com.finebi.provider.api.component. CustomComponentProvider

CustomComponentProvider

CustomComponentProvider.java

```
package com.finebi.provider.api.component;

import com.finebi.common.context.OperationContext;
import com.finebi.provider.api.component.data.DataModel;
import com.fr.common.annotations.Open;
import com.fr.stable.fun.mark.Mutable;
import com.fr.web.struct.AssembleComponent;

import java.util.List;

/**
 * <p></p>
 *
 * @author Jimmy.Zheng
 * @since 6.0
 * Created on 2023/5/22
 **/
@Open
public interface CustomComponentProvider extends Mutable {
    String XML_TAG = "CustomComponentProvider";

    int CURRENT_LEVEL = 1;

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

    /**
     *
     *
     * @return
     */
```

```
    String getType();

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

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

    /**
     * html
     *
     * @param context
     * @return html
     */
    String getEditPageHTML(OperationContext context);

    /**
     * css js
     *
     * @param context
     * @return css js
     */
    AssembleComponent editClient(OperationContext context);

    /**
     * html
     *
     * @param context
     * @return html
     */
    String getPreviewPageHTML(OperationContext context);

    /**
     * css js
     *
     * @param context
     * @return css js
     */
    AssembleComponent previewClient(OperationContext context);


    String config();

    /**
     *
     *
     * @param customComponentContext
     * @return
     */
    boolean needDataProcess(CustomComponentContext customComponentContext);

    /**
     * BI
     *
     * @param dataModels
     * @param customComponentContext
     * @return
     */
    List<DataModel> process(List<DataModel> dataModels, CustomComponentContext customComponentContext);
}
```

bi.plugin.client.ts

```
import { BIPluginAction, BIPluginMessage } from '@webui/bi/plugin/bi.plugin.constant';

export class BIPluginClient {
    private connID: string | null = null;

    /**
     * datanull
     * config
     * saveSessionCallback(data: Object): PromisejsonPromise
     * closeSessionCallbackBI
     * @param render
     */
    public init(
        render: (
            data: any,
            config: any,
            saveSessionCallback: (data: any) => Promise<any>,
            closeSessionCallBack: () => void,
            extensionCallback?: (action: string, value: any) => void
        ) => void
    ): void {
        addEventListener('message', (event: MessageEvent) => {
            if (event.source === window.parent && event.data) {
                const eventData = JSON.parse(event.data);
                if (!this.connID && eventData.action === BIPluginAction.INIT) {
                    this.connID = eventData.connID;
                } else {
                    if (this.connID === eventData.connID && eventData.action === BIPluginAction.RENDER) {
                        const { data, config } = eventData;
                        render(
                            data,
                            config,
                            this.saveSessionCallback.bind(this),
                            this.closeSessionCallback.bind(this),
                            this.extensionCallback.bind(this)
                        );
                    }
                }
            }
        });
    }

    private sendMessage(action: BIPluginAction, data: any, config?: any) {
        if (this.connID) {
            const message: BIPluginMessage = { connID: this.connID, action, data, config };
            parent.postMessage(JSON.stringify(message), '*');
        }
    }

    private saveSessionCallback(data: any): Promise<any> {
        return new Promise(resolve => {
            const saveListener = (event: MessageEvent) => {
                if (event.source === window.parent && event.data) {
                    const eventData = JSON.parse(event.data);
                    if (this.connID === eventData.connID && eventData.action === BIPluginAction.SAVE_COMPLETE) {
                        removeEventListener('message', saveListener);
                        resolve(eventData.data);
                    }
                }
            };
            this.sendMessage(BIPluginAction.SAVE, data);
            addEventListener('message', saveListener);
        });
    }
```

```
    private closeSessionCallback(): void {
        this.sendMessage(BIPluginAction.CLOSE, null);
    }

    private extensionCallback(action: string, value: any) {
        this.sendMessage(action, value);
    }
}

// @ts-ignore
window.BIPlugin = BIPluginClient;
```

config.json json

dataModelfiledscolData

dataModelfields

BI	6.0.13-		

plugin.xml

```
<extra-core>
    <CustomComponentProvider class="your class name"/>
</extra-core>
```

FinePluginService

com.finebi.common.api.service.plugin.FinePluginService#getCustomComponentPagehtml

com.finebi.common.api.service.plugin.FinePluginService#getAllCustomComponentInfoname

demoplugin-bi-custom-chart-demo