**操作手册**

版本控制信息

| 版本 | 日期 | 拟稿和修改 | 说明 |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

常用功能

## Tab标签面板

### HTML标签：div

### 样式（class）

* k-tab：标识样式

### 属性（data-）

* data-active-index：指定初始激活第几个标签页
* data-on-active：指定子面板激活时触发的JS事件处理函数
* data-on-before-active：指定子面板激活前触发的JS事件处理函数，可以在此函数里返回false以阻止页面切换

### 子面板

#### 属性（data-）

data-closeable：true/false，表示该子面板是否可关闭

data-title：tab导航栏显示名称

## Step分步面板

### HTML标签：div

### 样式（class）

* k-step：标识样式

### 属性（data-）

* data-nav-switch：true/false，是否可通过点击分步导航栏切换子面板，默认为false
* data-on-switch：指定子面板切换时触发的JS事件处理函数，此函数接受两个参数：currPanel切换前子面板，nextPanel切换后子面板
* data-on-before-switch：指定子面板切换时触发的JS事件处理函数，可以在此函数里返回false以阻止页面切换
* data-before-switch-next：指定子面板向下一步切换前触发的JS事件处理函数，返回false将阻止页面向下切换。此函数接受两个参数：currPanel切换前子面板，nextPanel切换后子面板
* data-on-switch-next：指定子面板向下一步切换时触发的JS事件处理函数，此函数接受两个参数：currPanel切换前子面板，nextPanel切换后子面板
* data-on-switch-prev：指定子面板向上一步切换时触发的JS事件处理函数，此函数接受两个参数：currPanel切换前子面板，nextPanel切换后子面板

### 子面板

* data-title：tab导航栏显示名称
* data-step-type：next/prev，定义向上一步/下一步按钮

## form表单

### HTML标签：form

### 样式（class）

* k-form：标识样式
* col1：表示一行一个字段排列(最大宽度380px)
* col2：表示每行显示两个字段(最大宽度760px)
* col3：表示每行显示三个字段(最大宽度1140px)
* label130:表示label的宽度为130px
* label150:表示label的宽度为150px
* label-inline:表示label的宽度为整个字段的宽度，label与输入框各占一行

### 属性（data-）

### 字段

#### HTML标签：div

#### 样式（class）

* k-field：标识样式（默认365px）
* colspan2：表示当前字段占两个字段宽度（默认730px）
* colspan3：表示当前字段占三个字段宽度（默认1095px）
* clear-left：表示当前字段不跟在前一字段之后，另起一行

#### 属性（data-）

#### Label文本

##### HTML标签：label

##### 样式（class）

* k-field-label：标识样式，必须定义在k-form-field下作为第一个元素

##### 属性（data-）

### 输入控件（data-fieldtype）

#### 文本框扩展text

##### HTML标签：input

##### 样式（class）

* k-field-text：标识样式

##### 属性（data-）

* data-validate-type：验证类型，可选值如下
  + email：验证电子邮箱
  + code：限制只能输入数字文本，左对齐
  + int：验证整型数，右对齐
  + number：验证包含小数点的数字，右对齐
  + money：验证金额，右对齐
  + text:允许输入中文，数字和英文字母，右对齐
* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-min-length：最小长度，提示未达到长度
* data-max-length：最大长度，限制不能输入超长的字符
* data-max-value：指定字段最大值，默认为小于等于，如果需要设置成小于，在设置值的右边加上半角右括号“)”，例：data-max-value="1000)"表示小于1000的值
* data-min-value：指定字段最大值，默认为大于等于，如果需要设置成大于，在设置值的左边加上半角左括号“(”，例：data-min-value="(0"表示大于0的值
* data-disabled：true/false，输入控件是否不可用
* data-show-gbmoney：true/false，定义data-validate-type=”money”时，是否显示大写中文金额
* data-regx：可指定用于验证输入的正则表达式
* data-regx-text：可指定正则验证失败的提示信息
* data-digits：指定有多少位小数（未实现）

#### 显示display

##### HTML标签：span

##### 样式（class）

* k-field-display：标识样式

##### 属性（data-）

* data-dict：指定翻译值的数据字典
* data-value：默认初始值
* data-datatype：指定值的类型，可选值如下
  + number：格式化为带千分位的数字
  + money：格式化为带千分位和两位小数的数字
  + date：格式化为“-”分隔的日期显示
  + time：格式化为带“:”号的时间显示
  + datetime：日期部分格式化为“-”分隔，时间部分格式化为带“：”号显示
  + dict：根据data-dict翻译
* data-renderer：指定用于格式化值的JS函数

#### 日期date

##### HTML标签：input

##### 样式（class）

* k-field-date：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-date-format：指定date控件显示的日期格式，默认为yyyy-dd-mm
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-max-value：指定字段最大值，默认为小于等于，如果需要设置成小于，在设置值的右边加上半角右括号“)”，例：data-max-value="20140101)"表示小于2014-01-01的日期
* data-min-value：指定字段最大值，默认为大于等于，如果需要设置成大于，在设置值的左边加上半角左括号“(”，例：data-min-value="(20140101"表示大于2014-01-01的日期
* data-disabled：true/false，输入控件是否不可用
* data-workday：true则表示只能选择系统工作日日期，其他值则可直接指定工作日方案代码

#### 时间time

##### HTML标签：input

##### 样式（class）

* k-field-time：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-max-value：指定字段最大值，默认为小于等于，如果需要设置成小于，在设置值的右边加上半角右括号“)”，例：data-max-value="121000)"表示小于12:10:00的时间
* data-min-value：指定字段最大值，默认为大于等于，如果需要设置成大于，在设置值的左边加上半角左括号“(”，例：data-min-value="(120000"表示大于12:00:00的时间
* data-disabled：true/false，输入控件是否不可用

#### 单选select

##### HTML标签：input

##### 样式（class）

* k-field-select：标识样式

##### 属性（data-）

* data-value：设置默认值
* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-disabled：true/false，输入控件是否不可用
* data-data：直接指定下拉数据，格式：{‘value1’:’text1’, ‘value2’:’text2’,…}
* data-dict：指定数据字典为下拉数据来源，如果指定了data-exeid，则以data-exeid为数据来源，data-dict只用作翻译值的参考
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数

#### 多选mselect

##### HTML标签：input

##### 样式（class）

* k-field-mselect：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-disabled：true/false，输入控件是否不可用
* data-data：直接指定下拉数据，格式：{‘value1’:’text1’, ‘value2’:’text2’,…}
* data-dict：指定数据字典为下拉数据来源，如果指定了data-exeid，则以data-exeid为数据来源，data-dict只用作翻译值的参考
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数
* data-list-max-width：指定下拉框的最大宽度
* data-item-max-width：指定下拉框的每一项的最大宽度

#### 单选树tree

##### HTML标签：input

##### 样式（class）

* k-field-tree：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-disabled：true/false，输入控件是否不可用
* data-dict：树控件的dict只用作翻译值的参考
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数
* data-async：true/false，指定是否异步加载数据
* data-select-branch：true/false，是否能选择分支节点

#### 多选树mtree（未实现）

##### HTML标签：input

##### 样式（class）

* k-field-mtree：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-on-focus：定义获取焦点事件
* data-on-blur：定义失去焦点事件
* data-disabled：true/false，输入控件是否不可用
* data-dict：树控件的dict只用作翻译值的参考
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数
* data-async：true/false，指定是否异步加载数据

#### 复选checkbox

##### HTML标签：div

##### 样式（class）

* k-field-checkbox：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-name：类似input标签的name属性
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-disabled：true/false，输入控件是否不可用
* data-data：直接指定下拉数据，格式：{‘value1’:’text1’, ‘value2’:’text2’,…}
* data-dict：指定数据字典为下拉数据来源，如果指定了data-exeid，则以data-exeid为数据来源
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数

#### 单选radio

##### HTML标签：div

##### 样式（class）

* k-field-checkbox：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-name：类似input标签的name属性
* data-validate：指定一个JS函数，用于输入验证
* data-on-change：定义修改事件
* data-disabled：true/false，输入控件是否不可用
* data-data：直接指定下拉数据，格式：{‘value1’:’text1’, ‘value2’:’text2’,…}
* data-dict：指定数据字典为下拉数据来源，如果指定了data-exeid，则以data-exeid为数据来源
* data-exeid：指定exeid查询为下拉数据来源
* data-params：指定固定的查询参数值
* data-url：指定数据来源url
* data-value-field：指定查询数据中作为值的字段名称
* data-display-field：指定查询数据中作为文本显示的字段名称，可以指定多个字段名（以半角逗号分隔），那么显示的时候就以data-display-separator分隔符串起来
* data-display-separator：指定显示多个字段值的时候字段值之间的分隔符，默认为“-”
* data-on-beforeload：指定加载数据之前触发调用的JS函数
* data-on-afterload：指定加载数据之后触发调用的JS函数

#### 多输入框mInput

##### HTML标签：div

##### 样式（class）

* k-field-mInput：标识样式

##### 属性（data-）

* data-allowblank：true/false，是否允许为空
* data-disabled：true/false，输入控件是否不可用
* data-validate-type:money,number，支持平台的各种文本类型
* data-type:与data-validate-type的类型一致，支持平台的各种文本类型
* data-max-value:控件的该属性主要是针对下拉框的输入框的类型约束，如最大值为99，则下拉框输入值不能超过99
* data-min-value:同上
* data-max-length:同上
* data-min-length:同上
* data-on-change：定义修改事件

#### 切换开关bswitch

##### HTML标签：input

##### 样式（class）

* k-field-bswitch：标识样式

##### 属性（data-）

* data-on-value：开关为开状态的值，默认为true
* data-off-value：开关为关状态的值，默认为true
* data-value：设置默认值
* data-on-change：定义修改事件
* disabled：true/false，输入控件是否不可用
* data-size：尺寸大小，'mini', 'small', 'normal', 'large'，默认'mini'
* data-animat：开关按钮是否动画 true, false
* data-on-color：开关按钮左边的颜色 'primary', 'info', 'success', 'warning', 'danger', 'default'
* data-off-color：开关按钮右边的颜色 'primary', 'info', 'success', 'warning', 'danger', 'default' 'default'
* data-on-text：开关按钮左边的文字，默认'ON'
* data-off-text：开关按钮右边的文字，默认'OFF'
* data-label-text：开关按钮中间的Label文本， 默认' '

## grid列表

### HTML标签：table

### 样式（class）

* k-grid：标识样式

### 属性（data-）

* data-checkbox：是否多选
* data-checkbox-id：指定查询结果集的主键字段，多个字段用半角逗号分隔
* data-page-size：每页记录数
* data-autoload：是否自动加载数据
* data-exeid：数据来源SQL
* data-url：数据来源地址
* data-params：查询的固定参数值
* data-on-rowselect：绑定行选中事件
* data-on-db-click：绑定行双击事件
* data-on-before-load：绑定数据加载前执行事件，如果JS里返回false，则阻止数据加载
* data-on-after-load：绑定数据加载完成后执行事件
* data-treegrid：true/false，是否为树列表，默认为false
* data-treechild：树列表的每一列的唯一标识（以后会去掉）
* data-treedisplay：树列表导航条显示数据
* data-layout-fixed:设置列表的宽度与屏幕的宽度是否一致

Edit属性未实现

* data-editable：true/false，是否可编辑，可编辑的列表字段需要指定editor
* data-edit-with：什么情况下进入编辑状态，可选值如下
  + click：单击进入编辑状态，缺省值
  + dbclick：双击进入编辑状态
* data-edit-mode：编辑模式，可选值如下
  + inline：直接在列表行上编辑，缺省值
  + popup：弹窗式
* data-edit-on-save：编辑状态保存时调用
* data-edit-on-cancel：编辑状态取消时调用
* data-padding：当data-type定义为double时，可使用该属性设置小数位不足时填0的位数

### 列表字段

#### HTML标签：td

#### 样式（class）

* k-grid-column：这个要不要无所谓。。。
* k-align-left：指定该列数据左对齐排列
* k-align-right：指定该列数据右对齐排列
* k-align-center：指定该列数据居中排列

