的种类为 java：package 名
    2.案件删除监听器的种类为 script：WEB-INF/jssp 的路径
-->
<param>
    <param-name>delete-complete-matter-listener-path</param-name>
    <param-value></param-value>
</param>
```

※也可从工作流参数画面进行设定。

有关设定方法的详细信息，请参照“IM-Workflow 管理员操作指南”或“IM-Workflow 式样书”。

5.3 过去案件删除处理监听器

过去案件删除处理监听器指的是删除了过去案件时执行的程序。

通常在“**内容定义**”中设定过去案件删除处理监听器。

另外，以 Tenant 为单位进行处理时，需设定下述文件。

%PUBLIC_STORAGE%/im_workflow/conf/param/param_group_%Tenant ID%.xml

```
----------------------------------------------------------------------------------------------------------------------

<!--
    过去案件删除监听器的种类
        [java] 或 [script] 或 []（无指定）
        []设定了（无指定）时监听器不启动
-->
<param>
    <param-name>delete-archive-matter-listener-type</param-name>
    <param-value></param-value>
</param>
<!--
    过去案件删除监听器的路径
    1.案件删除监听器的种类为 java：package 名
    2.案件删除监听器的种类为 script：WEB-INF/jssp 的路径
-->
<param>
    <param-name>delete-archive-matter-listener-path</param-name>
    <param-value></param-value>
</param>
```

※也可从工作流参数画面进行设定。

有关设定方法的详细信息，请参照“IM-Workflow 管理员操作指南”或“IM-Workflow 式样书”。

5.4 案件存档处理监听器

案件存档处理监听器指的是进行了案件存档处理时执行的程序。
通常在执行了 Job“IM-Workflow/存档”时被调用。

通常在“**内容定义**”中设定案件存档处理监听器。

另外，以 Tenant 为单位进行处理时，需设定下述文件。

%PUBLIC_STORAGE%/im_workflow/conf/param/param_group_%Tenant ID%.xml

```
----------------------------------------------------------------------------------------------------------------

<!--
    案件存档监听器的种类
        [java] 或 [script] 或 []（无指定）
        []设定了（无指定）时监听器不启动
-->
<param>
    <param-name>archive-proc-listener-type</param-name>
    <param-value>java</param-value>
</param>
<!--
    案件存档监听器的路径
    1.案件存档监听器的种类为 java：package 名
    2.案件存档监听器的种类为 script：WEB-INF/jssp 的路径
-->
<param>
    <param-name>archive-proc-listener-path</param-name>
    <param-value></param-value>
</param>
```

※也可从工作流参数画面进行设定。

有关设定方法的详细信息，请参照“IM-Workflow 管理员操作指南”或“IM-Workflow 式样书”。

6 附录

6.1 模板

提供了生成用户程序和各监听器程序时要用的模板。

- 脚本开发模式

</./jssp/src/sample/im_workflow/template/>

No	处理	物理名
1	案件开始处理	MatterStartProcess.js
2	案件结束处理	MatterEndProcess.js
3	动作处理	ActionProcess.js
4	到达处理	ArriveProcess.js
5	分支开始处理/分支结束处理	RuleCondition.js
6	未完成案件删除处理监听器	WorkflowActvMatterDeleteListener.js
7	已完成案件删除处理监听器	WorkflowCplMatterDeleteListener.js
8	过去案件删除处理监听器	WorkflowArcMatterDeleteListener.js
9	案件存档处理监听器	WorkflowMatterArchiveListener.js
10	处理对象者插件	WorkflowAuthorityExecEventListener.js

- JavaEE 开发模式

JavaEE 开发模式[java 文件]的示例程序保存在产品媒介中。
另外，也可从产品最新信息下载页面（http://www.intra-mart.jp/download/product/index.html）取得。

<%示例程序目录%/src/main/java/jp/co/intra_mart/sample/workflow/template/>

No	处理	物理名
1	案件开始处理	MatterStartProcess.java
2	案件结束处理	MatterEndProcess.java
3	动作处理	ActionProcess.java
4	到达处理	ArriveProcess.java
5	分支开始处理/分支结束处理	RuleCondition.java
6	未完成案件删除处理监听器	WorkflowActvMatterDeleteListener.java
7	已完成案件删除处理监听器	WorkflowCplMatterDeleteListener.java
8	过去案件删除处理监听器	WorkflowArcMatterDeleteListener.java
9	案件存档处理监听器	WorkflowMatterArchiveListener.java
10	处理对象者插件	WorkflowAuthorityExecEventListener.java
11	检索器登记文档添加监听器	WorkflowCrawlingAddListener.java

6.2 示例程序

安装 IM-Workflow 时进行“示例数据安装”，并对导入了示例数据时可使用的示例程序进行说明。

示例程序包括脚本开发模式和 JavaEE 开发模式的示例程序。
虽然开发模式不同，两种示例都是“物品采购”的申请书，且动作式样相同。

6.2.1 画面

6.2.1.1 申请/临时保存/申请(申请案件)/再申请画面

对 PC 用画面和智能手机用画面进行说明。

6.2.1.1.1 PC用画面

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/apply.html>
<./jssp/src/sample/im_workflow/purchase/screen/apply.js>
```

- JavaEE 开发模式

```
<./lib/im_workflow-8.0.x-sample.jar/service-config-imw_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------------


  <service>
    <service-id>apply</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApplyServiceController</co
ntroller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApplyServiceTransition</tra
nsition-class>
    <next-page>
      <page-path>/sample/im_workflow/purchase/apply.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.1.2 **智能手机用画面**

- 脚本开发模式

```
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/apply.html>
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/apply.js>
```

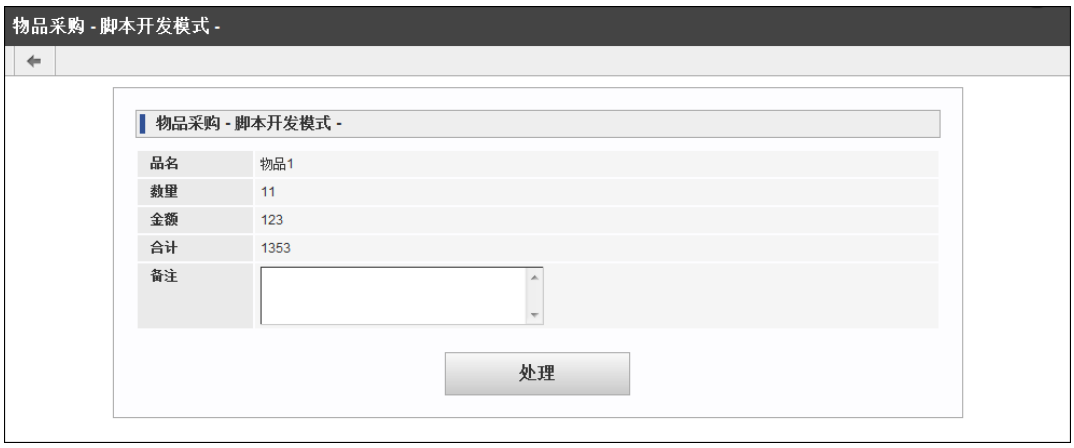
- JavaEE 开发模式

```
<./lib/im_workflow_smartphone-8.0.x-sample.jar/service-config-imw_sp_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------

  <service>
    <service-id>apply</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApplyServiceController</co
ntroller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApplyServiceTransition</tra
nsition-class>
    <next-page>
      <page-path>/sample/im_workflow_smartphone/purchase/apply.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.2 处理画面

对 PC 用画面和智能手机用画面进行说明。

6.2.1.2.1 PC用画面



■ 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/approve.html>
<./jssp/src/sample/im_workflow/purchase/screen/approve.js>
```

■ JavaEE 开发模式

```
<./lib/im_workflow-8.0.x-sample.jar/service-config-imw_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------------


  <service>
    <service-id>approve</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApproveServiceController</
controller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApproveServiceTransition</t
ransition-class>
    <next-page>
      <page-path>/sample/im_workflow/purchase/approve.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.2.2 **智能手机用画面**

- 脚本开发模式

```
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/approve.html>
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/approve.js>
```

