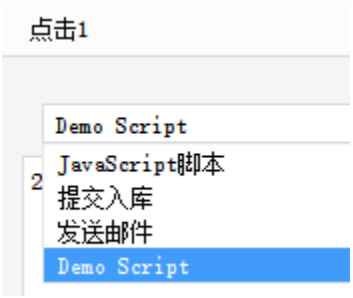
com.fr.design.fun.JavaScriptActionProvider

-
-
-
-
-
-
-
-
-

JavaScriptActionProvider

/WEB



JS

JavaScriptActionProvider.java

```
package com.fr.design.fun;

import com.fr.design.beans.FurtherBasicBeanPane;
import com.fr.design.javascript.JavaScriptActionPane;
import com.fr.design.mainframe.JTemplate;
import com.fr.js.JavaScript;
import com.fr.stable.fun.mark.Mutable;

/**
 *
 */
public interface JavaScriptActionProvider extends Mutable{

    String XML_TAG = "JavaScriptActionProvider";

    int CURRENT_LEVEL = 1;

    /**
     *
     */
    FurtherBasicBeanPane<? extends JavaScript> getJavaScriptActionPane();

    /**
     *
     * @see com.fr.design.mainframe.JWorkBook
     * @see com.fr.design.mainframe.JForm
     */
    boolean accept(JTemplate template);

    @Deprecated
    FurtherBasicBeanPane<? extends JavaScript> getJavaScriptActionPane(JavaScriptActionPane pane);

    @Deprecated
    boolean isSupportType();
}
```

FR	8.0		
FR	9.0		
FR	10.0		

plugin.xml

```
<extra-designer>
        <JavaScriptActionProvider class="your class name"/>
</extra-designer>
```

JSJavaScriptActionPaneListenerEditPaneJavaScriptActionProvider

Scriptcptfrm

4

boolean isSupportType();FurtherBasicBeanPane<? extends JavaScript> getJavaScriptActionPane(JavaScriptActionPane pane);

FurtherBasicBeanPane JavaScript

FurtherBasicBeanPane.java

```
package com.fr.design.beans;

import com.fr.common.annotations.Open;
import com.fr.stable.StringUtils;

@Open
public abstract class FurtherBasicBeanPane<T> extends BasicBeanPane<T> {
    /**
     *
     *
     * @param ob
     * @return
     */
    public abstract boolean accept(Object ob);

    /**
     * title
     *
     * @return
     */
    @Override
    public String title4PopupWindow() {
        return StringUtils.EMPTY;
    }

    /**
     *
     */
    public abstract void reset();

}
```

JavaScript

```
package com.fr.js;

import com.fr.decision.authority.base.constant.DeviceType;
import com.fr.json.JSONException;
import com.fr.json.JSONObject;
import com.fr.script.Calculator;
import com.fr.stable.ColumnRow;
import com.fr.stable.ParameterProvider;
import com.fr.stable.script.CalculatorKey;
import com.fr.stable.script.CalculatorProvider;
import com.fr.stable.script.ExTool;
import com.fr.stable.web.Repository;
import com.fr.stable.xml.XMLable;

/**
 * Javascript
 */
public interface JavaScript extends XMLable {

    CalculatorKey RECALCULATE_TAG = CalculatorKey.createKey("shouldRecalculate");

    DeviceType ALL_DEVICE = new DeviceType().supportAll();

    /**
```

```
 * XML
 */
String XML_TAG = "JavaScript";

/**
 * javascript
 *
 * @param repo
 * @return Javascript
 */
String createJS(Repository repo);

/**
 * JS
 *
 * @param repo
 * @param content
 * @return JS
 */
JavaScript append(Repository repo, String content);

/**
 * JS
 *
 * @param repo
 * @param content
 * @return JS
 */
JavaScript prepend(Repository repo, String content);

/**
 * JSON
 *
 * @param repo
 * @return JSON
 */
JSONObject createJSONObject(Repository repo) throws JSONException;

/**
 * javascript
 *
 * @param map
 */
void addParameterMap(java.util.Map map);

/**
 * Javascript
 *
 * @return Javascript
 */
ParameterProvider[] getParameters();

/**
 * Javascript
 *
 * @param ps
 */
void setParameters(ParameterProvider[] ps);

/**
 *
 *
 * @return
 */
ParameterProvider[] getParameterizedConfig();

/**
 *
 *
 * @param calculator
 * @param exTool
```

```
     * @param currentCr
     */
    void analyzeCorrelative(CalculatorProvider calculator, ExTool exTool, ColumnRow currentCr);

    /**
     *
     *
     * @return truefalse
     */
    boolean shouldRecalculate();

    /**
     *
     *
     * @param recalculate true
     */
    void setShouldRecalculate(boolean recalculate);

    /**
     *
     *
     * @param title
     */
    void setLinkTitle(String title);

    /**
     * JavaScript
     */
    void renderContent(Calculator calculator);

    /**
     *
     *
     * @return
     */
    DeviceType getDeviceType();
}
```

