```zig
const std = @import("std");

const dds = @import("dds");

// keyboard
const kbd = @import("cursed").kbd;

// panel
const pnl = @import("forms").pnl;
// button
const btn = @import("forms").btn;
// label
const lbl = @import("forms").lbl;
// menu
const mnu = @import("forms").mnu;
// flied
const fld = @import("forms").fld;
// line horizontal
const lnh = @import("forms").lnh;
// line vertival
const lnv = @import("forms").lnv;

// grid
const grd = @import("grid").grd;

// full delete for produc
const forms = @import("forms");
const allocator = std.heap.page_allocator;

pub fn SavJson(XPANEL: *std.ArrayList(pnl.PANEL)) !void {
    var fjson :std.fs.File = undefined;

    fjson = try std.fs.cwd().createFile("Zdspf.txt", .{ .read = true });

    const out  = fjson.writer();

    var w = std.json.writeStream(out, .{ .whitespace = .indent_2 });

    const Ipanel = std.enums.EnumIndexer(pnl.Epanel);

    try w.beginObject();
    try w.objectField("PANEL");
    var nbrPnl: usize = XPANEL.items.len;
    var np: usize = 0;
    while (np < nbrPnl) : (np += 1) {
        try w.beginArray();
```

```
47:         try w.beginObject();
48:         var p: usize = 0;
49:         while (p < Ipanel.count) : (p += 1) {
50:             switch (Ipanel.keyForIndex(p)) {
51:                 .name => {
52:                     try w.objectField(@tagName(pnl.Epanel.name));
53:                     try w.print("\"{s}\"", .{XPANEL.items[np].name});
54:                 },
55:                 .posx => {
56:                     try w.objectField(@tagName(pnl.Epanel.posx));
57:                     try w.print("{d}", .{XPANEL.items[np].posx});
58:                 },
59:                 .posy => {
60:                     try w.objectField(@tagName(pnl.Epanel.posy));
61:                     try w.print("{d}", .{XPANEL.items[np].posy});
62:                 },
63:                 .lines => {
64:                     try w.objectField(@tagName(pnl.Epanel.lines));
65:                     try w.print("{d}", .{XPANEL.items[np].lines});
66:                 },
67:                 .cadre => {
68:                     try w.objectField(@tagName(pnl.Epanel.cadre));
69:                     try w.print("{s}", .{@tagName(XPANEL.items[np].frame.cadre)});
70:                 },
71:                 .title => {
72:                     try w.objectField(@tagName(pnl.Epanel.title));
73:                     try w.print("{s}", .{XPANEL.items[np].frame.title});
74:                 },
75:                 .button => {
76:                     const Ibutton = std.enums.EnumIndexer(btn.Ebutton);
77:                     var nbrBtn: usize = XPANEL.items[np].button.items.len;
78:                     var bp: usize = 0;
79:                     try w.objectField("button");
80:                     try w.beginArray();
81:                     while (bp < nbrBtn) : (bp += 1) {
82:                         try w.beginObject();
83:                         var b: usize = 0;
84:                         while (b < Ibutton.count) : (b += 1) {
85:                             switch (Ibutton.keyForIndex(b)) {
86:                                 .name => {
87:                                     try w.objectField(@tagName(btn.Ebutton.name));
88:                                     try w.print("\"{s}\"", .{XPANEL.items[np].button.items[bp].name});
89:                                 },
90:                                 .key => {
91:                                     try w.objectField(@tagName(btn.Ebutton.key));
92:                                     try w.print("\"{s}\"", .{@tagName(XPANEL.items[np].button.items[bp].key)});
```

```
                                },
                                .show => {
                                    try w.objectField(@tagName(btn.Ebutton.show));
                                    try w.print("{d}", .{@intFromBool(XPANEL.items[np].button.items[bp].show)});
                                },
                                .check => {
                                    try w.objectField(@tagName(btn.Ebutton.check));
                                    try w.print("{d}", .{@intFromBool(XPANEL.items[np].button.items[bp].check)});
                                },
                                .title => {
                                    try w.objectField(@tagName(btn.Ebutton.title));
                                    try w.print("\"{s}\"", .{XPANEL.items[np].button.items[bp].title});
                                },
                            }
                        }
                        try w.endObject();
                    }

                    try w.endArray();
                },
                .label => {
                    const Ilabel = std.enums.EnumIndexer(lbl.Elabel);
                    var l: usize = 0;
                    var nbrLbl: usize = XPANEL.items[np].label.items.len;

                    var lp: usize = 0;
                    try w.objectField("label");
                    try w.beginArray();
                    while (lp < nbrLbl) : (lp += 1) {
                        try w.beginObject();
                        l = 0;
