com.fr.design.fun.HyperlinkProvider

-
-
-
-
-
-
-
-
-

1.URLURL

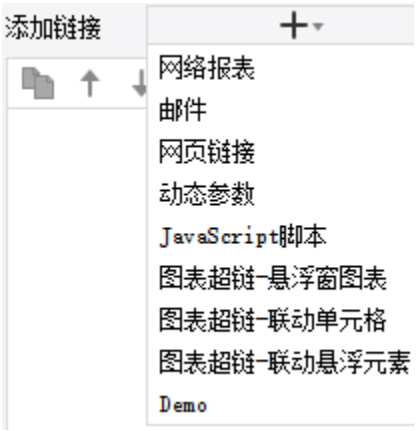
2.JSJavascript

UIKEY/IDNAME

FR.("","","","","")

APIJSJSON

HyperlinkProvider



HyperlinkProvider.java

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

import com.fr.design.beans.BasicBeanPane;
import com.fr.design.gui.controlpane.NameableCreator;
import com.fr.js.Hyperlink;
import com.fr.stable.fun.mark.Mutable;

/**
 * Created by zack on 2016/1/20.
 */
public interface HyperlinkProvider<T extends Hyperlink> extends Mutable {
    String XML_TAG = "HyperlinkProvider";

    int CURRENT_LEVEL = 2;


    /**
     *
     * @see HyperlinkProvider#text()
     * @see HyperlinkProvider#target()
     * @see HyperlinkProvider#appearance()
     *
     * @return
     */
    NameableCreator createHyperlinkCreator();

    /**
     *
     * @return
     */
    String text();

    /**
     *
     * @return
     */
    Class<T> target();

    /**
     *
     * @return
     */
    Class<? extends BasicBeanPane<T>> appearance();
}
```

Hyperlink.java

```
package com.fr.js;

import com.fr.base.BaseFormula;
import com.fr.base.Utils;
import com.fr.general.ComparatorUtils;
import com.fr.general.GeneralContext;
import com.fr.general.web.ParameterConstants;
import com.fr.json.JSONException;
import com.fr.json.JSONObject;
import com.fr.stable.AssistUtils;
import com.fr.stable.BaseSessionFilterParameterManager;
import com.fr.stable.FilterParameterManager;
import com.fr.stable.ParameterProvider;
import com.fr.stable.StringUtils;
import com.fr.stable.web.Repository;
import com.fr.stable.xml.XMLPrintWriter;
import com.fr.stable.xml.XMLableReader;
```

```
import java.util.Date;
import java.util.Map;

/**
 *
 */
public abstract class Hyperlink extends AbstractJavaScript {

    //,
    private static final String BLANK_FRAME = "_blank";

    private static final int MAX_PARA_NAME_LENGTH = 20;

    private static final String WIDTH = "width";

    private static final String HEIGHT = "height";

    protected static final String CHART_SOURCE_NAME = "__CHARTSOURCENAME__";

    protected static final String CHART_SIZE = "__CHARTSIZE__";

    private String targetFrame = BLANK_FRAME;

    private int width = 0;

    private int height = 0;

    //(1,1...)iframe,namejavascrpt,
    // name{data:FR.tc....,name:***},dataname,data
    private String title;

    /**
     *
     *   json
     *
     * @return  json
     */
    public JSONObject features4NewWindow(Repository repository) {

        JSONObject res = JSONObject.create();
        res.put("width", Math.max(this.width, 0));
        res.put("height", Math.max(this.height, 0));
        return res;
    }

    /**
     * @return  string
     * @see Hyperlink#features4NewWindow(Repository)
     * @deprecated
     */
    @Deprecated
    public String features4NewWindow() {

        StringBuilder sb = new StringBuilder();

        if (width > 0) {
            sb.append(WIDTH).append('=').append(width).append(',');
        }
        if (height > 0) {
            sb.append(HEIGHT).append('=').append(height).append(',');
        }

        return sb.toString();
    }

    /**
     *
     * JSON
     *
     * @param repo
     * @param jo   JSON
```

```
     * @throws JSONException e
     */
    public void putExtendParameters(Repository repo, JSONObject jo) throws JSONException {

        Map paraMap = repo.getReportParameterMap();
        putExtendParameters(paraMap, jo);
    }

    /**
     *
     *
     * @param paraMap
     * @param jo      jo
     * @throws JSONException e
     */
    protected void putExtendParameters(Map paraMap, JSONObject jo) throws JSONException {

        removeUnusedPara(paraMap);

        //
        ParameterProvider[] linkParas = this.getParameters();

        for (Object o : paraMap.entrySet()) {
            Map.Entry entry = (Map.Entry) o;
            String key = (String) entry.getKey();
            Object value = entry.getValue();

            if (key.length() > MAX_PARA_NAME_LENGTH) {
                //, jsf, ,
                continue;
            }

            boolean hasLinkPara = false;
            for (ParameterProvider linkPara : linkParas) {
                if (key.equalsIgnoreCase(linkPara.getName())) {
                    hasLinkPara = true;
                }
            }

