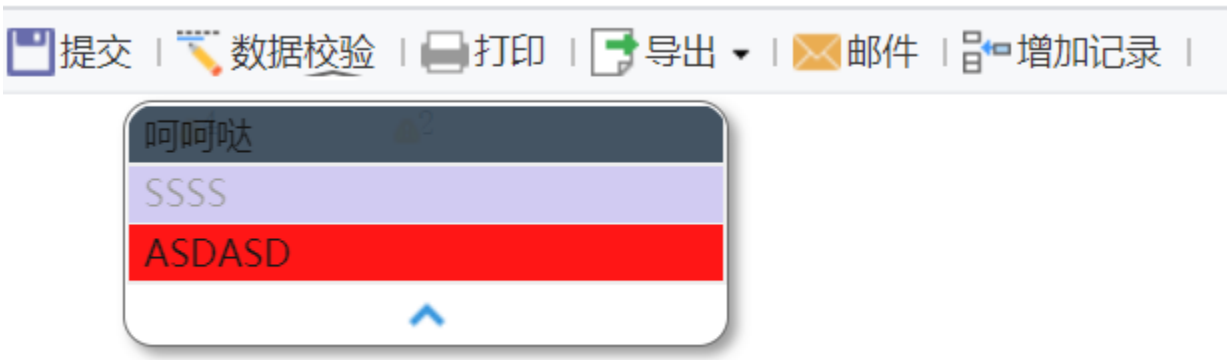
com.fr.design.fun.VerifyDefineProvider



VerifyDefineProvider.java

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

import com.fr.data.Verifier;
import com.fr.design.beans.BasicBeanPane;
import com.fr.stable.fun.mark.Mutable;

/**
 * Created by richie on 16/6/8.
 */
public interface VerifyDefineProvider extends Mutable {

    String MARK_STRING = "VerifyDefineProvider";

    int CURRENT_LEVEL = 1;

    /**
     *
     * @return
     */
    Class<? extends Verifier> classForVerifier();

    /**
     *
     * @return
     */
    Class<? extends BasicBeanPane> appearanceForVerifier();

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

    /**
     *
     * @return
     */
    String iconPath();
}
```

Verifier.java

```
package com.fr.data;

import com.fr.general.data.MOD_COLUMN_ROW;
import com.fr.json.JSONException;
import com.fr.json.JSONObject;
import com.fr.script.Calculator;
import com.fr.stable.Nameable;
import com.fr.stable.script.CalculatorKey;
import com.fr.stable.xml.XMLable;

/**
 *
 */
public interface Verifier extends XMLable, Nameable {

    CalculatorKey KEY = CalculatorKey.createKey(Verifier.class.getName());

    String XML_TAG = "TopVerifier";


    /**
     *
```

```
 *
 * @param item
 */
void addVerifyItem(VerifyItem item);

/**
 *
 *
 * @param index
 * @return
 */
VerifyItem getVerifyItem(int index);

/**
 *
 *
 * @return
 */
int getVerifyItemsCount();

/**
 *
 */
void clearVerifyItems();


/**
 *
 *
 * @param ca
 * @throws Exception
 */
void execute(Calculator ca) throws Exception;


/**
 *
 *
 * @return
 */
boolean isValid();

/**
 *
 *
 * @return
 */
boolean isBuiltInVerify();

/**
 * JSON
 *
 * @return JSON
 * @throws JSONException JSON
 */
JSONObject toJSONObjectContent() throws JSONException;

/**
 * ,
 *
 * @param mod
 * @return
 */
Object __mod_column_row(MOD_COLUMN_ROW mod);

enum Status {

    SUCCESS(0), ERROR(1), WARNING(2);

    private int type;
```

```
        Status(int type) {
            this.type = type;
        }

        public static Status parse(int type) {
            for (Status status : Status.values()) {
                if (status.type == type) {
                    return status;
                }
            }
            return Status.SUCCESS;
        }