#### 属性（data-）

* data-header：指定该列列表头显示的数据
* data-name：指定该列的数据列表name，
* data-type：指定该列的数据类型
  + date
  + time
  + datetime
  + money
  + double
  + bigdecimal
  + Percent
  + long
  + int
  + string 默认为string类型，
* data-renderer：该列数据显示前的加载事件，返回值将作为显示内容，参数有：row（列数据所在行的数据），k（该列的data-name）,text（该字段的值或者td中的html）
* data-sort：true/false，是否能根据该列排序，默认为true。
* data-hidden：true/false，是否隐藏该列，默认为false
* data-dict：根据字典来回显该列的数据，需要data-type为string

#### 编辑控件（editor）

在td下定义此div，在此div下定义一个输入控件，它将作为此td字段编辑状态下的输入控件

##### HTML标签：div

##### 样式（class）

k-grid-editor：标识样式

## treegrid列表

## button按钮

### HTML标签：button/a

### 样式（class）

* k-btn：标识样式
* k-btn-collapse：在.panel-heading下定义具有此样式的对象，既可让面板可折叠

### 属性（data-）

* data-descript：按钮详细说明的提示信息
* data-functype：按钮类型，可选值如下
  + RESET
  + SUBMIT
  + EXPORT
  + POPUP
  + PAGE
* data-menuid：当data-functype=”PAGE”的时候，指定跳转到的页面的menuid
* data-menuname：当data-functype=”PAGE”的时候，指定跳转到的页面的title
* data-icon：当data-functype=”PAGE”的时候，指定跳转到的页面的icon
* data-from：可配置一个CSS选择器字符串，按钮提交时将获取该选择器下所有输入字段作为参数提交
* data-target：操作目标，根据按钮类型有不同的作用
  + RESET：指定目标的CSS选择器，重置目标下的所有表单字段值
  + SUBMIT：指定目标的CSS选择器，提交去刷新的grid列表
  + EXPORT：指定目标的CSS选择器，导出目标列表数据（未实现）
  + POPUP：指定目标的CSS选择器，弹出的目标
  + PAGE：指定弹出的页面URL地址
* data-exeid：提交执行的SQL
* data-subimtUrl：提交请求的地址
* data-confirm：点击提交按钮时是否弹出确认提示框，为true则以按钮的descript为提示内容弹出确认提示，为其他字符串则以这个字符串为提示内容弹出确认提示
* data-params：指定提交的固定参数值
* data-disable-condition：不可用条件JS函数
* data-param-handler：参数处理函数，JS函数里返回false阻止按钮动作
* data-after-success：提交成功之后调用
* data-callback：提交完成之后调用
* data-disabled：是否不可用

## toolbar工具栏

## footbar底部栏

## API

### K.init.form([selector])

初始化form对象

### K.init.grid([selector])

初始化k-gird对象

### K.init.tab([selector])

初始化k-tab对象

### K.init.step([selector])

初始化k-step对象

### K.init.treegrid([selector])

### K.field.reloadData(selector | $field [, params])

重新加载字段的下拉数据

#### 语法

|  |
| --- |
| K.field.reloadData("selector" [, params])  K.field.reloadData($field [, params]) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取）

params

可选项。指定用于查询数据时的参数

#### 示例

|  |
| --- |
| K.field.reloadData('input[name=prod\_code]', {status:'1'}); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.reloadData($field); |

### K.field.allowblank(selector | $field, boolean)

设置字段是否允许为空

#### 语法

|  |
| --- |
| K.field.allowblank("selector", boolean)  K.field.allowblank($field, boolean) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

boolean

必选项。Boolean值，指定是否允许为空

#### 示例

|  |
| --- |
| //指定字段不允许为空  K.field.allowblank('input[name=prod\_code]', false); |
| //设置所有text类型的文本输入框不允许为空  K.field.allowblank('input.k-field-type', false); |
| //指定字段可以为空  var $field = K.field.get('M5011F01', 'prod\_code');  K.field.allowblank($field, true); |

### K.field.disable(selector | $field)

设置表单字段不可编辑

#### 语法

|  |
| --- |
| K.field.disable("selector")  K.field.disable($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.disable('input[name=prod\_code]'); |
| //设置所有text类型的文本输入框不可编辑  K.field.disable('input.k-field-type'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.disable($field); |

### K.field.enable(selector | $field)

设置表单字段可以编辑

#### 语法

|  |
| --- |
| K.field.enable("selector")  K.field.enable($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.enable('input[name=prod\_code]'); |
| //设置所有text类型的文本输入框可以编辑  K.field.enable('input.k-field-type'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.enable($field); |

### K.field.dict(selector | $field, newDict)

设置（修改）字段的数据字典属性并且重新加载新的数据字段下拉数据

#### 语法

|  |
| --- |
| K.field.dict("selector", "newDict")  K.field.dict($field, "newDict") |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取）

newDict

必选项。string值，指定新的数据字典名称

#### 示例

|  |
| --- |
| K.field.dict('input[name=prod\_code]', 'prod\_code\_buy'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.dict($field, ' prod\_code\_buy'); |

### K.field.get(selector | formName[, fieldname])

获取并返回输入字段的jQuery对象

#### 语法

|  |
| --- |
| K.field.get("formName", "fieldname")  K.field.get("selector") |

#### 参数

formName

必选项。字符串，指定字段所在的表单（form标签）的name值

fieldName

必选项。字符串，指定字段名称（input/select/textarea的name或其他标签的data-name值）

selector

必选项。字符串，指定被操作的字段的CSS选择器

#### 示例

|  |
| --- |
| var $field = K.field.get('product', 'prod\_code'); |
| var $field = K.field.get('form[name=product] input[name=prod\_code]'); |
| var $field = K.field.get('input[name=prod\_code]'); |

### K.field.show(selector | $field)

显示表单字段

#### 语法

|  |
| --- |
| K.field.show("selector")  K.field.show($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.show('input[name=prod\_code]'); |
| //显示所有text类型的文本输入框  K.field.show('input.k-field-type'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.show($field); |

### K.field.hide(selector | $field)

隐藏表单字段

#### 语法

|  |
| --- |
| K.field.hide("selector")  K.field.hide($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.hide('input[name=prod\_code]'); |
| //隐藏所有text类型的文本输入框  K.field.hide('input.k-field-type'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.hide($field); |

### K.field.reset(selector | $field)

重置表单字段

#### 语法

|  |
| --- |
| K.field.reset("selector")  K.field.reset($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.reset('input[name=prod\_code]'); |
| //重置name为product的表单下的所有字段  K.field.reset('form[name=product]'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  K.field.reset($field); |

### K.field.value(selector | $field[, value])

获取或设置表单字段值

#### 语法

|  |
| --- |
| K.field.value("selector")  K.field.value($field)  K.field.value("selector", value)  K.field.value($field, value) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取）

value

可选项。如果没有value参数，表示是获取字段值；如果传了value参数，表示设置字段值

#### 示例

|  |
| --- |
| //获取字段值  var v = K.field.value('input[name=prod\_code]'); |
| //获取字段值  var $field = K.field.get('M5011F01', 'prod\_code');  var v = K.field.value($field); |
| //设置字段值  K.field.value('input[name=prod\_code]', 'KF001'); |
| //设置字段值  var $field = K.field.get('M5011F01', 'prod\_code');  K.field.value($field, 'KF001'); |

### K.field.validate(selector | $field)

触发表单字段验证并返回验证结果

如果验证了多个字段，只有所有字段都验证通过才会返回true，只要有一个验证失败都返回false

#### 语法

|  |
| --- |
| K.field.validate("selector")  K.field.validate($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

返回

true/false 。

#### 示例

|  |
| --- |
| var v = K.field.validate('input[name=prod\_code]'); |
| //验证name为product的表单下的所有字段  var v = K.field.validate('form[name=product]'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  var v = K.field.validate($field); |

### K.field.clearerr(selector | $field)

清除表单字段的验证失效提示

#### 语法

|  |
| --- |
| K.field.clearerr("selector")  K.field.clearerr($field) |

#### 参数

selector

必选项。字符串，指定被操作的字段的CSS选择器，选择器得到的结果可以是多个字段

$field

必选项。jQuery对象，指定被操作的字段的jQuery对象（可用K.field.get函数获取），可以是多个字段的jQuery引用对象

#### 示例

|  |
| --- |
| K.field.clearerr('input[name=prod\_code]'); |
| //清除name为product的表单下的所有字段的验证失败提示  K.field.clearerr('form[name=product]'); |
| var $field = K.field.get('M5011F01', 'prod\_code');  var v = K.field.clearerr($field); |

### K.btn.enable(selector|$btn)

显示指定的按钮对象

|  |
| --- |
| K.field.enable("selector")  K.field.enable($btn) |

selector

必选项。字符串，指定被操作的按钮的CSS选择器（此选择器得到的目标必须是个.k-btn对象）

$btn

必选项。jQuery对象，指定被操作的按钮的jQuery对象（此选择器得到的目标必须是个.k-btn对象）

### K.btn.disable(name)

隐藏指定的按钮对象

|  |
| --- |
| K.field.disable("selector")  K.field.disable($btn) |

selector

必选项。字符串，指定被操作的按钮的CSS选择器（此选择器得到的目标必须是个.k-btn对象）

$btn

必选项。jQuery对象，指定被操作的按钮的jQuery对象（此选择器得到的目标必须是个.k-btn对象）

### K.btn.handle(name)

执行指定的按钮对象的handle事件

|  |
| --- |
| K.field.handle("selector")  K.field.handle($btn) |

selector

必选项。字符串，指定被操作的按钮的CSS选择器（此选择器得到的目标必须是个.k-btn对象）

$btn

必选项。jQuery对象，指定被操作的按钮的jQuery对象（此选择器得到的目标必须是个.k-btn对象）

### K.tab.add(selectorTarget | $panelTarget, selector | $panel [, index])

给TAB面板添加子面板

#### 语法

|  |
| --- |
| K.tab.add("selectorTarget", "selector")  K.tab.add("selectorTarget", $panel)  K.tab.add("selectorTarget", "selector", index)  K.tab.add("selectorTarget", $panel, index)  K.tab.add($panelTarget, "selector")  K.tab.add($panelTarget, $panel)  K.tab.add($panelTarget, "selector", index)  K.tab.add($panelTarget, $panel, index) |

#### 参数

selectorTarget

必选项。字符串，指定作为添加子面板的目标TAB面板的CSS选择器（此选择器得到的目标必须是个div.k-tab对象）

$panelTarget

必选项。jQuery对象，指定作为添加子面板的目标TAB面板的jQuery对象（此jQuery的目标必须是个div.k-tab对象）

selector

必选项。字符串，指定要添加到TAB中的子面板的CSS选择器

$panel

必选项。jQuery对象，指定要添加到TAB中的子面板的jQuery对象

index

可选项。整数，指定添加子面板到TAB的第几个子面板后面，index从0开始，即0表示插入第一个子面板，如果要插入到第一个子面板前，请使用-1

#### 示例

|  |
| --- |
| //#tab01必须是个div.k-tab-panel对象  //将id=sub-panel01的HTML对象添加到tab的最后  K.tab.add('#tab01.k-tab-panel', '#sub-panel01'); |
|  |
|  |