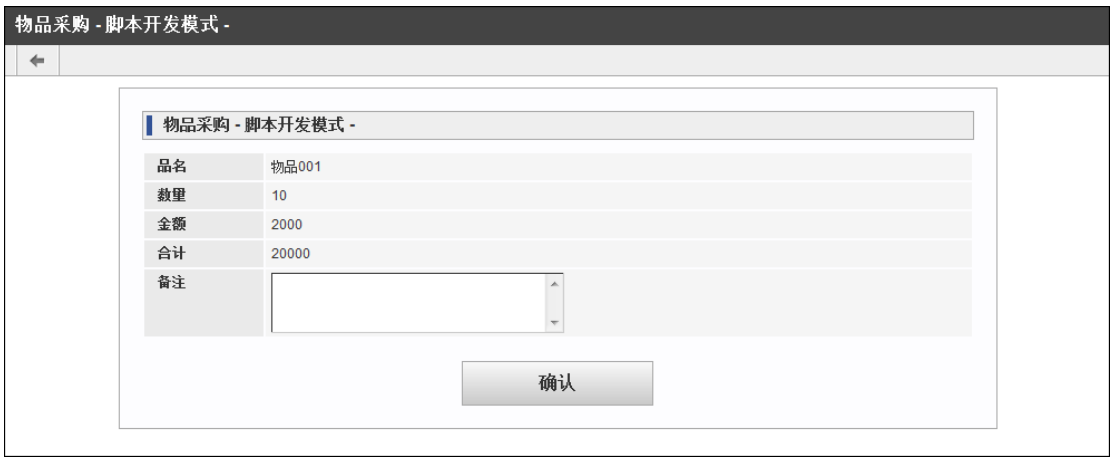
- JavaEE 开发模式

```
<./lib/im_workflow_smartphone-8.0.x-sample.jar/service-config-imw_sp_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------------

  <service>
    <service-id>approve</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApproveServiceController</
controller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ApproveServiceTransition</t
ransition-class>
    <next-page>
      <page-path>/sample/im_workflow_smartphone/purchase/approve.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.3 确认画面

对 PC 用画面和智能手机用画面进行说明。

6.2.1.3.1 PC用画面



- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/confirm.html>
<./jssp/src/sample/im_workflow/purchase/screen/confirm.js>
```

- JavaEE 开发模式

```
<./lib/im_workflow-8.0.x-sample.jar/service-config-imw_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------

  <service>
    <service-id>confirm</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ConfirmServiceController</c
ontroller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ConfirmServiceTransition</t
ransition-class>
    <next-page>
      <page-path>/sample/im_workflow/purchase/confirm.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.3.2 **智能手机用画面**

- 脚本开发模式

```
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/confirm.html>
<./jssp/src/sample/im_workflow_smartphone/purchase/screen/confirm.js>
```

- JavaEE 开发模式

```
<./lib/im_workflow_smartphone-8.0.x-sample.jar/service-config-imw_sp_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------

  <service>
    <service-id>confirm</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ConfirmServiceController</controller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.ConfirmServiceTransition</transition-class>
    <next-page>
      <page-path>/sample/im_workflow_smartphone/purchase/confirm.jsp</page-path>
    </next-page>
  </service>
```

6.2.1.4 处理详细信息/参照详细信息/过去案件详细信息/确认详细画面

■ 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/detail.html>
<./jssp/src/sample/im_workflow/purchase/screen/detail.js>
```

■ JavaEE 开发模式

```
<./lib/im_workflow-8.0.x-sample.jar/service-config-imw_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------

  <service>
    <service-id>detail</service-id>

<controller-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.DetailServiceController</co
ntroller-class>

<transition-class>jp.co.intra_mart.sample.workflow.purchase.controller.service.DetailServiceTransition</tra
nsition-class>
    <next-page>
      <page-path>/sample/im_workflow/purchase/detail.jsp</page-path>
    </next-page>
  </service>
```

在处理详细信息/参照详细信息/过去案件详细信息/确认详细画面（以下，简称详细画面）中会显示通过内容定义定义的画面。因此，在详细画面显示 IM-Workflow 信息（案件名和附件等）时，需使用 IM-Workflow 提供的标签库。

用于显示案件信息的标签库。

案件号	0000000018
案件名	物品购买2
申请人	冈山益男
申请标准日	2012/09/19

■ 脚本开发模式 detail.html

```
43 | </header>
44 | <imart type="workflowMatterData" systemMatterId=$data.imwSystemMatterId
45 |                                  displayItem="matter_number,matter_name,apply_user,apply_base_date" />
46 | <table class="imui-form">
```

■ JavaEE 开发模式 detail.jsp

```
51 | </header>
52 | <workflow:workflowMatterData systemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>'
53 |                                  displayItem="matter_number,matter_name,apply_user,apply_base_date" />
54 | <table class="imui-form">
```

用于显示案件附件的标签库。

附件			
文件名	**长度**	**登记者**	**登记时间**
报价单	1 KB	冈山益男	2012/09/19 19:49

■ 脚本开发模式 detail.html

```
69 |  </table>
70 |  <imart type="workflowMatterFile" systemMatterId=$data.imwSystemMatterId />
71 | </div >
```

■ JavaEE 开发模式 detail.jsp

```
78 |  </table>
79 |  <workflow:workflowMatterFile systemMatterId='<%=(String)request.getAttribute("imwSystemMatterId")%>' />
80 | </div >
```

◆ 若案件无附件，则不显示。

从智能手机版 IM-Workflow 迁移到这些详细画面时，会在新窗口中打开 PC 用画面。想在从智能手机的画面迁移中显示 PC 用画面时，需要明确地将客户端类型切换成 PC。

■ 脚本开发模式 detail.js

```
19 | function init ( request ) {
20 |    ClientTypeSwitcher.oneTimeSwitchTo('pc');
21 |
```

■ JavaEE 开发模式 detail.jsp

```
 7 | <%
 8 | ClientTypeSwitcher.oneTimeSwitchTo("pc");
 9 | %>
```

有关 ClientTypeSwitcher，详细信息请参照 API 列表。

另外，若不想在新窗口的画面中显示全局浏览和 My Menu，需要在下述过滤器中添加画面的路径。

<./conf /theme-head-only-path-config.xml>

■ 脚本开发模式

```
 7 |  <path>/sample/im_workflow/purchase/screen/detail</path>
```

■ JavaEE 开发模式

```
 8 |  <path>/imw_sample_purchase-detail.service</path>
```

6.2.2 用户程序

6.2.2.1 动作处理程序

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/action/ActionProcess1.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/ActionProcess1.java>
```

示例数据中，定义了“ActionProcess1”作为申请节点的动作处理。

“ActionProcess1”中进行了下述两个处理。

- 将用户应用程序的数据保存在表中。
  - 进行了申请或临时保存时，在画面上输入的信息会登记/更新到由用户应用程序定义的独自的数据表中。
- 获取案件编号生成。
  - 需要通过申请的动作处理来设定案件编号。
  - 此处，通过 IM-Workflow 提供的“WorkflowNumberingManager#getNumber()”来获取案件编号。
- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/action/ActionProcess2.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/ActionProcess2.java>
```

示例数据中，定义了“ActionProcess2”作为申请节点的动作处理。

在“ActionProcess2”中，进行把通过画面输入的“数量×金额”的合计金额”登记为案件属性的处理。

6.2.2.2 案件结束处理程序

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/action/MatterEndProcess.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/MatterEndProcess.java>
```

示例数据中定义了“MatterEndProcess”作为案件结束处理。

在“MatterEndProcess”中，对用户应用程序中定义的独自的数据表进行了更新处理。

6.2.2.3 分支开始处理程序

- 流程定义“分支路径【脚本开发模式】”中使用的分支开始处理程序

```
<./jssp/src/sample/im_workflow/purchase/action/RuleCondition1.js>
<./jssp/src/sample/im_workflow/purchase/action/RuleCondition2.js>
<./jssp/src/sample/im_workflow/purchase/action/RuleCondition3.js>
```

- 流程定义“分支路径【JavaEE 开发模式】”中使用的分支开始处理程序

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/RuleCondition1.java>
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/RuleCondition2.java>
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/action/RuleCondition3.java>
```

在“RuleCondition1”中，合计金额在未达 10000 时，作为结果标志成功返回（true）。

在“RuleCondition2”中，“合计金额”在 10000 到 50000 时，作为结果标志成功返回（true）。

在“RuleCondition3”中，“合计金额”在超过 50000 时，作为结果标志成功返回（true）。

6.2.3 监听器

6.2.3.1 未完成案件删除处理监听器

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/listener/WorkflowActvMatterDeleteListener.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/listener/WorkflowActvMatterDeleteListener.j
```

“WorkflowActvMatterDeleteListener”中进行了下述两个处理。

- 从表中删除用户应用程序的数据。
  - 在案件删除的同时删除申请时登记的用户应用程序数据。
- 删除案件属性。
  - 从案件属性中删除申请时登记到案件属性中的“合计金额”。

6.2.3.2 已完成案件删除处理监听器

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/listener/WorkflowCplMatterDeleteListener.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/listener/WorkflowCplMatterDeleteListener.ja
```

