How to Debug JSDialogs

Szymon Kłos
Software Engineer
szymon.klos@collabora.com

Collabora Online

Technical Day
COOL days

LibreOffice Technology

Example of a problem

“I click on a widget and it does nothing”

Where we should look first?

1. Browser console

It should emit command which will be sent to the server

```
OUTGOING: dialogevent 2 {"id":"nextstyle", "cmd":
"selected", "data": "1;Endnote", "type": "combobox"}
```

```
dialogevent <WINDOW ID> {"id":<WIDGET ID>, "cmd":
<ACTION>, "data": <PARAMETERS>, "type": <WIDGET TYPE>}
```

Processing user commands

2. Executor in the server (**vcl/jsdialog/executor.cxx**)

It should find a window and the widget.

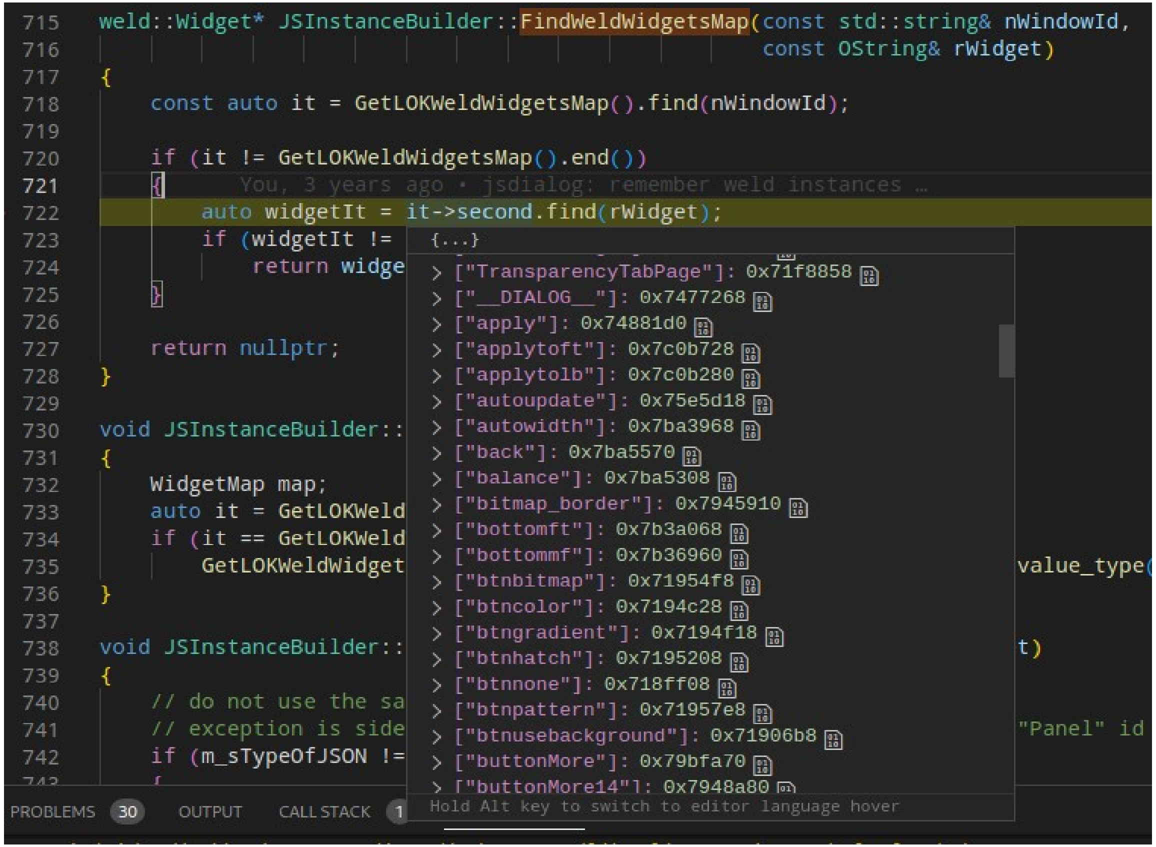
```
bool ExecuteAction(const std::string& nWindowId, const OString&
rWidget, StringMap& rData)
{
    weld::Widget* pWidget =
        JSInstanceBuilder::FindWeldWidgetsMap(nWindowId, rWidget);
```

Check pWidget result. If it is non-null follow the code and see if action is implemented correctly.