### K.tab.remove(selector | $panel, index)

删除TAB面板下的第index位子面板

#### 语法

|  |
| --- |
| K.tab.remove("selector", index)  K.tab.remove($panel, index) |

selector

必选项。字符串，指定TAB面板的CSS选择器（此选择器得到的目标必须是个div.k-tab-panel对象）

$panel

必选项。jQuery对象，指定TAB面板的jQuery对象（此jQuery的目标必须是个div.k-tab对象）

index

必选项。整数，指定要删除第几个子面板，从0开始，即第一个面板的index为0

### K.tab.active(selector | $panelTarget, index)

指定TAB面板下的第几个子面板为显示面板

|  |
| --- |
| K.tab.active($panelTarget, index)  K.tab.active("selector", index) |

selector

必选项。字符串，指定TAB面板的CSS选择器（此选择器得到的目标必须是个div.k-tab对象）

$panelTarget

必选项。jQuery对象，指定TAB面板的jQuery对象（此jQuery的目标必须是个div.k-tab对象）

index

必选项。整数，指定要显示的面板的位置，从0开始，即第一个面板的index为0

### K.step.next(selector | $panelTarget)

从当期STEP子页面跳转到下一子页面

|  |
| --- |
| K.tab.next($panelTarget)  K.tab.next("selector") |

selector

必选项。字符串，指定STEP面板的CSS选择器（此选择器得到的目标必须是个div.k-step对象）

$panelTarget

必选项。jQuery对象，指定STEP面板的jQuery对象（此jQuery的目标必须是个div.k-step对象）

### K.step.prev(selector | $panelTarget)

从当期STEP子页面跳转到上一子页面

|  |
| --- |
| K.tab.prev($panelTarget)  K.tab.prev("selector") |

selector

必选项。字符串，指定STEP面板的CSS选择器（此选择器得到的目标必须是个div.k-step对象）

$panelTarget

必选项。jQuery对象，指定STEP面板的jQuery对象（此jQuery的目标必须是个div.k-step对象）

### K.step.switchto(selector | $panelTarget, index)

从当期STEP子页面跳转到指定位置的子页面

|  |
| --- |
| K.tab.prev($panelTarget, index)  K.tab.prev("selector", index) |

selector

必选项。字符串，指定STEP面板的CSS选择器（此选择器得到的目标必须是个div.k-step对象）

$panelTarget

必选项。jQuery对象，指定STEP面板的jQuery对象（此jQuery的目标必须是个div.k-step对象）

index

必选项。整数，指定要显示的面板的位置，从0开始，即第一个面板的index为0

### K.submit(selector | $panelTarget, json)

提交指定的form表单

|  |
| --- |
| K.submit($panelTarget, json)  K.submit("selector", json) |

selector

必选项。字符串，指定FORM表单的CSS选择器

$panelTarget

必选项。jQuery对象，指定FORM表单的jQuery对象

json

必选项。Json对象，

Json = {

exeid : ‘’, //可选，表单提交执行的exeid

submitUrl:’’, //可选，表单提交执行的url 。如果其值为”bizware”，则json参数增加serviceCode、businessCode，调用后台bizware方法

callback:function(){}, //可选，form表单提交执行成功后的回调函数

afterSuccess:function(){} //可选，form表单提交执行成功后执行的函数

}

### K.form.params(selector | $panelTarget)

返回指定的form表单的参数

|  |
| --- |
| K.form.params($panelTarget)  K.form.params("selector") |

selector

必选项。字符串，指定FORM表单的CSS选择器

$panelTarget

必选项。jQuery对象，指定FORM表单的jQuery对象

返回

false/json ， false：boolean对象，表单字段效验不通过。

Json：json对象，表单的参数值

### K.form.setparams(selector | $panelTarget,json)

设置指定的form表单的参数

|  |
| --- |
| K.form.setparams($panelTarget,json)  K.form.setparams("selector",json) |

selector

必选项。字符串，指定FORM表单的CSS选择器

$panelTarget

必选项。jQuery对象，指定FORM表单的jQuery对象

Json

Json对象，填写到表单里面的参数值

### K.form.reset(selector | $panelTarget)

重置指定的form表单的参数

|  |
| --- |
| K.form.setparams($panelTarget)  K.form.setparams("selector") |

selector

必选项。字符串，指定FORM表单的CSS选择器

$panelTarget

必选项。jQuery对象，指定FORM表单的jQuery对象

### K.grid.load(selector | $panelTarget[,params[,emptyFlag[,callback]]])

重新加载列表

|  |
| --- |
| K.grid.load($panelTarget)  K.grid.load("selector")  K.grid.load($panelTarget,params)  K.grid.load("selector",params)  K.grid.load($panelTarget,params,emptyFlag)  K.grid.load("selector",params,emptyFlag)  K.grid.load($panelTarget,params,emptyFlag,callback)  K.grid.load("selector",params,emptyFlag,callback) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

Params

可选项。json对象，指定重新加载列表时的参数

emptyFlag

可选项。Boolean对象，默认为true（默认会清空列表），是否清空列表，后再重新加载数据

Callback

列表加载后的回调函数，参数是查询结果集{results：..,rows:...}，this能获取该k-table对象

### K.grid.page(selector | $panelTarget,page)

跳转到列表的指定页

|  |
| --- |
| K.grid.page($panelTarget,page)  K.grid.page("selector",page) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

Page

必选项。Integer对象，指定要跳转到的页数，从0开始

### K.grid.add(selector | $panelTarget,rows[,index])

给列表添加行

|  |
| --- |
| K.grid.add($panelTarget,rows)  K.grid.add("selector",rows)  K.grid.add($panelTarget,rows,index)  K.grid.add("selector",rows,index) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

rows

必选项。Array对象[{},{}.....]，填进列表的数据集合

Index

可选项。从哪个位置插入列表中。如果要插入第一行，则index传-1

### K.grid.treeload(selector | $panelTarget,position,row,pre\_positon)

重新加载树列表

|  |
| --- |
| K.grid.page($panelTarget,position,row,pre\_position)  K.grid.page("selector",position,row,pre\_position) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

Position（以后版本更新会去掉）

必选项。String对象，跳转到的树列表位置，值为树列表对象指定的data-treechild的相对应的参数值（如sys\_org使用orgno作为data-treechild，position传机构代码就能跳到该机构的子页面）

Row

必选项。Json对象，要跳到的树列表节点的数据列的数据对象。

Pre\_posion

必选项。类似position

### K.grid.getData(selector | $panelTarget)

获取列表的所有数据

|  |
| --- |
| K.grid.getData($panelTarget)  K.grid.getData("selector") |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

返回

Json对象 {results:..., //数据总条数

rows:[{},{}..] //列表数据

}

注意：由于分页查询仅是部分查询，所以查到的数据只有5条，但是总记录数是10条。所以results是10，rows.length是5

### K.grid.setRow(selector | $panelTarget,row[,index])

修改列表中指定行的数据

|  |
| --- |
| K.grid.setRow($panelTarget,row)  K.grid.setRow("selector",row)  K.grid.setRow($panelTarget,row,index)  K.grid.setRow("selector",row,index) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）/或者指定修改的table中某列的css选择器（选择器得到的目标必须是个div.k-grid-tr对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）/或者指定修改的table中某列的jquery对象（此jQuery的目标必须是个div.k-grid-tr对象）

row

必选项。json对象，指定修改的数据

index

Integer对象，当$panelTarget/selector是k-grid对象时必选。当$panelTarget/selector是k-grid-tr对象时不用选

### K.grid.del(selector | $panelTarget,index)

删除列表中指定行

|  |
| --- |
| K.grid.del($panelTarget,index)  K.grid.del("selector",index) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

index

必选项。Integer对象，要删除的列

### K.grid.hideColumn(selector | $panelTarget,column)

隐藏列表中指定列

|  |
| --- |
| K.grid.hideColumn($panelTarget,column)  K.grid.hideColumn("selector",column) |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

column

必选项。要隐藏的列的data-name

### K.grid.selected(selector | $panelTarget)

在data-checkbox=”true”的时候，返回所有选中的data-checkbox-id

|  |
| --- |
| K.grid.hideColumn($panelTarget)  K.grid.hideColumn("selector") |

selector

必选项。字符串，指定table列表的CSS选择器（此选择器得到的目标必须是个div.k-grid对象）

$panelTarget

必选项。jQuery对象，指定table列表的jQuery对象（此jQuery的目标必须是个div.k-grid对象）

### K.index.module(moduieid | index)

切换子系统界面到指定界面

|  |
| --- |
| K.index.module(moduleid)  K.index.module(index) |

moduleid

必选项。String对象，子系统第一级菜单的menuid或moduleid。

index

必选项。number对象，传数字表示切换第几个子系统，从0开始

### K.index.openPage(argJson[,params])

用于打开子页面

|  |
| --- |
| K.index.openPage(argJson)  K.index.openPage(argJson,params) |

argJson

必选项。json对象，

{url : ..., //指定跳转到的页面url

icon :..., //指定title的图标

title : ...., //title显示的内容

colseable:....//打开的页面是否能关闭

}。

params

可选项。json对象，传给打开页面的参数

### K.index.getPage(menuid)

获取子页面（是个在index.jsp里的iframe）的DOM对象

|  |
| --- |
| K.index.getPage(menuid) |

menuid

必选项。指定页面的menuid，

### K.index.getPageK(menuid)

获取子页面的K对象

|  |
| --- |
| K.index.getPageK(menuid) |

menuid

必选项。指定页面的menuid

### K.index.closePage([menuid])

menuid，可选参数，如果不传menuid参数则关闭当前激活页面，传menuid则关闭，指定菜单页面

|  |
| --- |
| K.index.closePage()  K.index.closePage(menuid) |

menuid

可选项。指定页面的menuid

### K.index.home(menuid)

返回主页

|  |
| --- |
| K.index.home(menuid) |

menuid

必选项。主页的menuid

### K.popup(selector|$popup)

弹出指定表单

|  |
| --- |
| K.popup(selector)  K.popup($popup) |

selector

传要弹出的对象的选择器字符串

$popup

要弹出的表单的jQuery对象

### K.popup.close(selector|$popup)

关闭已经弹出的表单

|  |
| --- |
| K.popup.close(selector)  K.popup.close($popup) |

selector

要关闭的K-popup表单对象的选择器字符串

$popup

要关闭的K-popup表单的jQuery对像

## Tools

### Tools.Div(num1，num2)

整除函数，等同于java语言的 "/"操作

|  |
| --- |
| Tools.Div(num1，num2) |

num1

必选项，被除数，

num2

必选项，除数

返回

商

### Tools.accDiv(arg1，arg2)

//除法函数，用来得到精确的除法结果

//说明：javascript的除法结果会有误差，在两个浮点数相除的时候会比较明显。这个函数返回较为精确的除法结果。

//调用：accDiv(arg1,arg2)

//返回值：arg1除以arg2的精确结果

|  |
| --- |
| Tools.accDiv(arg1，arg2) |

arg1

必选项，被除数，

arg2

必选项，除数

返回

精确的商

### Tools.accMul(arg1，arg2)

//乘法函数，用来得到精确的乘法结果

//说明：javascript的乘法结果会有误差，在两个浮点数相乘的时候会比较明显。这个函数返回较为精确的乘法结果。

//调用：accMul(arg1,arg2)

//返回值：arg1乘以arg2的精确结果

|  |
| --- |
| Tools.accMul(arg1，arg2) |

arg1

必选项，

arg2

必选项，

返回

精确的积

### Tools.accAdd(arg1，arg2)

//加法函数，用来得到精确的加法结果

//说明：javascript的加法结果会有误差，在两个浮点数相加的时候会比较明显。这个函数返回较为精确的加法结果。

//调用：accAdd(arg1,arg2)

//返回值：arg1加上arg2的精确结果

|  |
| --- |
| Tools.accAdd(arg1，arg2) |

arg1

必选项，

arg2

必选项，

返回

精确的和

### Tools.makeChars(num,chr)

创建num个chr字符

|  |
| --- |
| Tools.makeChars(num，chr) |

num

必选项，创建的字符个数

chr

必选项，要创建的字符

### Tools.moneyStr(d)

将bigdecimal型的d转换成货币字符串:即每一千分位加","号,保留小数点后两位

|  |
| --- |
| Tools.moneyStr(d) |

d

必选项

### Tools.formatDecimal(d,num,returnString)

格式化d为带有num位小数位(多余的截去，少的补0)的数字返回

|  |
| --- |
| Tools.formatDecimal(d,num,returnString) |

d

必选项

num

必选项

returnString

必选项

### Tools.dateFormater(str|date)

返回yyyy-MM-dd格式的日期字符串

|  |
| --- |
| Tools.dateFormater(str)  Tools.dateFormater(date) |

str

必选项，string对象

date

必选项，date对象

### Tools.dateAddMonth(d,month)

///日期对象添加月份

///参数：

///d : 要添加月份的日期对象

///month : 添加的月份数

|  |
| --- |
| Tools.dateAddMonth(d,month) |

### Tools.dateAddDay(d,days)

//日期对象添加天数

|  |
| --- |
| Tools.dateAddDay(d,days) |

### Tools.trim(str)

去掉前后空格

|  |
| --- |
| Tools.trim(str) |

### Tools.padLeft(str, len, chr)

当str不够len位时，在左边补足chr字符

|  |
| --- |
| Tools.padLeft(str) |

### Tools.padRight(str, len, chr)

当str不够len位时，在右边补足chr字符

|  |
| --- |
| Tools.padRight(str, len, chr) |

### Tools.apply(o,c)

将c对象中的属性复制并覆盖到o对象中

|  |
| --- |
| Tools.apply(o,c) |

### Tools.arrayRemove(arr, items)

\* 移除数组中的某一个或多个元素

\* 此函数将返回一个删除元素后的新数组，并不修改arr

|  |
| --- |
| Tools.arrayRemove(arr, items) |

### Tools.alert(message,icon,okFn)

弹出alert提示框

|  |
| --- |
| Tools.alert(message,icon,okFn) |

message

必选项，string对象，提示信息

icon

可选项，提示的图标，有错误，警告，提示（默认）三种

okfn

可选项，确认函数

### Tools.confirm(message,fn)

弹出确认框

|  |
| --- |
| Tools.confirm(message,fn) |

message

必选项，string对象，提示信息

fn

必选项，确认函数

### Tools.isChinaIDCard(StrNo)

\*判断身份证号码格式函数

\*公民身份号码是特征组合码，

\*排列顺序从左至右依次为：六位数字地址码，八位数字出生日期码，三位数字顺序码和一位数字校验码

|  |
| --- |
| Tools.isChinaIDCard(StrNo) |

### Tools.gbMoney(num)

将小写金额值转成中文大写金额

|  |
| --- |
| Tools.gbMoney(num) |

### Tools.json2str(json,includeFunction,isConfOption)

将JSON对象转换成字符串

参数：includeFunction boolean类型，true则包含function值的属性也转成string，其他值则不转function属性，默认为false