在“WorkflowCplMatterDeleteListener”中进行了下述处理。

在案件删除的同时删除申请时登记的用户应用程序数据。

◆ 案件属性的信息会在案件删除时由 IM-Workflow 模块自动删除，因此不需要进行个别删除。

6.2.3.3 过去案件删除处理监听器

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/listener/WorkflowArcMatterDeleteListener.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/listener/WorkflowArcMatterDeleteListener.ja
```

在“WorkflowArcMatterDeleteListener”中进行了下述处理。

在案件删除的同时删除申请时登记的用户应用程序数据。

◆ 案件属性的信息会在案件删除时由 IM-Workflow 模块自动删除，因此不需要进行个别删除。

6.2.3.4 案件存档处理监听器

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/listener/WorkflowMatterArchiveListener.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/
        jp/co/intra_mart/sample/workflow/purchase/listener/WorkflowMatterArchiveListener.java
```

在“WorkflowMatterArchiveListener”中进行了下述处理。

对用户应用程序中定义的独自的数据表进行了更新处理。

7 定制

7.1 调出画面的初始显示值的指定

此处记载的内容对下述观点是通用的。

- 开发模式
- 客户端类型

对由 IM-Workflow 提供的各处理（申请/再申请/申请（申请案件）/临时保存/处理/确认）画面被调出时，外部指定调出画面中的初始显示值的方法进行说明。

7.1.1 可指定的参数

通过在“workflowOpenPage”标签内部记述下述参数，可外部指定调出画面的初始显示值。

No	参数（物理名）	参数（逻辑名）	调出画面侧的 相应项目	动作对象调出画面
1	imwMatterName	案件名	案件名	申请/临时保存/申请(申请案件)/再申请
2	imwComment	备注	备注	全部
3	imwForcedParamFlag	强制参数标志	※动作控制用标志	-

另外，在下述条件下，只有将“imwForcedParamFlag”（强制参数标志）的值设定为"1"后，初始显示值指定才会生效。
若不将“imwForcedParamFlag”（强制参数标志）的值设为"1"，或不记述“imwForcedParamFlag”（强制参数标志）时，优先使用已登记的信息。

No	调出画面	条件
1	申请	从临时保存进行申请时
2	临时保存	再次保存临时保存的信息时
3	申请（申请案件）	-
4	再申请	-

7.1.2 实现例

在作为示例所提供的“物品采购”申请书中，初始显示时，把在申请画面输入的“商品名”显示为“案件名”，把“备考”显示为“备注”的例子。

此外，只准备了 PC 用画面的示例。
智能手机用画面的全体流程也相同。请注意，在实现中使用的标签库和 Client-side JavaScript API 不同。

下述程序是记述了为了进行初始显示的处理的程序。

- 脚本开发模式

<./jssp/src/sample/im_workflow/purchase/screen/apply_display.html>

- JavaEE 开发模式

<（展开的 war）/sample/im_workflow/purchase/apply_display.jsp>

将这些文件变更为下述文件名并覆盖保存，即可在申请画面对本功能进行动作确认。

- 脚本开发模式

<./jssp/src/sample/im_workflow/purchase/screen/apply.html>

- JavaEE 开发模式

<（展开的 war）/sample/im_workflow/purchase/apply.jsp>

通过记述下述处理，即可实现初始显示。

```
<imart type=“head” >
<title>
  <imart type="string" value=$msg.cap010 escapeXml="true" escapeJs="false" />
</title>

<imart type="workflowOpenPageCsjs" />
<script src="ui/libs/jquery-validation-1.9.0/jquery.validate.js"></script>
<script type="text/javascript">
.
.
.

function setParam() {
  $('#imwMatterName').val($('#item_name').val());
  $('#imwComment').val($('#item_comment').val());
  $('#imwForcedParamFlag').val('1');
}

$(function(){
.
.
.
  $('#openPage1').click(function(){
    setParam();
    workflowOpenPage('1');
  });
.
.
.
}
</script>
</imart>

<imart type="workflowOpenPage"
       name="workflowOpenPageForm"
       id="workflowOpenPageForm"
```

```
        method="POST"
        target="_top"
        imwUserDataId=$data.imwUserDataId
        imwSystemMatterId=$data.imwSystemMatterId
        imwAuthUserCode=$data.imwAuthUserCode
        imwApplyBaseDate=$data.imwApplyBaseDate
        imwNodeId=$data.imwNodeId
        imwFlowId=$data.imwFlowId
        imwCallOriginalParams=$data.imwCallOriginalParams
        imwNextScriptPath=$data.imwCallOriginalPagePath>

  <input type="hidden" name="imwMatterName" id="imwMatterName" />
  <input type="hidden" name="imwComment" id="imwComment" />
  <input type="hidden" name="imwForcedParamFlag" id="imwForcedParamFlag" />
.
.
.
</imart>
```

7.2 生成处理对象者插件

对于在 IM-Workflow 各节点指定的“处理对象者”中添加独自生成的处理对象者的方法进行说明。

IM-Workflow 的处理对象者是以插件的形式实现功能扩展的。
若要添加插件，则需按照与扩充点相应的内容实现生成插件，并记载到设定文件中以便将插件安装到对象的扩充点上。
扩充点和插件之间的关系是由 intra-mart Accel Platform 的 API“PluginManager”来管理的。

7.2.1 对象节点

处理对象者的插件是根据节点的种类来决定“extension point”的。

No	节点	extension point
1	审批(※1)	jp.co.intra_mart.workflow.plugin.authority.node.approve
2	审批(※2)	jp.co.intra_mart.workflow.plugin.authority.node.approve.static
3	动态审批	jp.co.intra_mart.workflow.plugin.authority.node.dynamic
4	确认	jp.co.intra_mart.workflow.plugin.authority.node.confirm