AbstractJavaScript.java

```
package com.fr.js;

import com.fr.base.BaseXMLUtils;
import com.fr.base.Parameter;
import com.fr.decision.authority.base.constant.DeviceType;
import com.fr.general.ComparatorUtils;
import com.fr.json.JSONException;
import com.fr.json.JSONObject;
import com.fr.script.Calculator;
import com.fr.stable.ArrayUtils;
import com.fr.stable.ColumnRow;
import com.fr.stable.FormulaProvider;
import com.fr.stable.ParameterProvider;
import com.fr.stable.script.CalculatorKey;
import com.fr.stable.script.CalculatorProvider;
import com.fr.stable.script.ExTool;
import com.fr.stable.web.Repository;
import com.fr.stable.xml.StableXMLUtils;
import com.fr.stable.xml.XMLPrintWriter;
import com.fr.stable.xml.XMLableReader;

import java.util.HashMap;
import java.util.Map;

/**
 * JavaScript
 */
public abstract class AbstractJavaScript implements JavaScript {
    private static final long serialVersionUID = -3629245217096045333L;
```

```
private static final String JS_VERSION = "10";

/**
 * @deprecated use {@link JavaScript#RECALCULATE_TAG} instead
 */
@Deprecated
public static CalculatorKey RECALCULATE_TAG = JavaScript.RECALCULATE_TAG;

/**
 *
 */
public static final String CALLBACK = "callback";

/**
 *
 */
public static final String FEEDBACKMAP = "feedbackMap";

/**
 * javascript
 * protectedgetParameters()
 */
@Deprecated
protected ParameterProvider[] parameters;
/**
 *  callback
 *  map
 *
 */
protected Map<Object, Object> paraMap = new HashMap<Object, Object>();

private boolean recalculate;

private String itemName = "";//js

public String getItemName() {
    return this.itemName;
}

public void setItemName(String itemName) {
    this.itemName = itemName;
}

/**
 * javascript
 *
 * @return
 */
public ParameterProvider[] getParameters() {
    return this.parameters == null ? new ParameterProvider[0] : this.parameters;
}

/**
 * javascript
 *
 * @param parameters
 */
public void setParameters(ParameterProvider[] parameters) {
    this.parameters = parameters;
}

/**
 *
 *
 * @return truefalse
 */
public boolean shouldRecalculate() {
    return recalculate;
}
```

```
/**
 *
 *
 * @param recalculate true
 */
public void setShouldRecalculate(boolean recalculate) {
    this.recalculate = recalculate;
}

/**
 * javascript
 *
 * @param repo Session
 * @return javascript
 */
public String createJS(Repository repo) {
    //b:jsargumentsarguments
    //wei:js"use strict",arguments,arguments.
    return "var as=arguments; return FR.tc(function(){" + actionJS(repo) + "}, this, as)";
}

/**
 * JSON
 *
 * @param repo
 * @return JSON
 */
public JSONObject createJSONObject(Repository repo) throws JSONException {
    JSONObject res = JSONObject.create();
    res.put("version", JS_VERSION);
    res.put("data", createJS(repo));
    return res;
}

protected abstract String actionJS(Repository repo);

@Override
public JavaScript append(Repository repo, String content) {
    return this;
}

@Override
public JavaScript prepend(Repository repo, String content) {
    return this;
}

/**
 * javascript
 *
 * @param map
 */
public void addParameterMap(java.util.Map map) {
    paraMap.putAll(map);
}

public void renderContent(Calculator calculator) {

}

@Override
public DeviceType getDeviceType() {

    return ALL_DEVICE;
}

/**
 *
 *
 * @param calculator
 * @param exTool
 * @param currentCr
```

```
 */
public void analyzeCorrelative(CalculatorProvider calculator, ExTool exTool, ColumnRow currentCr) {
    ParameterProvider[] pps = getParameterizedConfig();
    for (int p = 0, len = pps.length; p < len; p++) {
        ParameterProvider provider = pps[p];
        Object pValue = provider.getValue();
        //
        if (pValue instanceof FormulaProvider) {
            exTool.setCreateRelation(true);
            calculator.exStatement(currentCr, ((FormulaProvider) pValue).getPureContent());
            exTool.setCreateRelation(false);
        }
    }
}

/**
 *
 *
 * @param title
 */
public void setLinkTitle(String title) {

}

public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (!(obj instanceof AbstractJavaScript)) {
        return false;
    }

    return ComparatorUtils.equals(this.parameters, ((AbstractJavaScript) obj).parameters);
}

public void readXML(XMLableReader reader) {
    if (reader.isChildNode()) {
        if (Parameter.ARRAY_XML_TAG.equals(reader.getTagName())) { //parameters
            Parameter[] newParameters = BaseXMLUtils.readParameters(reader);
            this.setParameters(newParameters);
        }
    }
}

public void writeXML(XMLPrintWriter writer) {
    // Parameters
    StableXMLUtils.writeParameters(writer, this.getParameters());
}

public Object clone() throws CloneNotSupportedException {
    AbstractJavaScript cloned = (AbstractJavaScript) super.clone();
    if (parameters != null) {
        cloned.parameters = new Parameter[this.parameters.length];
        for (int i = 0; i < this.parameters.length; i++) {
            cloned.parameters[i] = (Parameter) this.parameters[i].clone();
        }
    }

    return cloned;
}

@Override
public ParameterProvider[] getParameterizedConfig() {
    ParameterProvider[] parameters = getParameters();
    ParameterProvider[] extra = getExtraParameterizedConfig();

    return ArrayUtils.addAll(parameters, extra);
}

/**
 *
```

```
     *
     * @return
     */
    public ParameterProvider[] getExtraParameterizedConfig() {
        return new ParameterProvider[0];
    }
}
```

FurtherBasicBeanPanetitle4PopupWindow

JavaScriptAbstractJavaScriptreadXML/writeXML/clone

demodemo-java-script-action-provider

web

JSCSS

demo-file-submit-oss