

com.fr.stable.fun.WidgetModifyProvider

•
•
•
•
•
•
•
•
•

PDFWidgetModifyProvider

WidgetModifyProvider.java

```
package com.fr.stable.fun;

import com.fr.form.ui.Widget;
import com.fr.report.cell.CellElement;
import com.fr.stable.fun.mark.Mutable;
import com.fr.stable.script.CalculatorProvider;

import java.awt.image.BufferedImage;

/**
 *
 *
 */
public interface WidgetModifyProvider extends Mutable {

    int CURRENT_LEVEL = 1;

    String MARK_STRING = "WidgetModifyProvider";

    /**
     *
     *
     * @param widget
     * @return
     */
    boolean accept(Widget widget);

    /**
     *
     *
     * @param ce
     * @param pxWidth
     * @param pxHeight
     * @param calculator
     * @return
     */
    BufferedImage toImage(CellElement ce, int pxWidth, int pxHeight, CalculatorProvider calculator);
}
```

FR	10.0		

plugin.xml

```
<extra-report>
    <WidgetModifyProvider class="your class name"/>
</extra-report>
```

WB.java

```
package com.fr.write.cal;
...
final public class WB extends CalculatableResWorkSheet implements WriteECReport, ShrinkFitable, WBProvider {
    ...
    /**
     * ,
     * @param resultbook
     * @return
     * @date 2014-9-24-8:51:41
     */
    public ResultECReport processWidgetAsImage(ResultECReport resultbook) {
        if (!CustomConfigManager.getInstance().isPrintWidget()) {
            return resultbook;
        }

        WB cloneWB;
        try {
            cloneWB = (WB) resultbook.clone();
        } catch (CloneNotSupportedException e) {
            cloneWB = (WB) resultbook;
        }

        Iterator it = cloneWB.cellIterator();
        DynamicUnitList rowHeightList = this.getRowHeightList_DEC();
        DynamicUnitList columnWidthList = this.getColumnWidthList_DEC();

        while (it.hasNext()) {
            BCE_WRITE ce = (BCE_WRITE) it.next();
            Widget widget = (Widget) ce.getOptionalAttributes().get(WriteCellElementAttribute.WIDGET);
            processWidgetAsImage(widget, rowHeightList, columnWidthList, ce);
        }

        return cloneWB;
    }

    /**
     * ,
     * @param widget
     * @param rowHeightList list
     * @param columnWidthList list
     * @param ce
     * @date 2014-9-24-8:53:24
     */
    private void processWidgetAsImage(Widget widget, DynamicUnitList rowHeightList,
        DynamicUnitList columnWidthList, BCE_WRITE ce) {
```

```

        if (widget == null || !widget.isVisible()) {
            return;
        }

        int resolution = Constants.DEFAULT_WEBWRITE_AND_SCREEN_RESOLUTION;
        //
        int pxHeight = rowHeightList.getRangeValue(ce.getRow(), ce.getRow() + ce.getRowSpan()).toPixelI
(resolution);
        int pxWidth = columnWidthList.getRangeValue(ce.getColumn(), ce.getColumn() + ce.getColumnSpan()).toPixelI
(resolution);
        //
        BufferedImage image = drawWidget(widget, ce, pxHeight, pxWidth);

        if (image != null) {
            // , ,
            ce.setValue(image);
            //, presentvalue
            ce.setPresentValue(image);
        }
        ce.getOptionalAttributes().remove(WriteCellElementAttribute.WIDGET);
    }

    private BufferedImage drawWidget(Widget widget, CellElement ce, int pxHeight, int pxWidth) {
        ExtraReportClassManager extraReportClassManager = PluginModule.getAgent(PluginModule.ExtraReport);
        if (extraReportClassManager != null) {
            Set<WidgetModifyProvider> providers = extraReportClassManager.getArray(WidgetModifyProvider.
MARK_STRING);
            for (WidgetModifyProvider provider : providers) {
                if (provider.accept(widget)) {
                    return provider.toImage(ce, pxWidth, pxHeight, this.getCalculator());
                }
            }
        }
        return widget.toImage(ce.getValue(), pxWidth, pxHeight, ce.getStyle(), this.getCalculator());
    }
    ...
}

```

CustomConfigManager.getInstance().isPrintWidget() CustomConfig.printWidgettrue

WidgettoImageWidgetModifyProvider

[demo-demo-widget-modify-provider](#)