※1 前一个节点是“申请节点”或“审批节点”时
※2 前一个节点是“申请节点”或“审批节点”之外的节点时

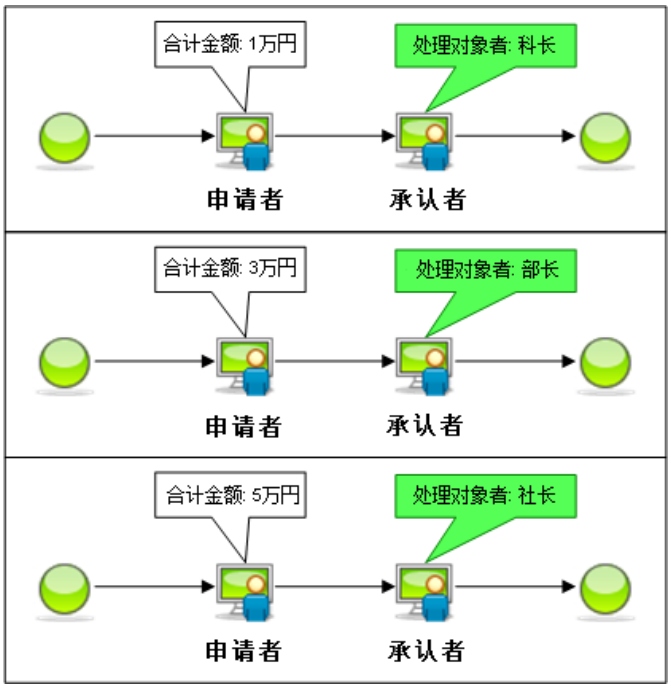
7.2.2 示例的说明

示例中提供的“处理对象者插件”是与相同示例中提供的“物品采购”画面一起联动的。

根据在“物品采购”画面中输入的“数量”和“金额”计算出“合计金额”来确定下一步的审批者。
具体就是，根据“合计金额”，如

- 1 万日元以下
  - 课长
- 1 万日元以上 5 万日元以下
  - 部长
- 5 万日元以上
  - 总经理

来指定不同职位作为处理对象者。



7.2.3 示例的执行准备

此处，将尝试使用在审批节点中使用根据“合计金额”来确定处理对象者的插件。

编辑下述文件。

<./plugin/jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve/plugin.xml>

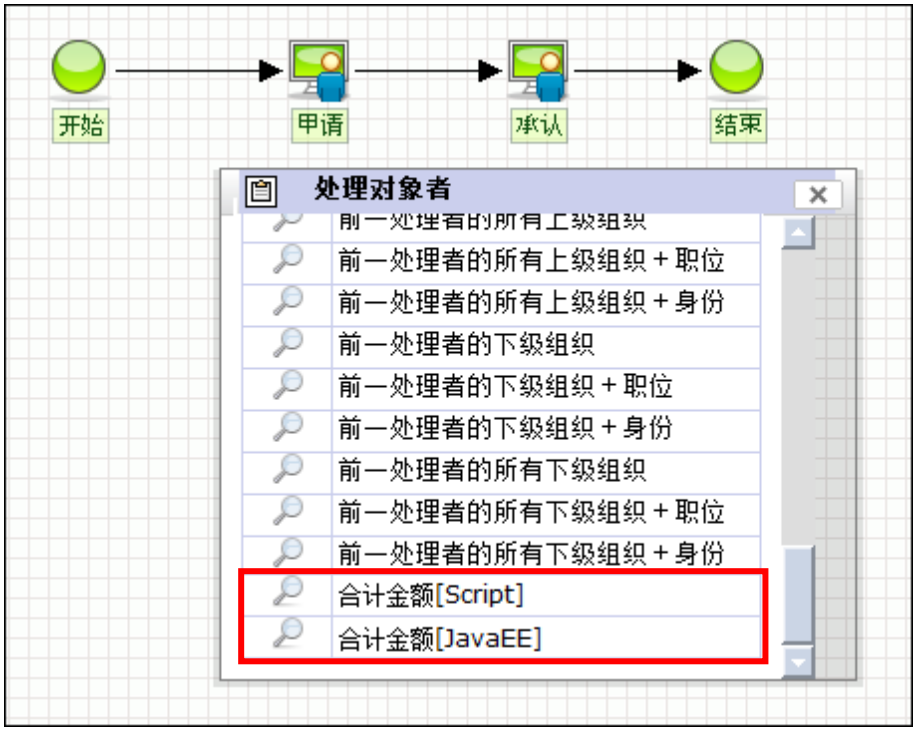
脚本开发模式

JavaEE 开发模式

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin>
    <extension point="jp.co.intra_mart.workflow.plugin.authority.node.approve" >

        <authority
            name="%jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.script"
            id="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.script"
            version="7.2.0"
            rank="910"
            enable="true">
            <configPage>
                <script pagePath="sample/im_workflow/purchase/plugin/authority/item_total/itemTotalConfig">
                    <parameter key="pluginName" value="SAMPLE.IMW.CAP.030" />
                </script>
            </configPage>
            <extend>
                <script
            file="sample/im_workflow/purchase/plugin/authority/item_total/WorkflowAuthorityExecEventListener" />
            </extend>
        </authority>

        <authority
            name="%jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.javaee"
            id="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.javaee"
            version="7.2.0"
            rank="920"
            enable="true">
            <configPage>
                <javaee applicationId="imw_sample_purchase" serviceId="authority_item_total">
                    <parameter key="pluginName" value="SAMPLE.IMW.CAP.031" />
                </javaee>
            </configPage>
            <extend>
                <java
    class="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.item_total.WorkflowAuthorityExecEventListener" />
            </extend>
        </authority>