Finding widget in a register

3. If widget is not found: look at map with registered widgets
(**vcl/jsdialog/jsdialogbuilder.cxx**)

- widget is missing?
- was registered?
- maybe is part of tab page?



```
weld::Widget* JSInstanceBuilder::FindWeldWidgetsMap(const std::string& nWindowId,
                                                     const OString& rWidget)
{
    const auto it = GetLOKWeldWidgetsMap().find(nWindowId);

    if (it != GetLOKWeldWidgetsMap().end())
    {        You, 3 years ago • jsdialog: remember weld instances …
        auto widgetIt = it->second.find(rWidget);
        if (widgetIt !=
            return widge
    }

    return nullptr;
}

void JSInstanceBuilder::
{
    WidgetMap map;
    auto it = GetLOKWeld
    if (it == GetLOKWeld
        GetLOKWeldWidget                                         value_type(
}

void JSInstanceBuilder::                                            t)
{
    // do not use the sa
    // exception is side                                            "Panel" id
    if (m_sTypeOfJSON !=
        {
```

```
{...}
> ["TransparencyTabPage"]: 0x71f8858
> ["__DIALOG__"]: 0x7477268
> ["apply"]: 0x74881d0
> ["applytoft"]: 0x7c0b728
> ["applytolb"]: 0x7c0b280
> ["autoupdate"]: 0x75e5d18
> ["autowidth"]: 0x7ba3968
> ["back"]: 0x7ba5570
> ["balance"]: 0x7ba5308
> ["bitmap_border"]: 0x7945910
> ["bottomft"]: 0x7b3a068
> ["bottommf"]: 0x7b36960
> ["btnbitmap"]: 0x71954f8
> ["btncolor"]: 0x7194c28
> ["btngradient"]: 0x7194f18
> ["btnhatch"]: 0x7195208
> ["btnnone"]: 0x718ff08
> ["btnpattern"]: 0x71957e8
> ["btnusebackground"]: 0x71906b8
> ["buttonMore"]: 0x79bfa70
> ["buttonMore14"]: 0x7948a80
Hold Alt key to switch to editor language hover
```

Verify action implementation

4. If widget was found check action (**vcl/jsdialog/executor.cxx**)

Maybe we don’t handle action yet?

Do we call callbacks for client code? (eg. LOKTrigger::trigger_changed())

```
if (sAction == "change")
{
    // it might be other class than JSComboBox
    auto pJSCombobox = dynamic_cast<JSComboBox*>(pWidget);
    if (pJSCombobox)
        pJSCombobox->set_entry_text_without_notify(rData["data"]);
    else
        pCombobox->set_entry_text(rData["data"]);
    LOKTrigger::trigger_changed(*pCombobox);
    return true;
}
```

Check dialog implementation

5. If action is executed correctly. What should happen?

How we react on action in the dialog implementation?

Maybe callback is not triggered?

```
m_xLeftMF->connect_value_changed(LINK(this, SvxBorderTabPage, ModifyDistanceHdl_Impl));
```

```
IMPL_LINK( SvxBorderTabPage, ModifyDistanceHdl_Impl, weld::MetricSpinButton&, rField, void)
{
    . . .
     m_xRightMF->set_value(nVal, FieldUnit::NONE);
   . . .
```

Do we send notification in JSDialog wrapper for used actions?
(**set_value()** function for **weld::MetricSpinButton**)

Check JSDialog wrapper for widget

6. JSDialog wrapper should override function (**vcl/jsdialog/jsdialogbuilder.cxx**)

It should call **sendUpdate()** or **sendAction()** to notify client about changes:

```
void JSSpinButton::set_value(sal_Int64 value)
{
    SalInstanceSpinButton::set_value(value);

    std::unique_ptr<jsdialog::ActionDataMap> pMap =
std::make_unique<jsdialog::ActionDataMap>();
    (*pMap)[ACTION_TYPE] = "setText";
    (*pMap)["text"] = OUString::number(m_rFormatter.GetValue());

    sendAction(std::move(pMap));
}
```

Thank you!

By Szymon Kłos

@CollaboraOffice
hello@collaboraoffice.com
www.collaboraoffice.com

Collabora Online

Technical Day

COOL days

LibreOffice Technology