            //, p1,, p1
            if (!hasLinkPara) {
                //   FArray
                jo.put(key, evalParameterValue(value));
            }
        }
    }

    private void removeUnusedPara(Map paraMap) {

        String[] extendParameterFilters = BaseSessionFilterParameterManager.getFilterParameters();
        for (String extendParameterFilter : extendParameterFilters) {
            paraMap.remove(extendParameterFilter);
        }

        for (int i = 0; i < ParameterConstants.ALL.length; i++) {
            paraMap.remove(ParameterConstants.ALL[i]);
        }

        for (String s : FilterParameterManager.getInstance().getEmbedded()) {
            paraMap.remove(s);
        }


        paraMap.remove(CHART_SOURCE_NAME);
        paraMap.remove(CHART_SIZE);
    }

    /**
     *
     */
    public String getTargetFrame() {
```

```
    return targetFrame;
}

/**
 *
 *
 * @param targetFrame
 */
public void setTargetFrame(String targetFrame) {

    this.targetFrame = targetFrame;
}

/**
 *
 *
 * @return
 */
public int getWidth() {

    return width;
}

/**
 *
 *
 * @param width
 */
public void setWidth(int width) {

    this.width = width;
}

/**
 *
 *
 * @return
 */
public int getHeight() {

    return height;
}

/**
 *
 *
 * @param height
 */
public void setHeight(int height) {

    this.height = height;
}

/**
 *
 *
 * @return
 */
public String getTitle() {

    return title;
}

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

```
    this.title = title;
}

/**
 * XML
 *
 * @param reader the element.
 */
@Override
public void readXML(XMLableReader reader) {

    super.readXML(reader);

    if (reader.isChildNode()) {
        if (ComparatorUtils.equals(reader.getTagName(), "TargetFrame")) {
            String tmpVal;
            if (StringUtils.isNotBlank((tmpVal = reader.getElementValue()))) {
                Hyperlink.this.setTargetFrame(tmpVal);
            }
        } else if (ComparatorUtils.equals(reader.getTagName(), "Features")) {
            String tmpVal;
            Hyperlink.this.setWidth(reader.getAttrAsInt(WIDTH, 0));
            Hyperlink.this.setHeight(reader.getAttrAsInt(HEIGHT, 0));

            if (StringUtils.isNotBlank((tmpVal = reader.getElementValue()))) {
                // alex:features(2010.7.14)
                String[] feArray = tmpVal.split(",");
                for (int i = 0; i < feArray.length; i++) {
                    String fe = feArray[i];
                    String[] nameAndValue = fe.split("=");
                    if (nameAndValue.length != 2) {
                        continue;
                    }
                    String feName = nameAndValue[0];
                    String feValue = nameAndValue[1];

                    if (WIDTH.equalsIgnoreCase(feName)) {
                        Hyperlink.this.setWidth(Utils.string2Number(feValue).intValue());
                    } else if (HEIGHT.equalsIgnoreCase(feName)) {
                        Hyperlink.this.setHeight(Utils.string2Number(feValue).intValue());
                    }
                }
            }
        }
    }
}

/**
 * json
 */
protected void para2JSON(JSONObject jo) throws JSONException {

    for (ParameterProvider parameter : getParameters()) {
        String name = parameter.getName();
        Object value = parameter.getValue();
        if (value == null || StringUtils.isEmpty(name)) {
            continue;
        }
        jo.put(name, evalParameterValue(value));
    }
}

/**
 *
 *
 */
@Deprecated
private Object evalParameterValue(Object o) {

    if (o instanceof BaseFormula) {
```

```
        //
        //juJSONFormulaSerializer
        o = ((BaseFormula) o).getResult();
        return evalParameterValue(o);
    } else if (o instanceof Date) {
        return GeneralContext.getDefaultValues().getDateTimeFormat().format(o);
    }
    return o;
}

@Override
public JSONObject createJSONObject(Repository repo) throws JSONException {

    JSONObject jo = super.createJSONObject(repo);
    jo.put("target", this.getTargetFrame());
    jo.put("parameters", createPara(repo));
    jo.put("type", getHyperlinkType());

    return jo;
}

/**
 *
 *
 * @return
 */
protected String getHyperlinkType() {
    // , abstract.
    return "none";
}

/**
 *
 *
 * @param repo
 * @return
 */
protected JSONObject createPara(Repository repo) throws JSONException {

    return JSONObject.create();
}

/**
 * XML
 *
 * @param writer the PrintWriter.
 */
@Override
public void writeXML(XMLPrintWriter writer) {

    super.writeXML(writer);

    if (StringUtils.isNotBlank(this.getTargetFrame())) {
        writer.startTAG("TargetFrame").textNode(this.getTargetFrame()).end();
    }

    writer.startTAG("Features");
    if (width > 0) {
        writer.attr(WIDTH, width);
    }
    if (height > 0) {
        writer.attr(HEIGHT, height);
    }
    writer.end();
}

protected boolean isPost() {

    return false;
}
```

```
    public boolean equals(Object obj) {

        return super.equals(obj)
            && obj instanceof Hyperlink
            && AssistUtils.equals(height, ((Hyperlink) obj).height)
            && AssistUtils.equals(width, ((Hyperlink) obj).width)
            && AssistUtils.equals(targetFrame, ((Hyperlink) obj).targetFrame);
    }
}
```

FR	8.0		
FR	9.0		
FR	10.0		

plugin.xml

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

Set<HyperlinkProvider> providers = ExtraDesignClassManager.getInstance().getArray(HyperlinkProvider.XML_TAG);

()HyperlinkGroupPane

HyperLinkPaneChartInteractivePaneVanChartHyperLinkPane

createHyperlinkCreatorHyperlinkXMLHyperlinkProviderxml

HyperlinkProviderAbstractHyperlinkProvidercreateHyperlinkCreatorequalshashCode 3demo

createHyperlinkCreatorNameableCreatorHyperlinkProviderclass:

Hyperlink

BasicBeanPane

HyperlinkactionJSJSwriteXMLreadXMLcloneequals 4

BasicBeanPane

com.fr.design.fun.JavaScriptActionProvider

demo-hyperlink-provider

demo-file-submit-oss