    </extension>
</plugin>
```

编辑完上述文件后，重新启动服务器。

从“路径定义”画面生成下面的路径。
进行审批节点的处理对象者检索后，会如下所示显示“合计金额‘Script’”和“合计金额‘JavaEE’”。



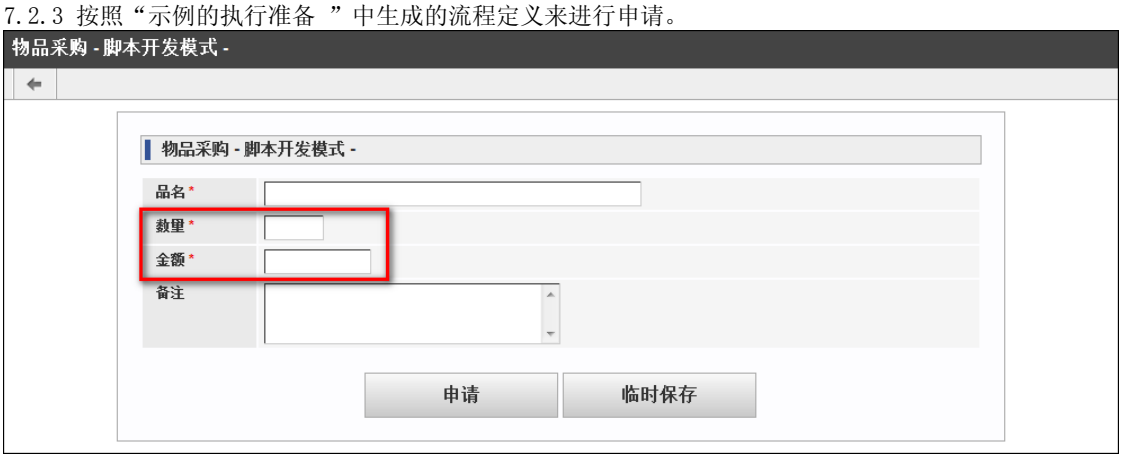
“合计金额‘Script’”和“合计金额‘JavaEE’”是由于实现方法（开发语言）不同而产生的不同显示，在处理内容上没有区别。

选择“合计金额‘Script’”或“合计金额‘JavaEE’”，生成路径。

接下来，从“流程定义”画面生成使用了上述生成的路径定义的流程定义。
此时，请选择示例中提供的“脚本开发模式”或“JavaEE 开发模式”作为内容。

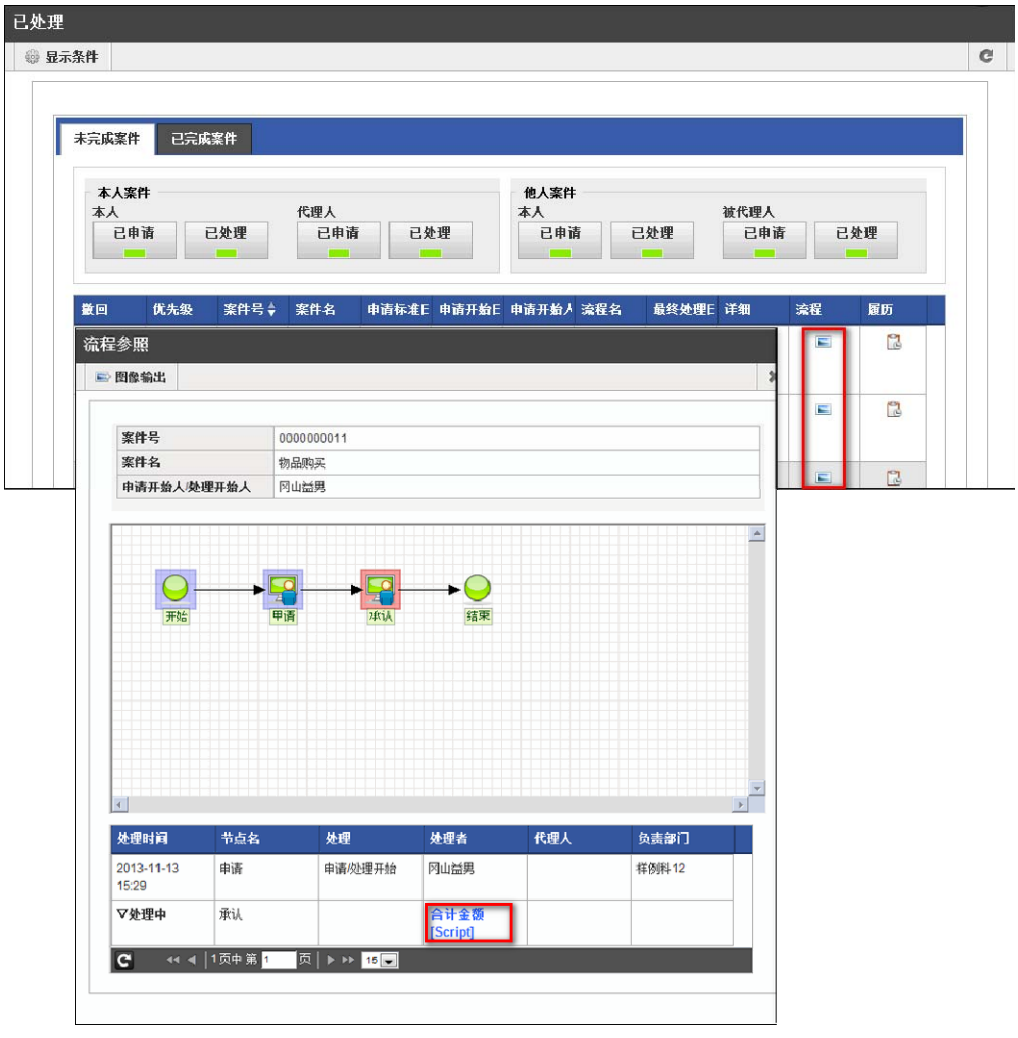
7.2.4 执行示例程序

7.2.3 按照“示例的执行准备 ”中生成的流程定义来进行申请。

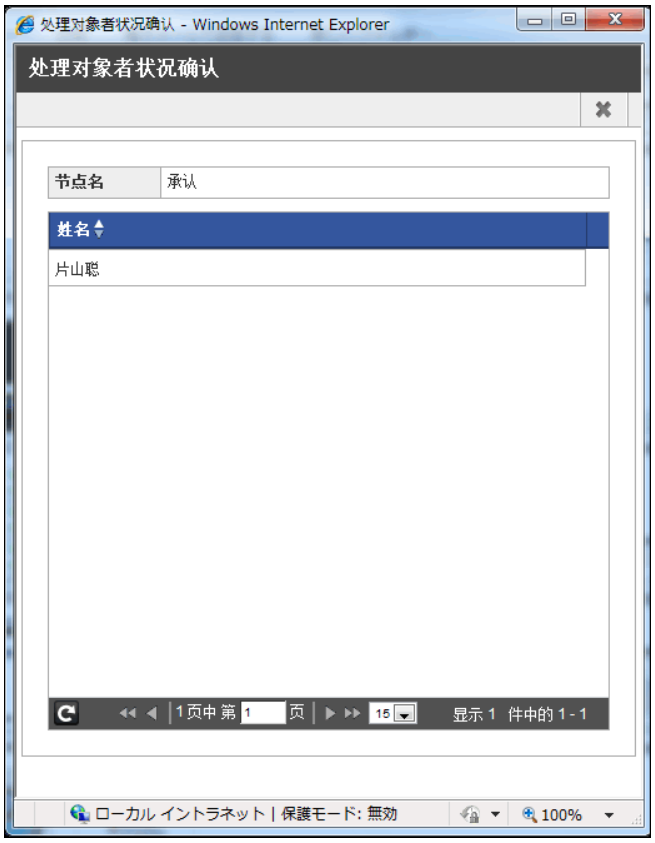


确认审批节点的处理对象者是否根据由“数量”和“金额”算出的“合计金额”发生了变化。

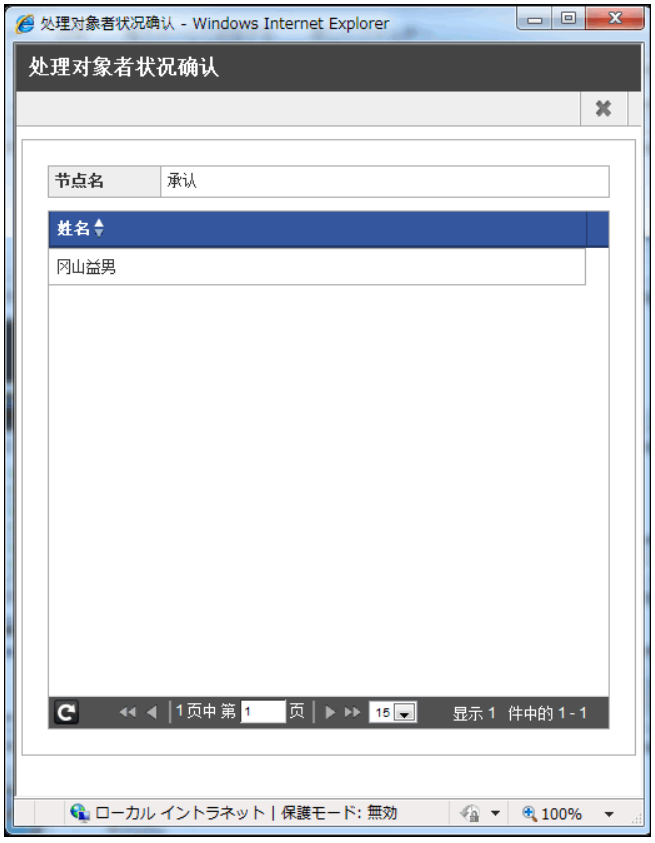
从“已处理”一览画面参照已完成申请的案件的流程。



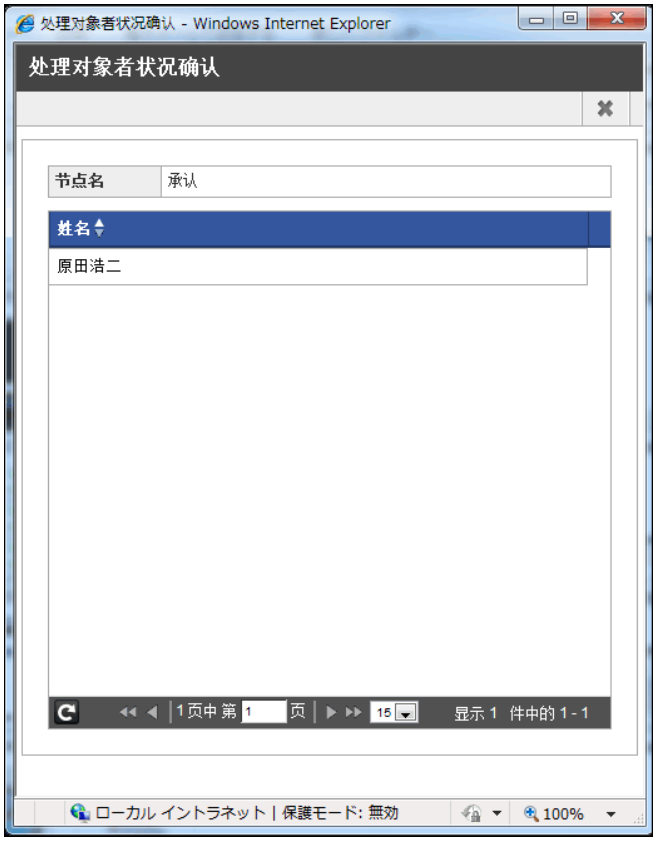
■ 合计金额为 1 万日元以下时



■ 合计金额为 1 万日以上 5 万日元以下时



■ 合计金额为 5 万日以上时



确认对象处理者根据“合计金额”的不同而不同。

7.2.5 关于处理对象者插件

若要生成处理对象者插件，需要生成下述 3 个文件。

7.2.5.1 “plugin.xml”

“plugin.xml”是由“PluginManager”管理的文件。

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin>
    <extension point="jp.co.intra_mart.workflow.plugin.authority.node.approve" >

        <authority
            name="%jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.script"
            id="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.script"
            version="7.2.0"
            rank="910"
            enable="true">
            <configPage>
                <script pagePath="sample/im_workflow/purchase/plugin/authority/item_total/itemTotalConfig">
                    <parameter key="pluginName" value="SAMPLE.IMW.CAP.030" />
                </script>
            </configPage>
            <extend>
                <script
            file="sample/im_workflow/purchase/plugin/authority/item_total/WorkflowAuthorityExecEventListener" />
            </extend>
        </authority>

        <authority
            name="%jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.javaee"
            id="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.item_total.javaee"
            version="7.2.0"
            rank="920"
            enable="true">
            <configPage>
                <javaee applicationId="imw_sample_purchase" serviceId="authority_item_total">
                    <parameter key="pluginName" value="SAMPLE.IMW.CAP.031" />
                </javaee>
            </configPage>
            <extend>
                <java
    class="jp.co.intra_mart.sample.workflow.purchase.plugin.authority.item_total.WorkflowAuthorityExecEventListener" />
            </extend>
        </authority>

    </extension>
</plugin>
```

在处理对象者插件中，重要的是下述元素。

<table>
<tr><td colspan="2"><extension point></td><td>根据要插入处理对象者插件的节点的种类不同，<extension point>也不同。
指定要插入到的节点的<extension point>。</td></tr>
<tr><td rowspan="2"><configPage></td><td><script pagePath></td><td rowspan="2"><configPage>是在“路径定义”画面中，从设定到节点的处理对象者一览画面选择了处理对象者插件时被调用的程序。

可用脚本开发模式或 JavaEE 开发模式来记述此程序。

使用脚本开发模式来生成此程序时，要在<script pagePath>中指定路径。

使用 JavaEE 开发模式来生成此程序时，要指定 applicationId 和 serviceId。</td></tr>
<tr><td><javaee applicationId serviceId></td></tr>
<tr><td>< extend ></td><td><script file></td><td>< extend >中指定的程序是决定处理对象者的程序。

可用脚本开发模式或 JavaEE 开发模式来记述此程序。

使用脚本开发模式来生成此程序时，要在<script file>中指定路径。

使用脚本开发模式来生成此程序时，要在<java class>中指定 package。</td></tr>
<tr><td></td><td><java class></td></tr>
</table>