|  |
| --- |
| Tools.json2str(json, includeFunction, isConfOption) |

**1.10.20 Tools.str2json(str)**

将string对象转换成json对象

参数：str，字符串对象

### Tools.str2Date(dateStr)

把yyyyddmm或者yyyy-dd-mm的日期字符串转成日期对象

|  |
| --- |
| Tools.str2Date(dateStr) |

### Tools.date2str(date)

把日期对象转换成YYYYMMDD格式的日期字符串

|  |
| --- |
| Tools.date2str(date) |

### Tools.getDateMonthBetween(date1,date2)

得到两个日期之间的天数 YYYY-MM-DD - YYYY-MM-DD

|  |
| --- |
| Tools.getDateMonthBetween(date1,date2) |

### Tools.addWorkDay(date,days)

查询date之后的第days个工作日

|  |
| --- |
| Tools.addWorkDay(date) |

### Tools.addWorkDayEx(date,days)

查询大于等于date的最近工作日，之后再加上days的那个工作日

若days为0，则返回最近工作日，如date是节假日（周六日），则返回最近的下一个工作日（周一）

|  |
| --- |
| Tools.addWorkDayEx(date) |

高级功能

权限控制

1. 适用场景

角色对界面有无访问权限

角色对界面按钮使用权限

1. 涉及的文件
2. \WebRoot\WEB-INF\tld\sysUser.tld
3. \WebRoot\WEB-INF\conf\spring\sys\context-base.xml
4. \WebRoot\page\M0\M0006.jsp
5. \WebRoot\page\M0\M0006SP01.jsp
6. \WebRoot\page\M0\M0006SP02.jsp
7. \WebRoot\WEB-INF\conf\sql\M0\sql-M0006-execute.xml
8. \WebRoot\WEB-INF\conf\sql\M0\sql-M0006-info.xml
9. \WebRoot\WEB-INF\lib\jsoup-1.8.1.jar
10. 使用方法

权限控制用于控制用户对界面按钮的使用权限，对于需要进行权限控制的按钮需加入以下标签**${sysuer:sysRgt('XXX')}**，如下

<a class="btn btn-success k-btn ${sysuer:sysRgt('M0005U01')}"

data-functype="POPUP"

data-target="#M0005P002"

data-handler="M0005resetinsert"

data-descript="新增用户信息">新增</a>

其中'M0005U01'为ID，ID要求系统内唯一，建议命名规则为界面名称+数字，例如M0001.jsp中可用**${sysuer:sysRgt(**M000101**)}**。

权限设置界面如下：

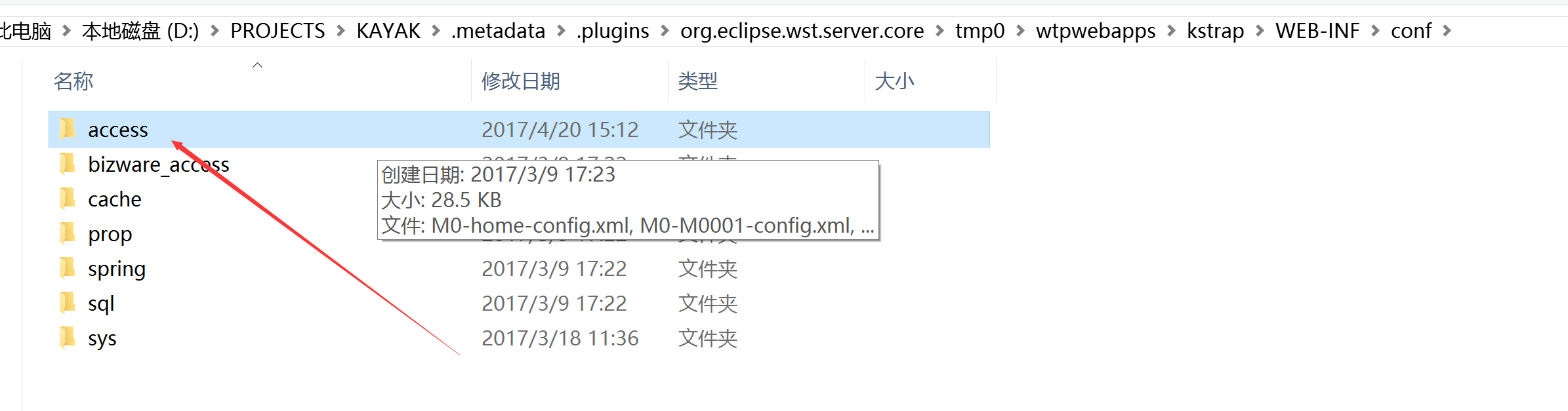


1. 访问控制

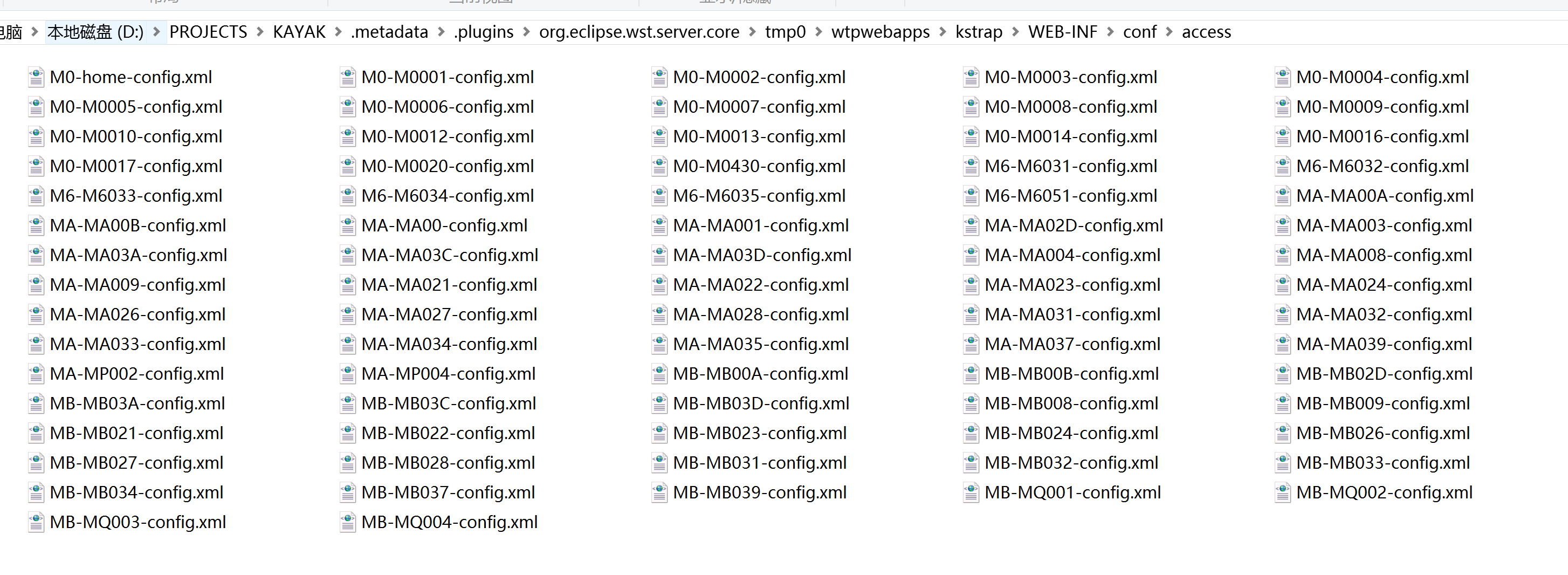
默认情况下，如果不配置访问权限，平台的权限控制仅仅只是对按钮的显示与隐藏，如果行方对权限控制要求高，就需要对每个访问请求做权限控制。将按钮ID与exeid关联起来，在sys\_button\_access表设置，如果是更新类型的exeid，则必须传入按钮ID，如果不传入，则不允许访问，如果传入按钮ID，则会验证按钮ID与当前用户的合法性，验证通过后方可访问。