    }
}
```

ValueVerifier.java

```
/*
 * Copyright(c) 2001-2010, FineReport Inc, All Rights Reserved.
 */
package com.fr.report.write;

import com.fr.data.VerifyItem;
import com.fr.general.data.MOD_COLUMN_ROW;
import com.fr.general.xml.GeneralXMLTools;
import com.fr.script.Calculator;
import com.fr.stable.FormulaProvider;
import com.fr.stable.xml.XMLConstants;
import com.fr.stable.xml.XMLPrintWriter;
import com.fr.stable.xml.XMLReadable;
import com.fr.stable.xml.XMLableReader;

import java.util.ArrayList;
import java.util.List;

/**
 * ValueVerifier.
 */
public class ValueVerifier extends AbstractVerifier {

    private List<VerifyItem> verifyItems;

    /**
     * Constructor.
     */
    public ValueVerifier() {

    }


    public void addVerifyItem(VerifyItem item) {
        if (verifyItems == null) {
            verifyItems = new ArrayList<VerifyItem>();
        }
        verifyItems.add(item);
    }

    public VerifyItem getVerifyItem(int index) {
        if (verifyItems == null) {
            return null;
        }
        if (index < 0 || index > verifyItems.size() - 1) {
            return null;
        }
        return verifyItems.get(index);
    }
```

```
public int getVerifyItemsCount() {
    return verifyItems == null ? 0 : verifyItems.size();
}

public void clearVerifyItems() {
    if (verifyItems != null) {
        verifyItems.clear();
    }
}

public Object __mod_column_row(MOD_COLUMN_ROW mod) {
    for (int i = 0, len = getVerifyItemsCount(); i < len; i++) {
        getVerifyItem(i).__mod_column_row(mod);
    }
    return mod;
}

/**
 * WB
 *
 * @param ca
 * @throws Exception
 */
public void execute(Calculator ca) throws Exception {

}

public void readXML(XMLableReader reader) {
    super.readXML(reader);
    if (reader.isAttr()) {
        clearVerifyItems();
    } else if (reader.isChildNode()) {
        String tagName = reader.getTagName();
        if (tagName.equals(VerifyItem.XML_TAG)) {
            VerifyItem item = (VerifyItem) GeneralXMLTools.readXMLable(reader);
            addVerifyItem(item);
        } else {
            readOldVersion(reader, tagName);
        }
    }
}

private void readOldVersion(XMLableReader reader, String tagName) {
    if ("VV".equals(tagName)) {
        final VerifyItem item = new VerifyItem();
        reader.readXMLObject(new XMLReadable() {
            public void readXML(XMLableReader reader) {
                String tagName = reader.getTagName();
                if (XMLConstants.OBJECT_TAG.equals(tagName)) {
                    Object obj = GeneralXMLTools.readObject(reader);
                    if (obj instanceof FormulaProvider) {
                        item.setFormula((FormulaProvider) obj);
                    }
                } else if ("Message".equals(tagName)) {
                    item.setMessage(reader.getElementValue());
                }
            }
        });
        addVerifyItem(item);
    }
}

public void writeXML(XMLPrintWriter writer) {
    super.writeXML(writer);
    if (verifyItems != null) {
        for (VerifyItem item : verifyItems) {
            GeneralXMLTools.writeXMLable(writer, item, VerifyItem.XML_TAG);
        }
    }
}
```

```
    /**
     *
     *
     * @return
     */
    public String toString() {
        return "Waiting complete!";
    }

    /**
     * Clone.
     */
    public Object clone() throws CloneNotSupportedException {
        ValueVerifier cloned = (ValueVerifier) super.clone();
        cloned.verifyItems = new ArrayList<VerifyItem>();
        for (int i = 0, len = getVerifyItemsCount(); i < len; i++) {
            cloned.addVerifyItem((VerifyItem) getVerifyItem(i).clone());
        }
        return cloned;
    }
}
```

FR	8.0		
FR	9.0		
FR	10.0		

plugin.xml

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

Set<VerifyDefineProvider> set = ExtraDesignClassManager.getInstance().getArray(VerifyDefineProvider.MARK_STRING);

VerifierListPane#createNameableCreatorscpt

clone

web

demodemo-verify-define-provider