示例如下所示。

- 审批节点

```
<./plugin/jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve/plugin.xml>
```

- 审批节点

```
<./plugin/jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.approve.static/plugin.xml>
```

- 动态审批节点

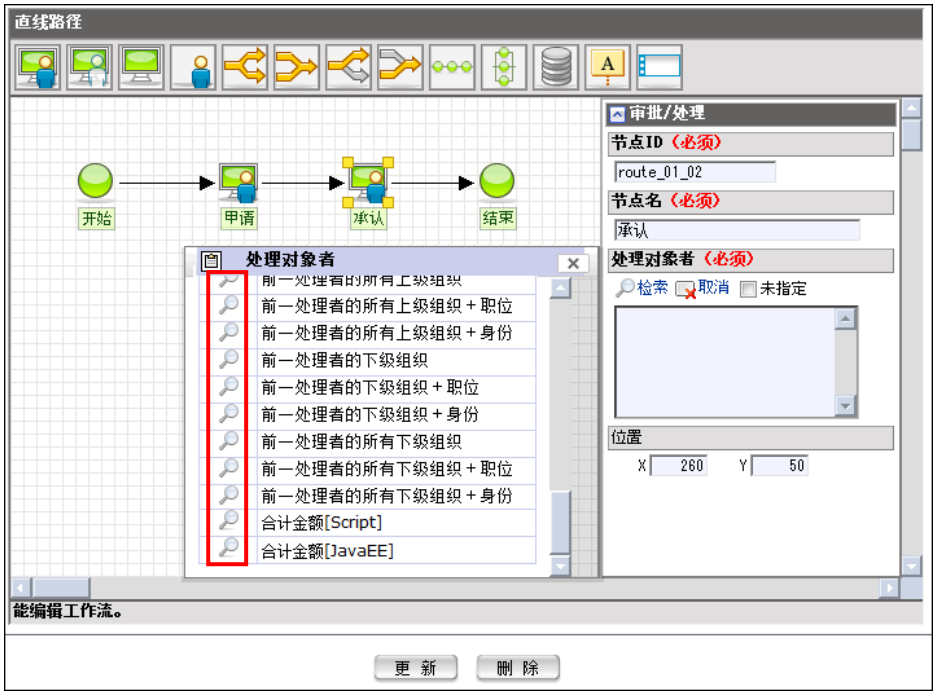
```
<./plugin/jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.dynamic/plugin.xml>
```

- 确认节点

```
<./plugin/jp.co.intra_mart.sample.workflow.purchase.plugin.authority.node.confirm/plugin.xml>
```

7.2.5.2 <configPage>中指定的程序

是在“路径定义”画面中，从设定到节点的处理对象者一览画面选择了处理对象者插件时被调用的程序。



将被选择的对象者插件的信息传递给“路径定义”画面。

示例程序如下。

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/plugin/authority/item_total/itemTotalConfig.html>
<./jssp/src/sample/im_workflow/purchase/plugin/authority/item_total/itemTotalConfig.js>
```

- JavaEE 开发模式

```
<./lib/im_workflow-8.0.0-sample.jar/service-config-imw_sample_purchase.xml>
-----------------------------------------------------------------------------------------------------------------------

<service>
    <service-id>authority_item_total</service-id>
        <controller-class>jp.co.intra_mart.sample.workflow.purchase.plugin.authority.
                item_total.controller.service.ItemTotalConfigServiceController</controller-class>
    <transition-class>jp.co.intra_mart.sample.workflow.purchase.plugin.authority.
                item_total.controller.service.ItemTotalConfigTransition</transition-class>
    <next-page>
      <page-path>/sample/workflow/purchase/plugin/authority/item_total/itemTotalConfig.jsp</page-path>
    </next-page>
  </service>
```

7.2.5.3 在<extend >中指定的程序

在决定处理对象者时执行的程序。

此处指定的程序需要实现下述 3 个方法。

方法	概要
execute	取得处理对象者的方法 在案件到达对象节点时执行。
getTargetUserList	取得处理对象用户一览的方法 在按下“案件操作”-“节点编辑”画面的“状况确认”按钮时显示的“对象者状况确认”画面中使用。
getDisplayName	取得处理对象者插件名称的方法 用于显示插件名称

示例程序如下。

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/plugin/authority/item_total/
                                                    WorkflowAuthorityExecEventListener.js>
```

- JavaEE 开发模式

```
<%示例程序目录%/src/main/java/jp/co/intra_mart/sample/
                workflow/purchase/plugin/authority/item_tota/WorkflowAuthorityExecEventListener.ja
```

7.3 画面输入信息的保持

此处记载的内容对于下述观点是通用的。

- 开发模式
- 客户端类型

在申请画面、临时保存画面、申请（申请案件）画面、再申请画面、处理画面、确认画面中，通过点击“关闭”链接（PC 用画面）或“返回”链接（智能手机用画面）关闭各画面后重新显示该画面时，能够在保持已输入内容的状态下显示画面。

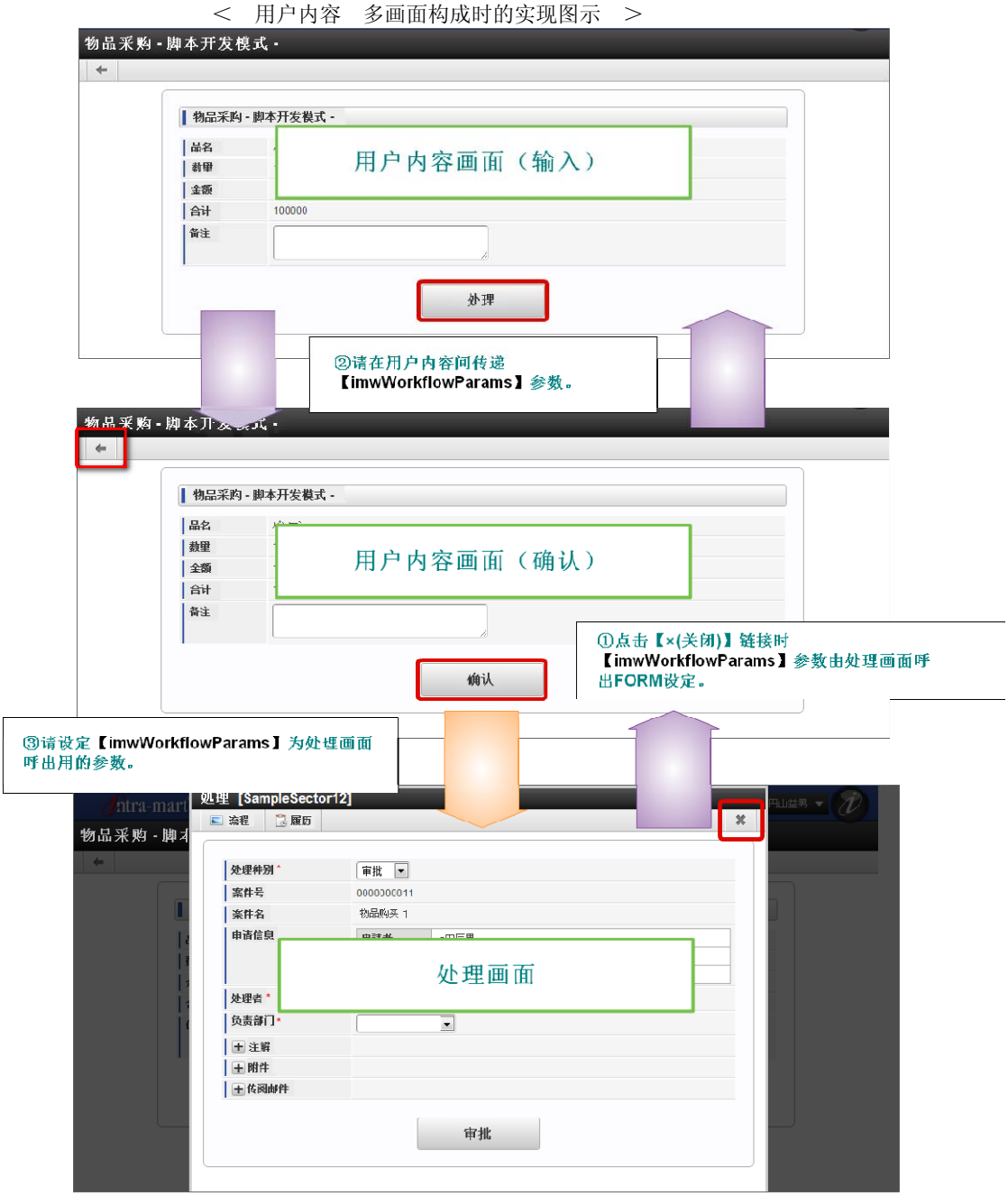
本功能的式样概要如下。

- 按下各处理画面的“关闭”“返回”链接时，对于用调用方用户内容内的画面调出用标签库生成的 FORM 添加名为“imwWorkflowParams”的 hidden 标签，在该标签内保存输入的信息。
- 再次显示画面时，若请求参数含有“imwWorkflowParams”，则根据画面的初始显示处理中所保持的信息执行复原显示