打开context-base.xml，配置按钮权限配置文件扫描，启动平台会生成基础的配置文件

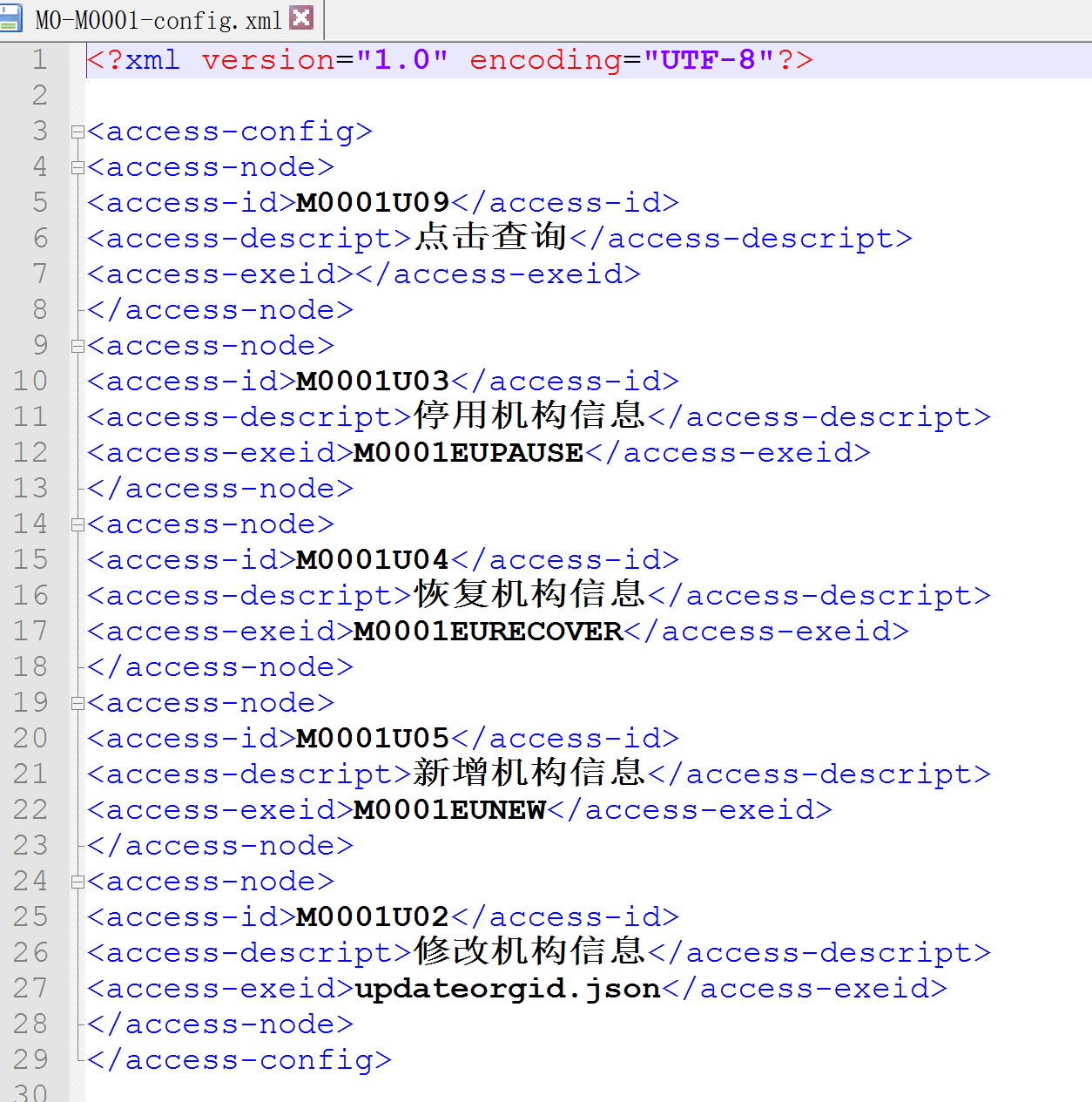




生成的目录在Tomcat webapps下的项目里面，如图所示

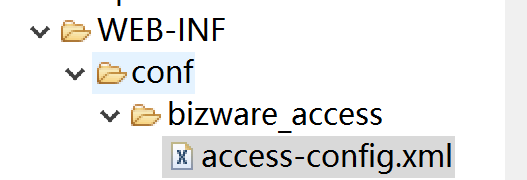


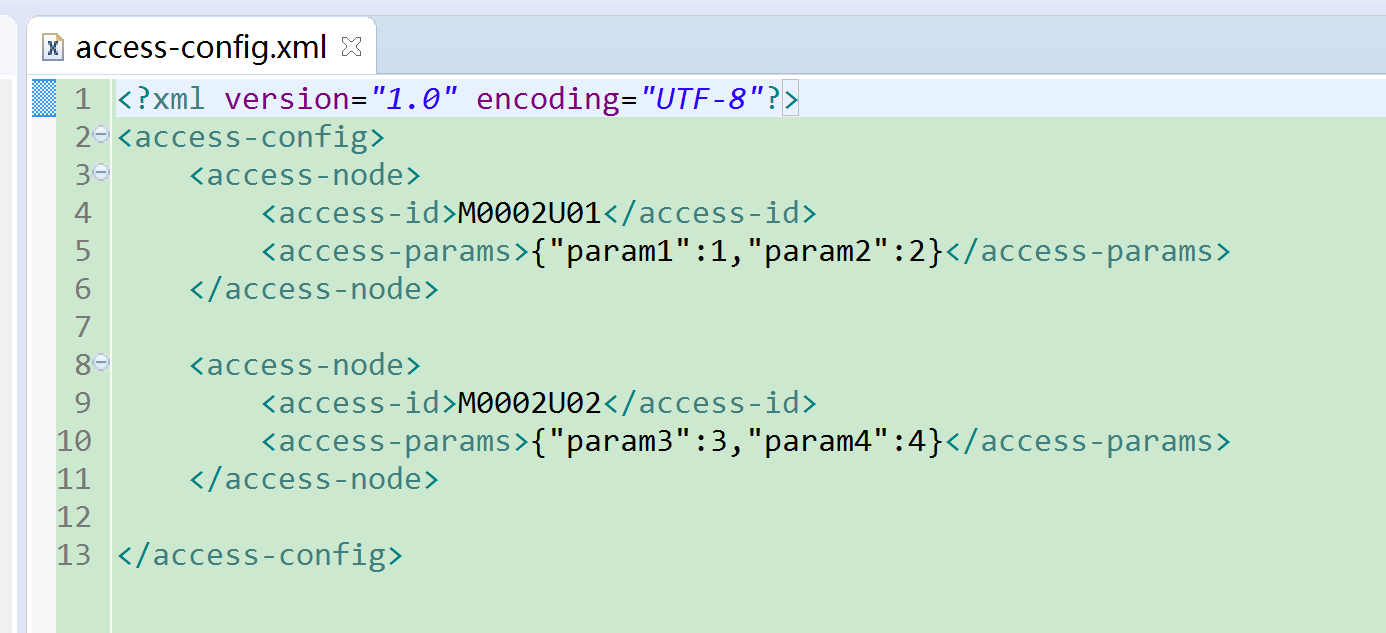
打开后会看到生成了许多配置文件，将该文件复制到项目的对应的conf/access下，打开生成的配置文件，如打开M0-M0001-config.xml



我们会看到有按钮id与对应的access-exeid对应，access-exeid对应按钮id执行的exeid或者url路径，配置好平台所有功能的按钮id与exeid或者url的对应关系即可，一个按钮id对应多个exeid或者url可以使用,分隔。配置好后即可对平台的访问进行控制。

对应bizware的请求，同样需要配置按钮id与请求的对应关系，bizware的访问控制是采用参数判断，配置文件所在目录如图





如图所示，配置了M0002U01的判断参数param1与param2，表示如果是按钮M0002U01发起的bizware请求，param1必须是1并且param2必须是2。具体参数根据具体业务功能配置。

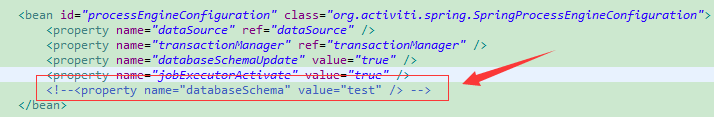
工作流

1. 涉及的文件
2. WebRoot\page\M0\page下
   1. M0012.jsp
   2. M0012P01.html
   3. M0012P01.jsp
   4. M0012P02.jsp
   5. M0013.jsp
   6. M0013P01.jsp
   7. M0014.jsp
   8. M0015.jsp
3. \WebRoot\kstrap下
   1. include.html
4. \WebRoot\WEB-INF\lib下
   1. activiti-bpmn-converter-5.16.3.jar
   2. activiti-bpmn-model-5.16.3.jar
   3. activiti-engine-5.16.3.jar
   4. activiti-image-generator-5.16.3.jar
   5. activiti-process-validation-5.16.3.jar
   6. activiti-spring-5.16.3.jar
   7. commons-io-2.4.jar（替换旧版本）
   8. ezmorph-1.0.1.jar
   9. joda-time-2.1.jar
   10. jsoup-1.8.1.jar
   11. mybatis-3.2.5.jar
   12. thymeleaf-2.1.4.RELEASE.jar
   13. thymeleaf-spring4-2.1.4.RELEASE.jar
   14. unbescape-1.1.0.RELEASE.jar
5. \WebRoot\js\binary下
   1. jquery.jqprint-0.3.js
   2. jquery-migrate-1.1.0.js
6. \WebRoot\WEB-INF\conf\sql\M0下
   1. sql-M0006-execute.xml
   2. sql-M0006-info.xml
   3. sql-M0015-execute.xml
   4. sql-M0015-info.xml
   5. sql-M0101-execute.xml
   6. sql-M0101-info.xml
7. \WebRoot\WEB-INF\conf\sql\sys下
   1. sql-act-execute.xml
   2. sql-act-info.xml
8. \WebRoot\WEB-INF\conf\spring\sys
   1. context-workflow.xml
   2. servlet.xml
9. 配置数据库

若第一次使用工作流，需配置工作流引擎创建工作流需要的表，创建之后，叫配置恢复原本方式，下次启动不需要创建（不可重复创建，若创建表的过程中失败导致表创建不全，需自行到进入数据库，删除ACT\_开头的表，序列，索引等）。

配置如下，

打开\WebRoot\WEB-INF\conf\spring\sys\context-workflow.xml



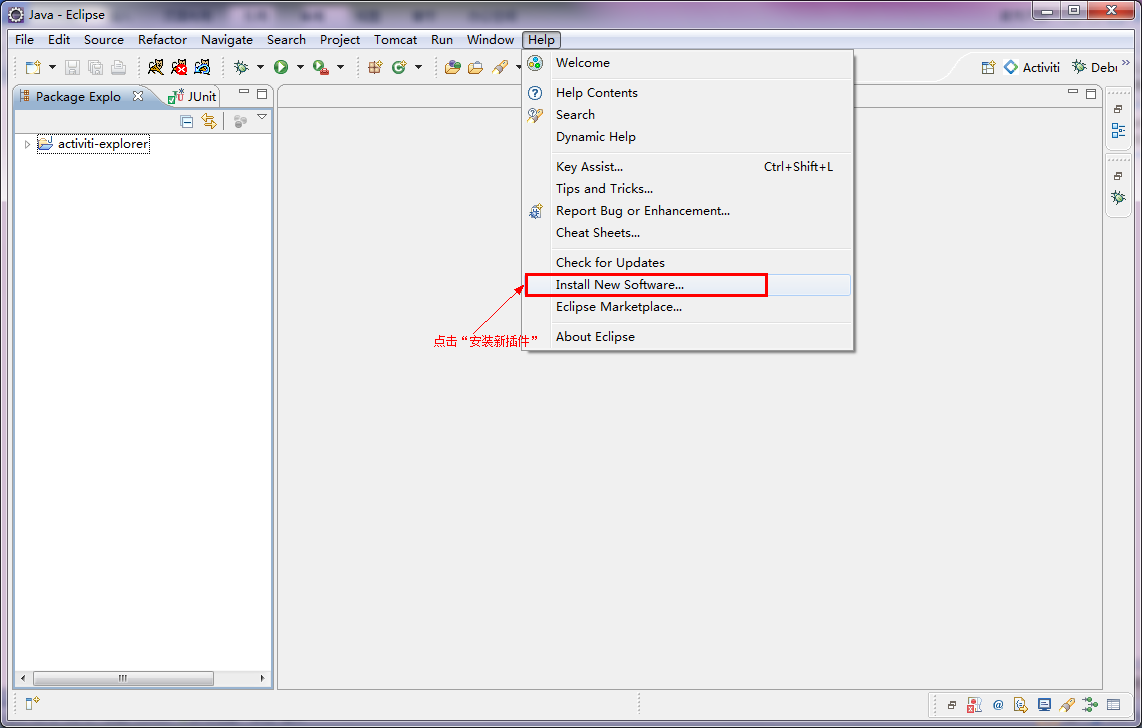
将下面的注释去掉，value修改为数据库的用户名，成功创建后，需在此把注释加上，以免重复创建错误。

1. 安装工作流绘图工具

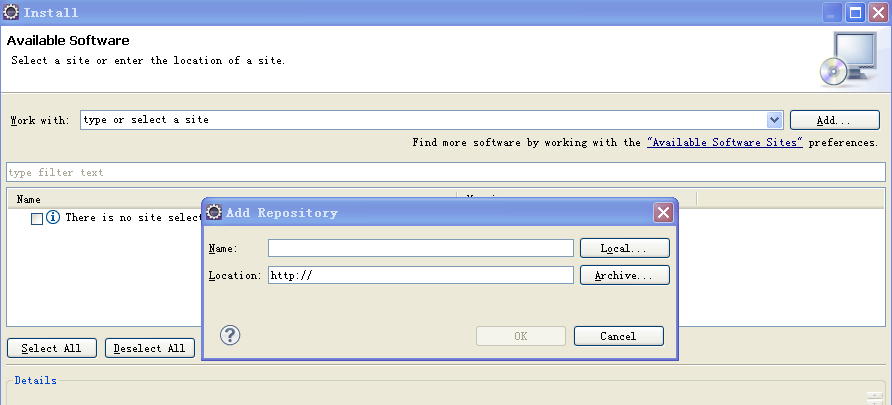
**安装方式一**

在**有网络**的情况下，安装流程设计器步骤如下：

1. 打开 Help -> Install New Software. 在如下面板中:



