Hollywood Programming

Class 3: Advanced level
Making Applications with RapaGUI
github.com/SamuraiCrow/AmiWest2022
github.com/SamuraiCrow/RapaEdit

Presented by Samuel D. Crow

RapaGUI Plug-in

- Cross-platform GUI plug-in
- Uses MUI on Amiga-likes
- Uses wxWidgets on Win/Mac/Linux
- Uses custom widgets on Android
- Doesn’t support iOS yet

RapaEdit Walk-through

- RapaEdit is an editor for GUI interfaces made for RapaGUI
- It employs the most advanced modular techniques possible in Hollywood: Object-Oriented Programming style
- Apache 2.0 Open-Source license

RapaEdit’s Own GUI

```
<?xml version="1.0" encoding="iso-8859-1"?>
<application id="RapaEdit">
    <menubar id="menu">
        <menu title="File" id="mn_file">
            <item id="mn_new" help="Start a new document">_New</item>
            <item id="mn_open" help="Open an existing document">_Open</item>
            <item id="mn_save" help="Save document to existing file">_Save</item>
            <item id="mn_saveas" help="Save document to a new file">Save _As...</item>
            <item />
            <item id="mn_preview" help="Preview GUI">_Preview</item>
            <item id="mn_export" help="Export to new XML file">E_xport Script As...</item>
            <item />
            <item id="mn_quit" help="Exit the program">_Quit</item>
        </menu>
        <menu title="Gadget" id="mn_gadget">
        ...
```

Toolbar

```
<window id="win_" title="RapaEdit" menubar="menu" notify="CloseRequest" width="640" height="400">
    <vgroup>
        <toolbar>
            <button id="tb_new" icon="2">New</button>
            <button />
            <button id="tb_open" icon="3">Open</button>
            <button id="tb_close" icon="4">Close</button>
            <button />
            <button id="tb_save" icon="5">Save</button>
            <button id="tb_saveas" icon="6">Save As</button>
            <button />
            <button id="tb_upgadget" icon="10">Move Up</button>
            <button id="tb_downgadget" icon="11">Move Down</button>
        </toolbar>
```

Sidebar

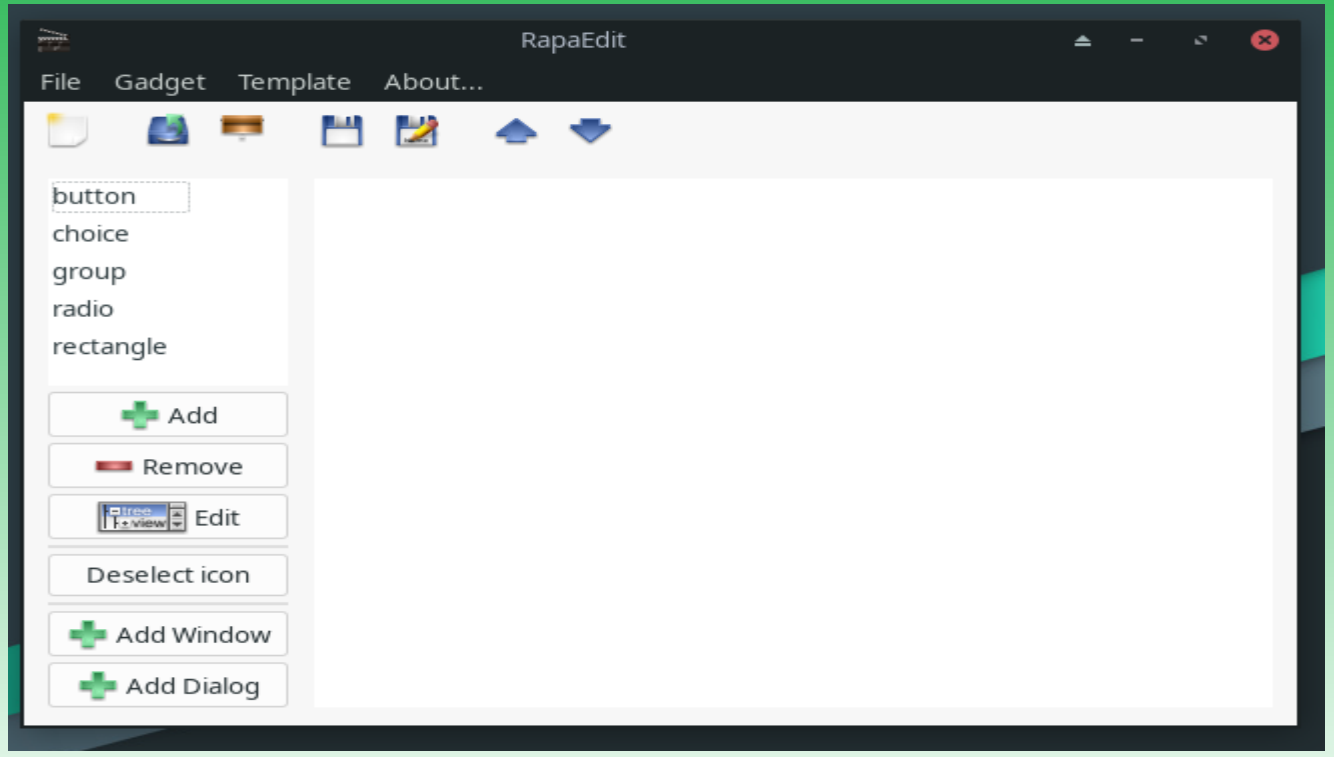
```
<hsplitter id="spl_split">
    <vgroup>
        <listview id="lv_gadgets" forcemode="listview" notify="DoubleClick">
            <column sortable="1" />
        </listview>
        <button id="tb_addgadget" icon="7">Add</button>
        <button id="tb_removegadget" icon="8">Remove</button>
        <button id="tb_editgadget" icon="9">Edit</button>
        <hline />
        <button id="tree_deselect">Deselect icon</button>
        <hline />
        <button id="wa_window" icon="7" >Add Window</button>
        <button id="wa_dialog" icon="7" >Add Dialog</button>
    </vgroup>
```

Main Treeview and Closers

```
            <treeview id="tree_hierarchy">
                <column />
            </treeview>
        </hsplitter>
    </vgroup>
</window>
</application>
```

- This is what the editor generates
- But it does it graphically!

What it Looks Like



Event Handling

- Gadgets are identified by the ID sub-tag in the XML definition (case-sensitive)
- Events are generated based on the gadget type
- The message data is another member of the event table type
- Usually handled with a large switch statement but I handle it with table look-ups

Global Event Handler

```
Function p_ProcessGUI(message)
   Switch message.action
   Case "RapaGUI":
      Local prefixLength=FindStr(message.id, "_")+1
      ;Check for Local prefixes and event handlers
      Local prefix$=LeftStr(message.id, prefixLength)
      Local handler=RawGet(prefixes, prefix$)
      ;Invoke local event handler from class
      handler(message, prefixLength)
   EndSwitch
EndFunction
```

Show Me the Source!

Here it comes!