由于是根据所传递的请求参数进行输入信息的重新显示，因此在用户内容是由单一画面构成时不需要去关注，但由多个画面构成时需要进行下述应对。

关闭各处理画面并进行用户内容间的画面迁移，若需要在保持已输入内容的状态下进行各处理画面的重新显示时，请在用户内容间传递“imwWorkflowParams”参数，并明确地在各处理画面显示用的标签库内容 hidden 标签中记述“imwWorkflowParams”参数。

＜　用户内容　多画面构成时的实现图示　＞



7.4 指定从调出画面调用的回调函数

此处记载的内容对于下述观点是通用的。

- 开发模式
- 客户端类型

在申请画面、临时保存画面、申请（申请案件）画面、再申请画面、处理画面、确认画面中，可指定通过点击“关闭”链接（PC用画面）或“返回”链接（智能手机用画面）关闭各画面时的回调函数。　　另外若未进行在“7.5 处理完成后的画面迁移 ”中记载的参数（imwNext～）指定回调函数时，也会在IM-Workflow提供的各处理（申请/再申请/申请（申请案件）/临时保存/处理/确认）画面完成后执行。

对执行调用方的用户内容画面的函数的方法进行说明。

7.4.1 实现例

示例中提供的“物品采购”申请书中，执行用 GreyBox 显示的申请画面关闭处理时，在“物品采购”申请书中定义的函数被作为回调函数执行的例子。

此外，只准备了 PC 用画面的示例。
智能手机用画面的全体流程也相同。请注意，在实现中使用的标签库和 Client-side JavaScript API 不同。

下述程序是为了实现回调函数的执行而记述的程序。

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/apply_callback.html>
```

- JavaEE 开发模式

```
<（展开的 war）/sample/im_workflow/purchase/apply_display.jsp>
```

将上述文件变更为下述文件名并覆盖保存，即可在申请画面对本功能进行动作确认。

- 脚本开发模式

```
<./jssp/src/sample/im_workflow/purchase/screen/apply.html>
```

- JavaEE 开发模式

```
<（展开的 war）/sample/im_workflow/purchase/apply.jsp>
```

通过记述下述处理，即可实现对回调函数的调用。

```
<imart type="head">

<imart type="workflowOpenPageCsjs" />
<script type="text/javascript">
function onClickOpenPage(pageType) {
    if (pageType != "1") {
        if(!inputCheck()) {
            return;
        }
    }

    workflowOpenPage(pageType, callbackFnc);
}

function callbackFnc() {
    alert("Callback function is executed.");
}

.
.
.

<imart type="form" name="backForm" method="POST" page=$data.imwCallOriginalPagePath>
    <imart type="hidden" imwCallOriginalParams=$data.imwCallOriginalParams />
</imart>
```

在 IM-Workflow 提供的各处理（申请/再申请/申请（申请案件）/临时保存/处理/确认）画面完成处理后回调函数被执行时，回调函数可接受所处理案件的信息作为参数。