1. 在如下Install界面板中，点击Add按钮：

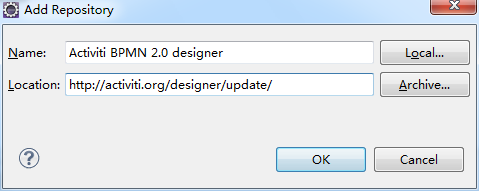


配置新装插件的地址和名称

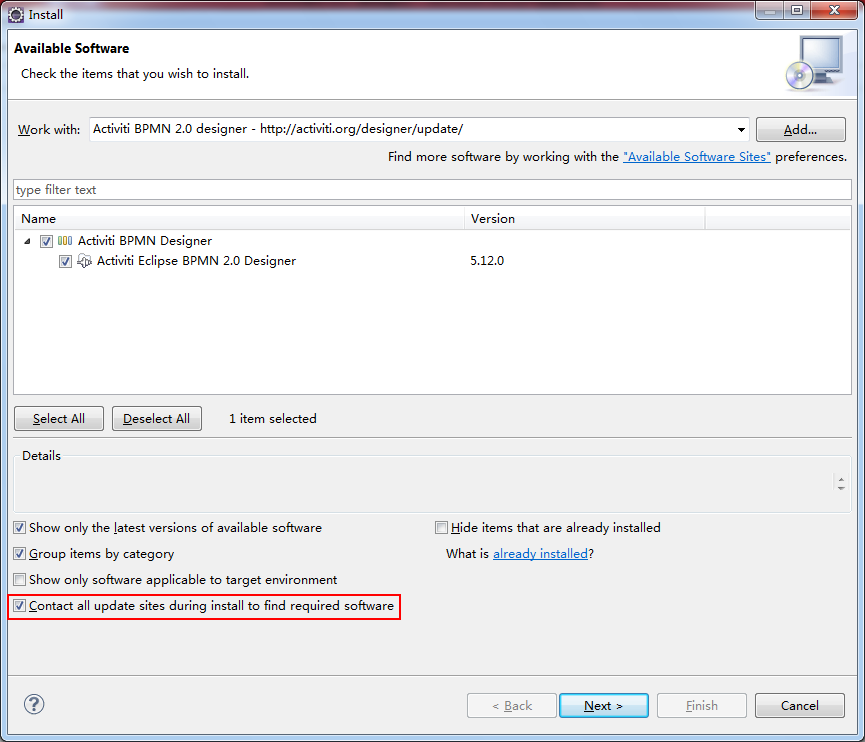
1. 然后填入下列字段

Name: Activiti BPMN 2.0 designer

Location: http://activiti.org/designer/update/



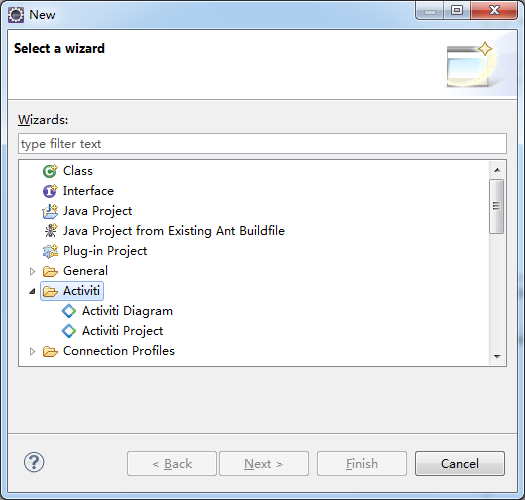
1. 回到Install界面，在面板正中列表中把所有展示出来的项目都勾上：



1. 点击复选框

在Detail部分记得选中 "Contact all updates sites.." , 因为它会检查所有当前安装所需要的插件并可以被Eclipse下载.

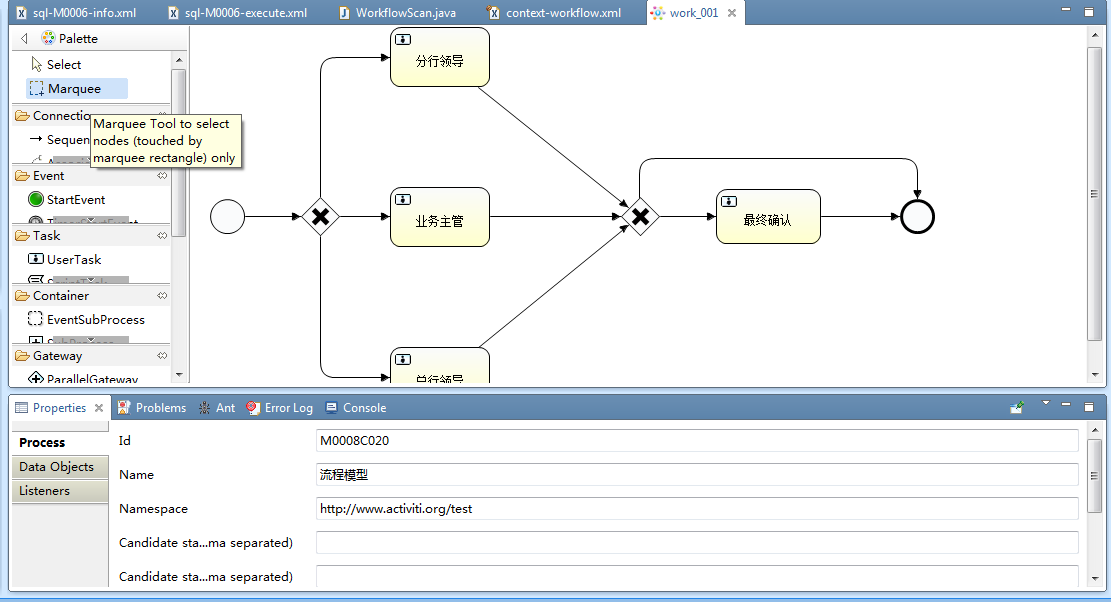
1. 安装完以后，点击新建工程new->Other…打开面板，如果看到下图内容：



说明安装成功了。

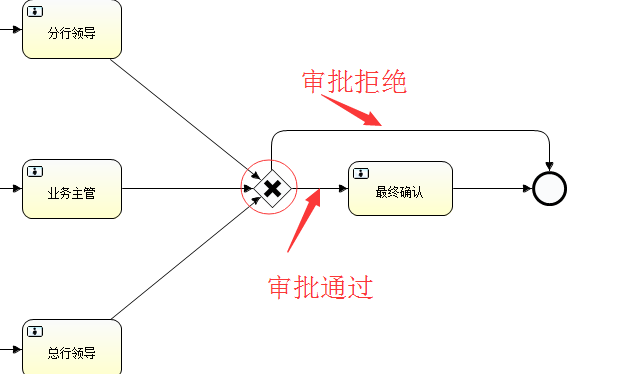
1. 绘制流程图

具体绘制以及绘制的节点可参考\工作流\doc下面文档，要部署的流程图需放到WebRoot\WEB-INF\workflow-bpmn下，并且工作流的ID值必须唯一

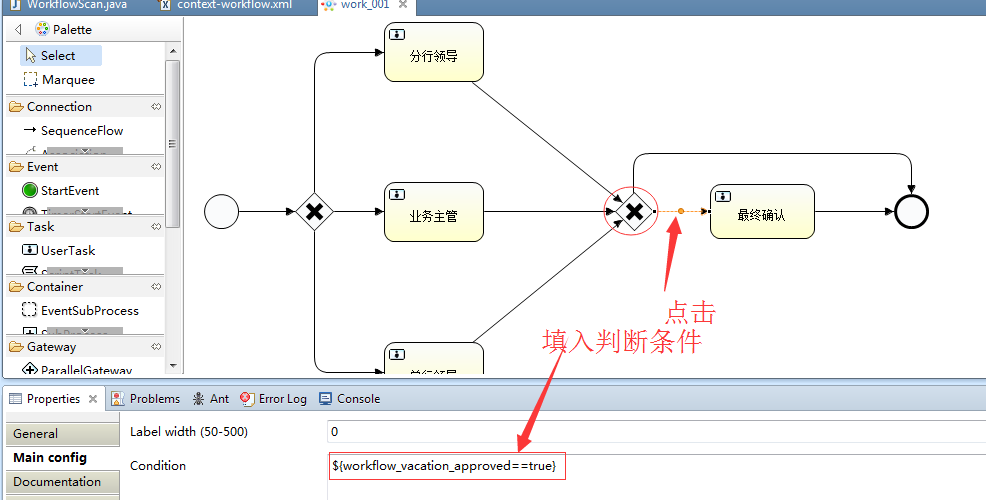
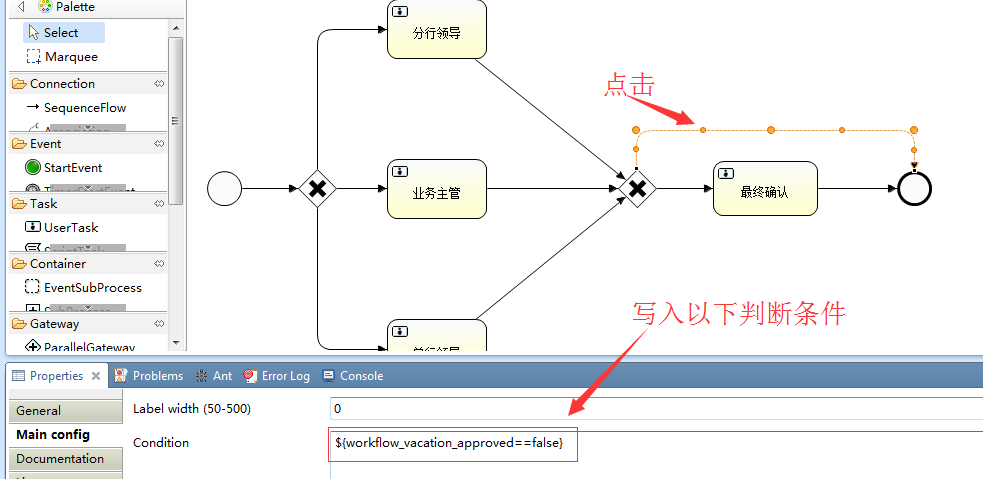


**在绘制流程图时，特别要注意任务到达下一个任务节点之间的转折，通过当前任务审批拒绝时，需直接到流程结束节点。（特别重要）**

**如下图：**



**在判断的选择的两天路线填入判断条件**



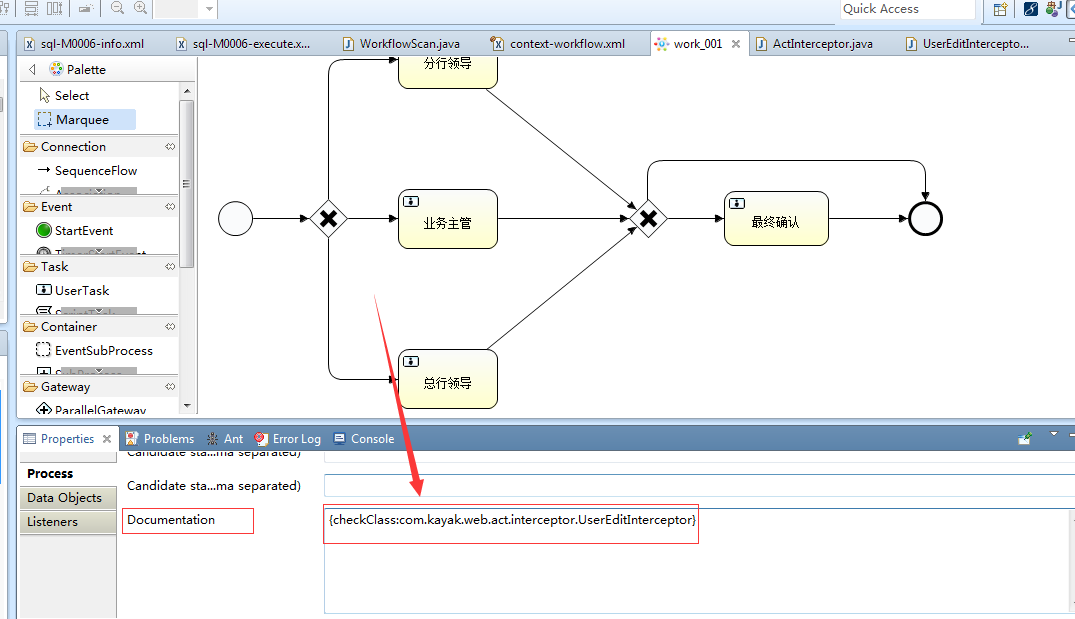
**如果有多个任务，都需要进行如上面的设置。**

1. 实现工作流拦截处理接口

定义一个类，作为工作流处理开启前判断以及载入前后对比数据的接口，实现ActInterceptor接口

可参考com.kayak.web.act.interceptor. UserEditInterceptor类

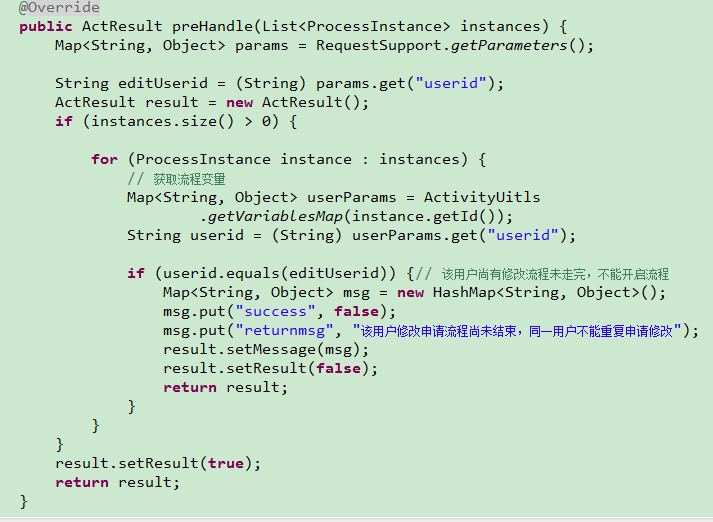
实现后，配置到工作流文件下，如下图



ActInterceptor接口有两个接口需要实现，下面针对这两个接口做详细的说明。

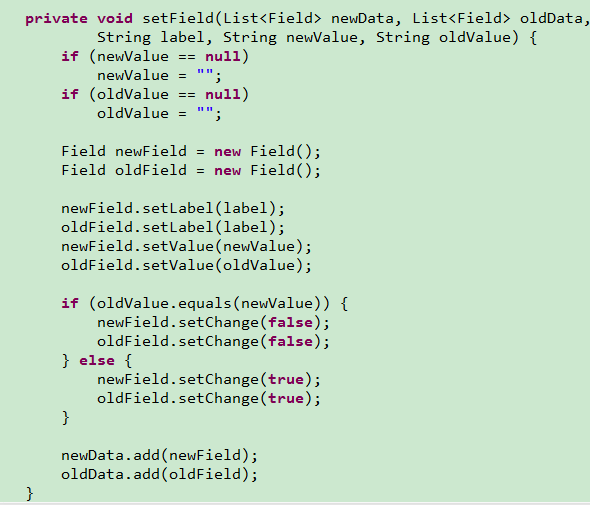
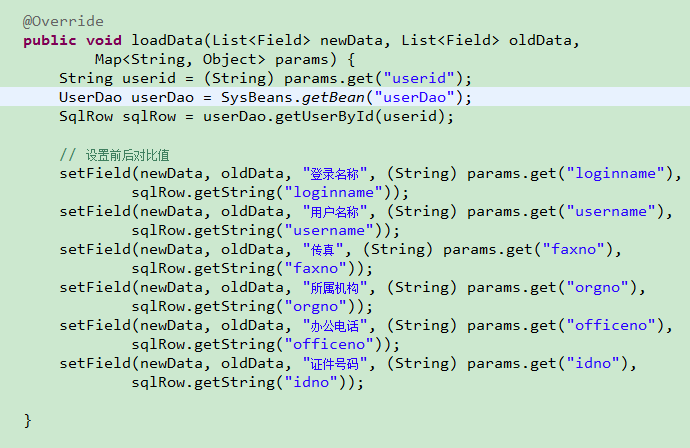
**preHandle接口：**

此接口表示流程事前校验接口，在流程开启之前，会执行该接口进行判断，然后根据判断结果选择是否开启流程。所以项目组在实现此接口时，需要根据具体的业务规则进行判断。如下案例，修改用户时，同一个用户不能重复申请修改



**loadData接口：**

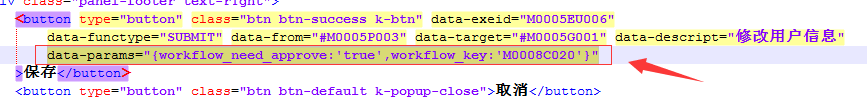
loadData接口在审批时数据展示会被调用，流程引擎根据调用的接口展示数据。接口有三个参入传入，List<Field> newData, List<Field> oldData, Map<String, Object> params。newData表示展示申请更新到数据库的数据，oldData表示当前数据库的数据，params表示用户提交的参数。项目组需要根据具体的业务进行实现，比如实现数据对比（在修改时需要数据对比），把对比后的数据分布插入newData与oldData。如果是新增操作，则不需要理会oldData。如修改用户时候的数据展示。



1. 配置页面指定按钮开启流程

工作流的控制是定位到页面按钮的，所以在需要使用工作流控制的按钮上配置相应的属性。

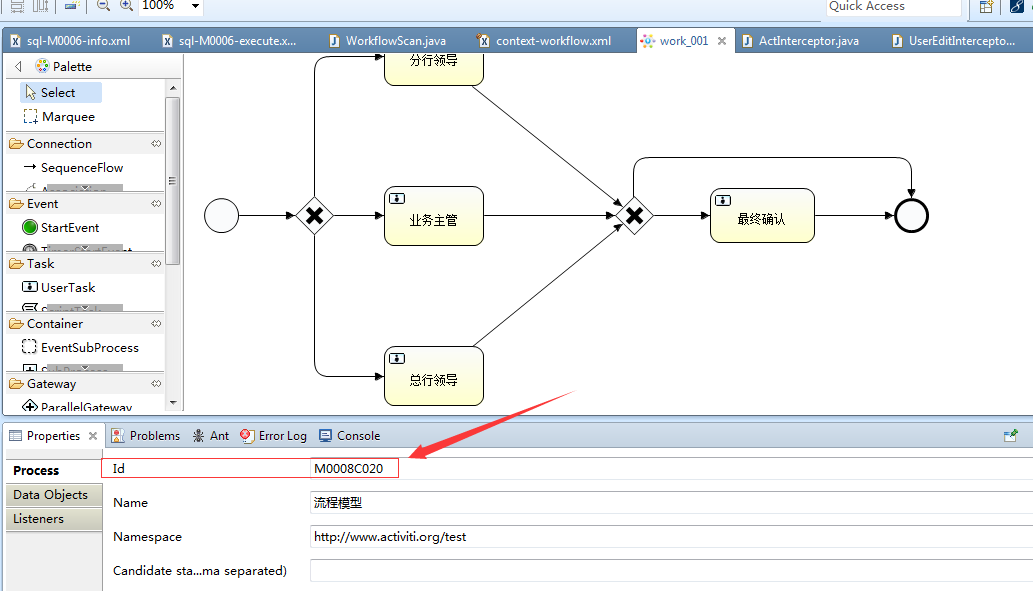
如下图



data-params="{workflow\_need\_approve:'true',workflow\_key:'M0008C020',workflow\_descript: '修改用户流程'}"

workflow\_need\_approve属性表示是否使用需要使用工作流

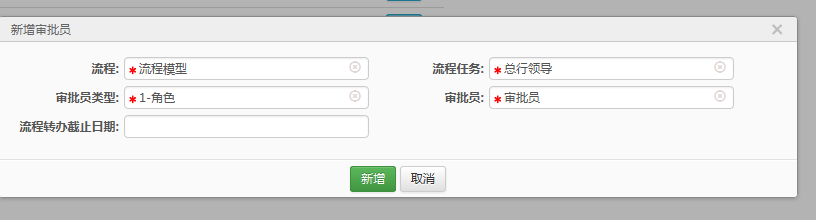
workflow\_key属性指定开启的具体的工作流id值



配置完成后，重启Tomcat，系统会自动扫描工作流相关文件，并记录到数据库中。

1. 配置审批员

工作流审批需要配置具体的角色类审批工作流具体的任务



在具体的任务下配置具体的审批人的角色，在审批工作流的时候，相应的角色可以审批该任务是否通过。

自定义工作流

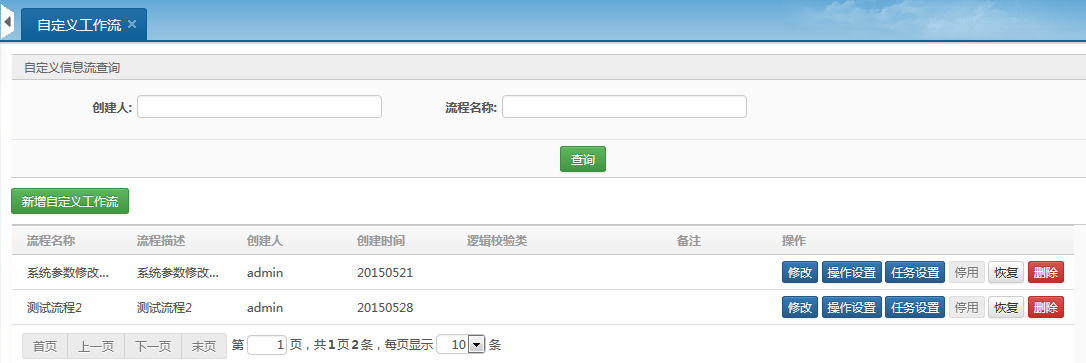
1. 使用方法

自定义工作流依赖权限控制功能，故在需要进行审批的流程中，需要先将改按钮纳入权限控制（相关资料请参考2.1）。

自定义工作流使用方法如下：

1. **新建自定义流程**

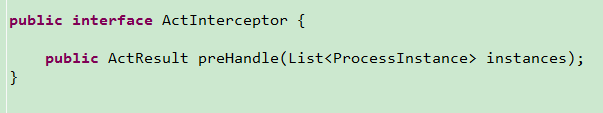
在自定义工作流模块新增自定义流程，如下图



新增窗口如下



**逻辑校验类：**如果流程需要做事前校验，可实现逻辑校验接口，然后把实现的类名填入（如果没有配置，可不填）。校验类的接口如下

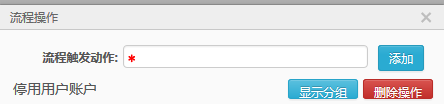


如果配置了逻辑校验类，在流程开启之前，会调用逻辑校验类进行事前校验，根据返回结果判断是否需要进入工作流。

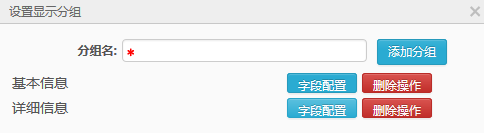
**停用：**停用改工作流，工作流停用后，新的操作都不会进入工作流审批，但是不会影响到未审批完成的工作流。

**恢复：**将停用的工作流恢复，恢复后，新的操作会进入工作流审批流程中。

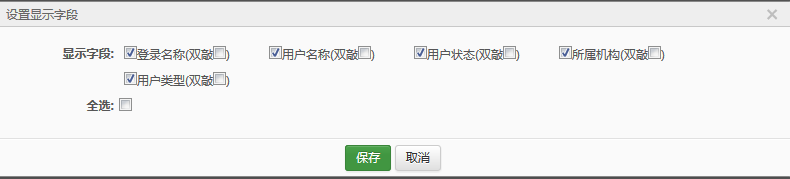
**操作设置：**操作设置可指定触发审批流程的按钮，以及审批中显示的字段等信息，如下图



添加流程操作后（同一个流程可同时指定多个触发的按钮，但是显示以及分组等需要需要单独配置），可配置分组信息，点击【显示分组】进入分组配置界面，如下图：



如上图创建了基本信息和详细信息的显示分组，然后把要显示的字段归纳到分组里面，点击【字段配置】配置分组的详细字段，如下图

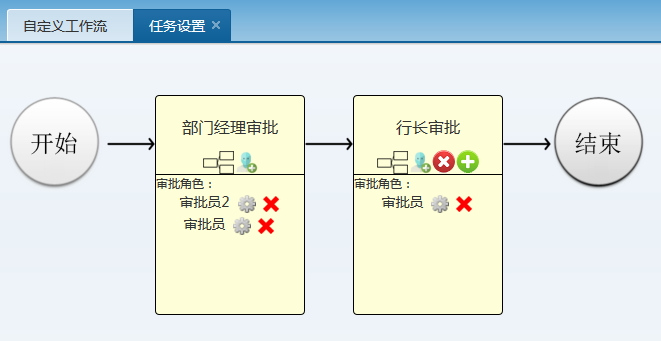
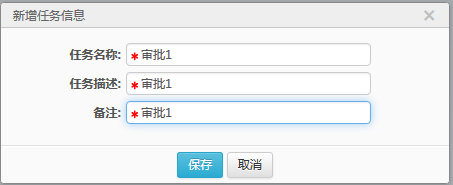


勾选该分组的显示字段，如果该字段需要双敲，可勾选相应的双敲按钮。

**任务设置：**创建好工作流的基本信息后，接着创建工作流的任务流向，点击【任务设置】进入工作流任务配置界面，如下图



在新创建的工作流中，无任何任务，所以需点击【开始】图标，添加第一个任务



添加后如图所示，会出现添加后的任务流程，以下分别解释每个按钮的含义：

：设置任务审批条件，任务审批条件是指，流程开启时，流程引擎会自动判断该任务是否存在审批条件，如果有审批条件，这根据判断是否满足审批条件，如果满足，则该任务需要进行审批，否则会在审批中去除在审批列表中。审批的条件可根据提交的参数进行判断。任务审批条件配置如下图：