```
function callbackFnc(result) {
    alert("Callback function is executed.");
    alert( result.imwSystemMatterId );    // 系统案件 ID
    alert( result. imwUserDataId );    // 用户数据 ID
}
```

处理种别和可接受信息之间的关系如下。

处理种别	系统案件 ID imwSystemMatterId	用户数据 ID imwUserDataId
申请	○	-
再申请	○	-
申请（申请案件）	○	-
临时保存	-	○
处理	○	-
确认	○	-

＜ 「○」 ： 可取得 / -: 不可取得＞

7.4.2 非同步执行标准画面时的注意点

从 IM-Workflow 版本 8.0.4 中添加了非同步进行标准画面处理的功能。
此功能有效时，在标准画面的调用方所指定的回调函数的动作有所不同。

IM-Workflow 提供的各处理在作为非同步被受理后就会向各处理画面发送处理结束的通知。就是说几乎在处理刚刚开始后就会发送处理结束的通知。
因此，在执行标准画面的调用方画面指定的回调函数时各处理还未完成的可能性相当高。所以当处理种别是申请时，无法接受系统案件 ID。

7.4.3 特别记载事项

7.4.3.1 IM-Workflow 8.0.2 版的改进

在 IM-Workflow 8.0.2 版中，对连续处理/连续确认中的回调函数调用动作式样进行了改进。

- 到 IM-Workflow 8.0.1 版为止的动作式样
  - 无论是否指定了回调函数，都不执行回调函数。
- 从 IM-Workflow 8.0.2 版开始的动作式样
  - 7.5 不指定“处理完成后的画面迁移”中记载的参数（imwNext～）时，在各处理结束后执行回调函数。

※ IM-Workflow 智能手机版中，由于没有连续处理/连续确认功能，不需在意上述记述。

7.5 处理完成后的画面迁移

此处记载的内容对于下述观点是通用的。

- 开发模式
- 客户端类型

IM-Workflow 提供的各处理（申请/再申请/申请（申请案件）/临时保存/处理/确认）画面处理完成后，可迁移至任意画面。

7.5.1 用于指定要迁移到的画面的参数

IM-Workflow 提供的各处理（申请/再申请/申请（申请案件）/临时保存/处理/确认）画面在调出时，若在“workflowOpenPage”标签的属性中记述下述参数，即可指定处理结束后要迁移到的画面。

No	参数（物理名）	省略	说明
1	imwNextScriptPath	可	处理完成后要迁移到的画面的脚本路径 处理后要迁移到的画面是用脚本开发的画面时，需要指定。
2	imwNextApplicationId	可	处理完成后要迁移到的画面的应用程序 ID 处理后要迁移到的画面是用 javaEE 开发的画面时，需要指定。
3	imwNextServiceId	可	处理完成后要迁移到的画面的服务 ID 处理后要迁移到的画面是用 javaEE 开发的画面时，需要指定。
4	imwNextPagePath	可	处理完成后要迁移到的画面的页面路径 处理后要迁移到的画面是 JSP 或 Servlet 时，需要指定。

请根据想要实现的画面迁移来决定指定的属性。

- 处理后想要迁移到用户内容的调用方一览画面时
  - 请在“imwNextScriptPath”中指定从一览传递过来的“imwCallOriginalPagePath”。
    ※在连续处理、连续确认的情况下，若存在下一个案件节点，则迁移至相应的用户内容。若不存在下一个案件节点，则迁移至调用方一览画面。
- 处理后想迁移至任意画面时
  - 请在“imwNext～”中指定想要迁移到的画面的路径。
- 想要在处理后执行用户内容独自的回调函数并关闭处理画面或只关闭处理画面时
  - 请不要在“imwNext～”中设定任何内容。

7.5.2 要迁移到的画面可接受的请求参数

根据原处理画面的种类，迁移到的画面可接受下述信息作为请求参数。

No	原处理画面	参数（物理名）	参数（逻辑名）	备注
1	申请/再申请/申请(申请案件)/处理/确认	imwSystemMatterId	系统案件 ID	-
2	临时保存	imwUserDataId	用户数据 ID	-
3	全部	imwCallOriginalParams	调用方参数	可接受与用户内容可从一览画面接受的请求参数相同的值。
4	全部 ※只限在连续处理/连续确认处理中的情况下	imwCallOriginalPagePath	调用方页面路径	可接受与用户内容可从一览画面接受的请求参数相同的值。

7.5.3 特别记载事项

7.5.3.1 IM-Workflow 8.0.2 版的改进

从 IM-Workflow 8.0.2 版开始，对连续处理/连续确认中的画面迁移式样进行了改进。

- 到 IM-Workflow 8.0.1 版为止的动作式样
  - 忽略处理完成后要迁移到的画面的指定。
  - 无论是否指定了处理完成后要迁移到的画面，在处理完成后都会显示下一个案件的用户内容。
- 从 IM-Workflow 8.0.2 版开始的动作式样
  - 若指定了处理完成后要迁移到的画面，处理完成后迁移至该指定的画面。

※ IM-Workflow 智能手机版中，由于没有连续处理/连续确认功能，不需在意上述记述。

7.6 用户内容与连续处理/连续确认的联动方法

此处记载的内容对于下述观点是通用的。

- 开发模式

指定“workflowOpenPage”标签的属性“imwNext～”，在 IM-Workflow 处理后迁移至任意画面（调用方一览画面以外的画面）时，或不指定“workflowOpenPage”标签的属性“imwNext～”，不进行 IM-Workflow 处理后画面迁移时，对用户内容与连续处理/连续确认的联动方法进行说明。

此外，IM-Workflow 智能手机版中，由于没有连续处理/连续确认功能，不需在意此处的记述事项。

7.6.1 继续执行连续处理/连续确认

继续进行连续处理、连续确认，若要迁移至与下一个案件节点相对应的用户内容画面时，请按如下方法实现。

- 请迁移至从一览传递过来的“imwCallOriginalPagePath”所指示的画面。
- 请将从一览传递过来的“imwCallOriginalParams”设定为要迁移到的画面的请求参数。

7.6.2 中断连续处理/连续确认

中断连续处理、连续确认，若要迁移至从一览传递过来的“imwCallOriginalPagePath”所指示的画面时，请在迁移至“imwCallOriginalPagePath”的画面之前，在会话中删除客户端的固有信息。
会话键为“ IMW_LAST_PROCESSED_MATTER_INFO_IN_SERIAL”。

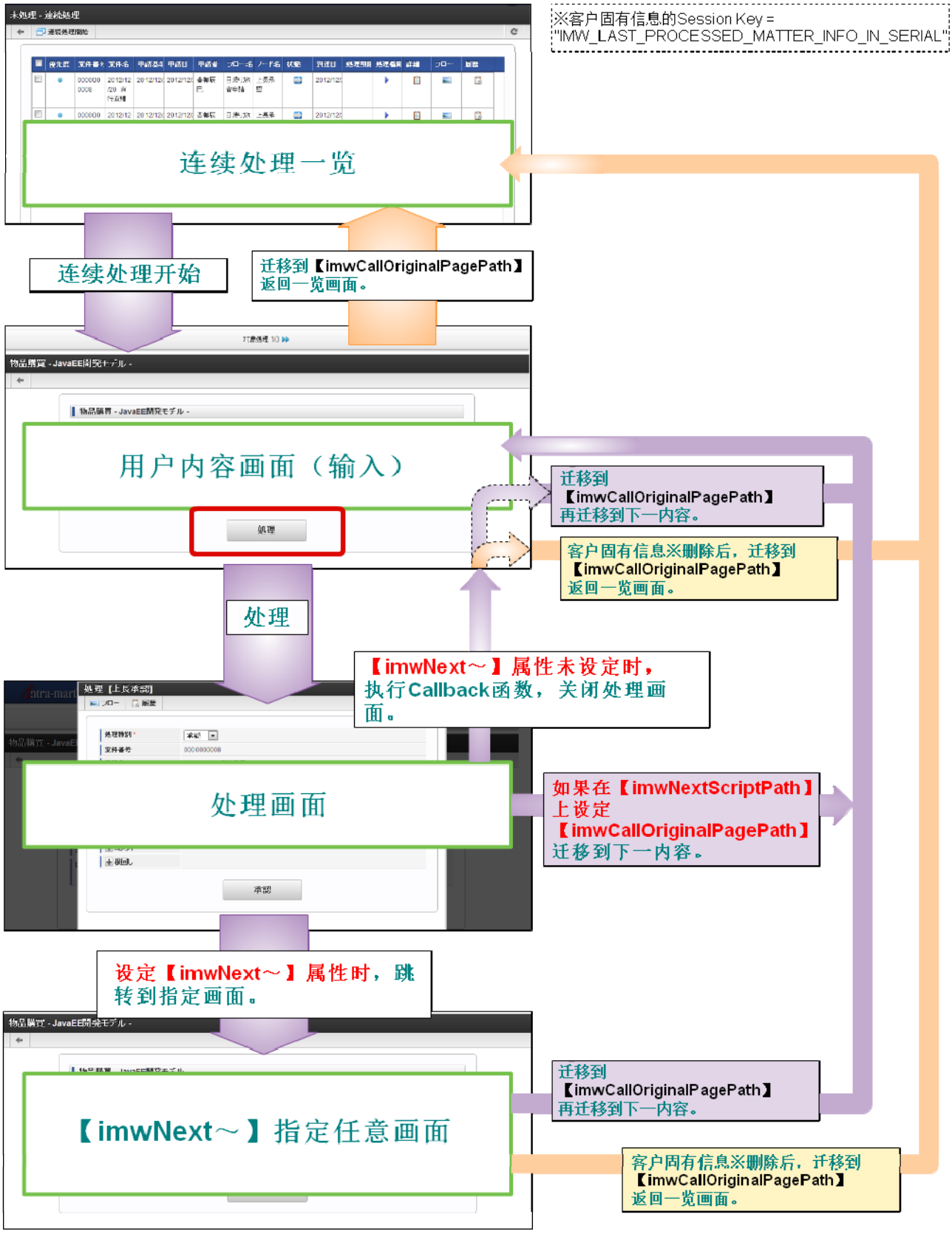
在脚本开发模式中从会话中删除客户端固有信息时，使用下述方法。

```
Client.remove(Strign key)
```

在 javaEE 开发模式中从会话中删除客户端固有信息时，使用下述方法。

```
HttpSession.removeAttribute(java.lang.String name)
```

连续处理时的画面迁移如图所示。此外，连续确认时的画面迁移也相同。



7.7 也可将PC版用户内容用于智能手机用画面

此处记载的内容对于下述观点是通用的。

- 开发模式

对于将生成的 PC 版用户内容画面当成智能手机用画面来动作的方法进行说明。
使用此方法后，可在同一个画面提供 PC 版用户内容和智能手机版用户内容。

但是，在智能手机终端上显示 PC 版用户内容时，有各种各样的限制事项。
因此，推荐分别实现 PC 版用户内容和智能手机版用户内容。

详情请参照“intra-mart Accel Platform 发行说明“的限制事项”。

7.7.1 必要工作

必需设定主表、修正实现。

7.7.1.1 设定主表定义的智能手机用画面

从“网站 Map”-“主表定义-内容定义”选择与设定对象的内容定义相对应的“画面”。
请新建生成或编辑智能手机用的画面定义。
作为画面路径，请将 PC 版用户内容指定为智能手机用画面。

之后，请根据需要进行流程定义的个别设定等操作。

主表定义的新建生成、编辑步骤，请参照“IM-Workflow 管理员操作指南”。

进行上述操作后，若在智能手机终端上显示了对象的流申请画面，就能够显示 PC 版用户内容。

但是，这样就会将智能手机用画面的主题应用于 PC 版用户内容，且有可能发生布局错乱。

因此，要对 PC 版用户内容的实现进行修正。

7.7.1.2 将客户端类型切换至PC

在用户内容的实现中，需要将客户端类型切换为 PC。

在进行画面显示时的服务器端逻辑中，请使用 ClientTypeSwitcher.oneTimeSwitchTo 将作为用户内容显示的画面的客户端类型无条件切换为 PC。

```
ClientTypeSwitcher.oneTimeSwitchTo("pc");
```

有关 ClientTypeSwitcher，详细信息请参照 API 列表。

作为智能手机用画面动作的 PC 版用户内容全部都是要进行实现修正的对象。

通过进行以上操作，即可避免布局错乱，使得 PC 版用户内容能够在智能手机终端上显示。

在此状态下、只有 IM-Workflow 提供的案件的各处理画面（在 GreyBox 上显示的画面）不能被正常显示（无法显示画面、画面布局混乱）时，请进行以下操作。

7.7.1.3 补充修正

作为为了显示执行工作流处理的画面而设定的 Client-side JavaScript API “workflowOpenPage” 的参数、将从各种一览画面中作为请求参数接收的“画面种别（imwPageType）”**照原样**传递时，需要修正。

客户端类型为智能手机时，可接受并传递值为智能手机用画面的来自各种一览传递的画面种别参数。
请在“workflowOpenPage”的参数中传递 PC 用画面种别的值。

画面种别与客户端类型的对应关系如下表所示。

画面种别 客户端类型	申请	临时保存	申请（申请案件）	再申请	处理	确认
PC	0	1	2	3	4	5
智能手机	10	11	12	13	14	15

intra-mart Accel Platform
IM-Workflow 编程指南

2013/07/01 第 4 版

TEL: 03-5549-2821
FAX: 03-5549-2816
E-MAIL: info@intra-mart.jp
URL: http://www.intra-mart.jp/