**条件连接符：**表示条件直接的连接条件，第一条件连接符为无，连接符可选范围（无，and，or，not）

**参数：**配置条件判断的参数，该参数是页面表单提交的参数，其中用户属性参数是指提交用户的属性值。

**参数类型：**参数的变量类型，可选范围（数字，字符串）

**逻辑符：**逻辑判断的符号

**判断值：**与参数值进行判断的值。

：设置审批角色，设置任务的审批角色，并且设置每个角色的审批条件，审批条件与任务审批条件类似，只是作用于角色，只有满足条件的该角色才能审批。



：删除任务。



：添加下一任务。

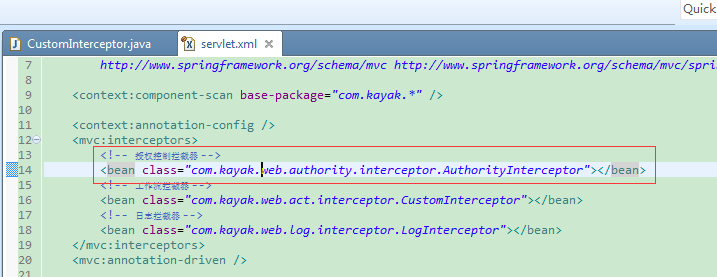


**注意：如果有多个条件判断时，第一个条件的逻辑连接符必须是无，以后的条件根据情况（and，or，not）选择。**

授权功能

1. 涉及的文件
2. \WebRoot\WEB-INF\conf\spring\sys\servlet.xml
3. \WebRoot\WEB-INF\conf\sql\sys\sql-auth-info.xml
4. \WebRoot\kstrap\k-util.js
5. \WebRoot\kstrap\include.jsp
6. \WebRoot\page\M0\M0010.jsp
7. \WebRoot\page\M0\M0010SP01.jsp
8. \WebRoot\WEB-INF\conf\sql\M0\sql-M0010-execute.xml
9. \WebRoot\WEB-INF\conf\sql\M0\sql-M0010-info.xml
10. 使用方法

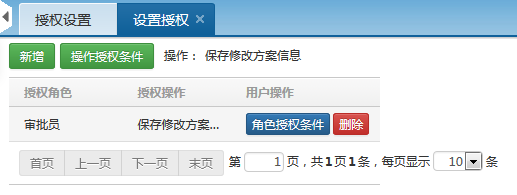
在使用之前，需要设置授权的操作的按钮需已纳入权限控制，权限控制相信息请查看2.1模块。并且授权的拦截器以及配置，如下图



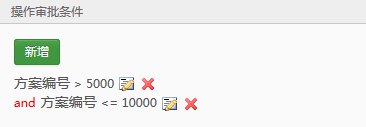
进入授权设置模块，点击对应的按钮的【设置】按钮进入授权设置界面



授权设置界面如下图



**操作授权条件：**指触发授权操作的条件，如果没有设置操作授权条件，则默认改按钮触发的操作都需要授权；如设置了授权条件，这在授权之前，会调用授权条件，如果满足条件，才需要进行授权，不满足则跳过授权直接执行业务操作。配置界面如下



**条件连接符：**表示条件直接的连接条件，第一条件连接符为无，连接符可选范围（无，and，or，not）

**参数：**配置条件判断的参数，该参数是页面表单提交的参数。

**参数类型：**参数的变量类型，可选范围（数字，字符串）

**逻辑符：**逻辑判断的符号

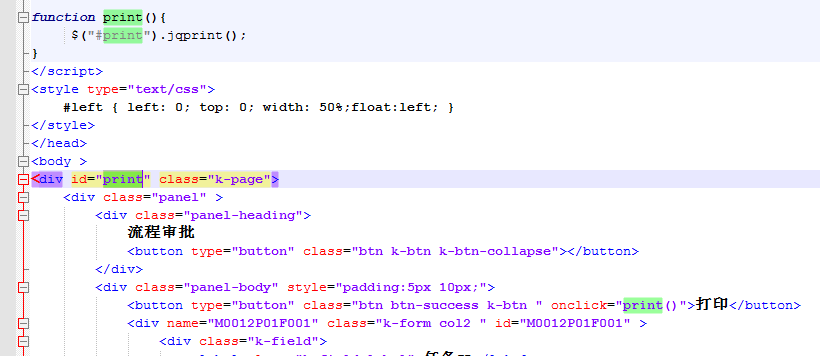
**判断值：**与参数值进行判断的值。

**角色授权条件：**角色授权条件是在操作授权条件的基础之上的。若设置了角色授权条件，这在该角色进行授权的时候，会进一步判断角色授权条件，只有满足条件的角色，才允许对该操作授权。若不设置角色授权条件，则只要有该操作的授权，该角色都有权限去授权该操作。

打印功能

1. 涉及的文件
2. \WebRoot\js\binary\jquery.jqprint-0.3.js
3. \WebRoot\js\binary\jquery-migrate-1.1.0.js
4. \WebRoot\kstrap\include.jsp
5. 使用方法

本打印方式使用js进行打印，使用时，只需要使用jqprint()函数，即可打印，如下图所示



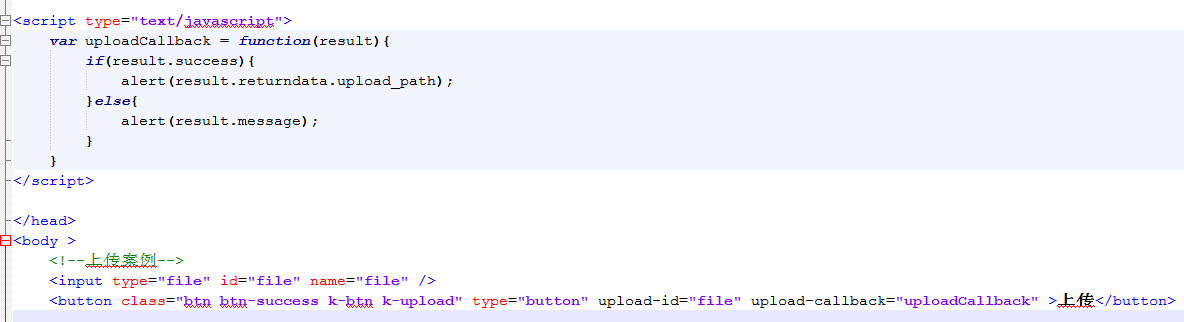
通过js获取要打印的对象，可以是div或者其他标签，然后使用jqprint方法，可见标签体内的内容打印出来。

注意：当前不支持ireport自定义打印，后期版本会增加

上传与下载功能

1. 使用方法

**上传案例**



定义两个标签，一个为Input，类型为file，并且定义一个id属性

另外一个为button标签（任何标签都行），在class属性增加k-upload，并且定义如下属性

**upload-id：**要上传的文件窗口的id值

**upload-callback：**定义上传成功后的回调函数，上传的结果会传入改函数。

返回结果如下：

{

"success":true,

"returnmsg":"上传成功",

"returndata":{"upload\_path":"upload/20150520/1524209158.ini",

"upload\_name":"application.ini"

}

}

**upload\_path**为上传成功后上传的文件在服务器的路径

**upload\_name**为上传的文件名

**下载案例**



下载必须是a标签（如果需要通过事件触发下，可使用window.location.href的方式），下载的链接为base/comn-download.action，传入两个参数：

**download\_path：**下载的文件在服务器上的路径

**download\_name：**下载的文件名

权限拦截器

1. 使用方法

平台默认情况下，没对exeid以及菜单等做拦截控制，这对于懂技术的人而已，无疑是一种安全漏洞，可以通过修改exeid参数执行本身没有权限执行的exeid。菜单也同样。所以平台增加的权限拦截器，使用方式很简单，只需要在web.xml配置以下信息即可。

<!-- 权限控制过滤器 -->

<filter>

<filter-name>accessFilter</filter-name>

<filter-class>com.kayak.web.base.filter.AccessFilter</filter-class>

<init-param><!-- 配置不需要过滤的页面 -->

<param-name>unfilter-page</param-name>

<param-value>/login.jsp,/index.jsp,/login.json,/login2.jsp,/logout.action,/loginCm.action,/validatecode.jsp,/version.jsp,/base/bizware-call-me.json,/genpass.jsp</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>accessFilter</filter-name>

<url-pattern>\*.jsp</url-pattern>

<url-pattern>\*.json</url-pattern>

<url-pattern>\*.action</url-pattern>

</filter-mapping>

1. 特殊的访问路径

平台权限拦截控制依赖的数据表为：sys\_button\_access与sys\_sub\_menu。系统在启动扫描的时候，会在JSP页面将按钮id与exeid的对应关系存入sys\_button\_access表，所以，如果是开发人员直接使用ajax的方式访问的链接，系统不会扫描到，这种情况需要开发人员手动在sys\_button\_access表加入相应的按钮id与exeid（url）的关系。对于菜单访问，默认系统启动扫描时会自动减菜单URL与菜单ID对应起来。如果是开发人员自己特殊定义的链接，需要开发人员手动在sys\_sub\_menu插入链接与菜单ID的关系。

1. 转审批操作使用方法

转审批操作的使用方法如下：

**第一、转审批授权流程显示以及转审批设置页面的跳转**

转审批授权显示页面如下图：



显示内容为当前用户可审批的流程。

转审批设置页面的跳转需要用户再次输入密码验证，以防用户不再的情况下其他人操作

具体显示如下图：



密码为登录密码。

1. **新增转审批设置**

转审批设置显示页面如下图：



用户可以选择转授权的其他用户，并且可以输入转授权开始日期和结束日期（也可以不输），完成可以直接点击启用。

新建转审批设置如下图：



任务为当前流程下用户可审批的任务。

代理用户为所有用户。

时间限制可启用、可不启用 。

状态有禁用与启用。

**停用：**

用户不需要转授权可以直接停止。

**恢复：**

将会恢复代理用户的审批权限。

**修改：**

同新增页面。

1. **代理用户的操作**

转授权启用后，被授权用户可以审批授权人的各种交易，

审批成功后在历史任务中能记录审批人+（转授权）如下图第一条。

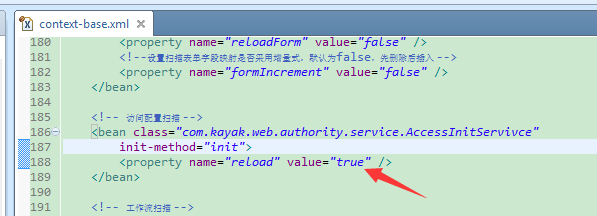


1. 请求权限控制

如果在sys.param.properties配置SYS\_ACCESS\_AUTH = true，则进行强权限认证，所有的请求都必须进行权限认证。权限认证的逻辑更新配置的菜单及按钮权限配置。

默认项目启动会自动扫描JSP页面，将按钮与exeid的对应关系扫描配置到xml中，xml文件名规则，模块ID-菜单ID-access.xml，生成后将生成的配置文件从tomcat复制到项目中，完善配置文件，由于扫描jsp页面的获取的对应关系并不完全，必须每个功能，每个exeid进行核对，将没有扫描进去的exeid配置上去。

扫描配置开关：



案例：

<access-id>M0001U03</access-id>

<access-descript>停用机构信息</access-descript>

<access-exeid>M0001EUPAUSE</access-exeid><!--对应多个exeid可以用,分隔-->

</access-node>

bizware权限控制

在WEB-INF/conf/bizware\_access建立按钮ID与bizware参数的对应关系

案例：

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

<access-config>

<access-node>

<access-id>M0002U01</access-id>

<access-params>{"param1":1,"param2":2}</access-params>

</access-node>

<access-node>

<access-id>M0002U02</access-id>

<access-params>{"param3":3,"param4":4}</access-params>

</access-node>

</access-config>

access-id为按钮ID，access-params为可确定的授权参数集合

比如按钮1的按钮ID为button1，bizware操作的交易码为 business\_code: B001

配置如下：

<access-node>

<access-id>button1</access-id>

<access-params>{"business\_code":"B001"}</access-params>

</access-node>

支持多个参数确定功能。