                        while (l < Ilabel.count) : (l += 1) {
                            switch (Ilabel.keyForIndex(l)) {
                                .name => {
                                    try w.objectField(@tagName(lbl.Elabel.name));
                                    try w.print("\"{s}\"", .{XPANEL.items[np].label.items[lp].name});
                                },
                                .posx => {
                                    try w.objectField(@tagName(lbl.Elabel.posx));
                                    try w.print("{d}", .{XPANEL.items[np].label.items[lp].posx});
                                },
                                .posy => {
                                    try w.objectField(@tagName(lbl.Elabel.posy));
                                    try w.print("{d}", .{XPANEL.items[np].label.items[lp].posy});
                                },
                                .text => {
```

```
139:                                     try w.objectField(@tagName(lbl.Elabel.title));
140:                                     try w.print("\"{s}\"", .{XPANEL.items[np].label.items[lp].text});
141:                                 },
142:                                 .title => {
143:                                     try w.objectField(@tagName(lbl.Elabel.title));
144:                                     try w.print("{d}", .{@intFromBool(XPANEL.items[np].label.items[lp].title)});
145:                                 },
146:                             }
147:                         }
148:                         try w.endObject();
149:                     }
150:
151:                     try w.endArray();
152:                 },
153:                 .field => {
154:                     const Ifield = std.enums.EnumIndexer(fld.Efield);
155:                     var f: usize = 0;
156:                     var nbrFld: usize = XPANEL.items[np].field.items.len;
157:
158:                     var fp: usize = 0;
159:                     try w.objectField("field");
160:                     try w.beginArray();
161:                     while (fp < nbrFld) : (fp += 1) {
162:                         try w.beginObject();
163:                         f = 0;
164:                         while (f < Ifield.count) : (f += 1) {
165:                             switch (Ifield.keyForIndex(f)) {
166:                                 .name => {
167:                                     try w.objectField(@tagName(fld.Efield.name));
168:                                     try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].name});
169:                                 },
170:                                 .posx => {
171:                                     try w.objectField(@tagName(fld.Efield.posx));
172:                                     try w.print("{d}", .{XPANEL.items[np].field.items[fp].posx});
173:                                 },
174:                                 .posy => {
175:                                     try w.objectField(@tagName(fld.Efield.posy));
176:                                     try w.print("{d}", .{XPANEL.items[np].field.items[fp].posy});
177:                                 },
178:                                 .reftyp => {
179:                                     try w.objectField(@tagName(fld.Efield.reftyp));
180:                                     try w.print("{s}", .{@tagName(XPANEL.items[np].field.items[fp].reftyp)});
181:                                 },
182:                                 .width => {
183:                                     try w.objectField(@tagName(fld.Efield.width));
184:                                     try w.print("{d}", .{XPANEL.items[np].field.items[fp].width});
```

```
185:                                     },
186:                                     .scal => {
187:                                         try w.objectField(@tagName(fld.Efield.scal));
188:                                         try w.print("{d}", .{XPANEL.items[np].field.items[fp].scal});
189:                                     },
190:                                     .requier => {
191:                                         try w.objectField(@tagName(fld.Efield.requier));
192:                                         try w.print("{d}", .{@intFromBool(XPANEL.items[np].field.items[fp].requier)});
193:                                     },
194:                                     .protect => {
195:                                         try w.objectField(@tagName(fld.Efield.protect));
196:                                         try w.print("{d}", .{@intFromBool(XPANEL.items[np].field.items[fp].protect)});
197:                                     },
198:                                     .edtcar => {
199:                                         try w.objectField(@tagName(fld.Efield.edtcar));
200:                                         try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].edtcar});
201:                                     },
202:                                     .errmsg => {
203:                                         try w.objectField(@tagName(fld.Efield.errmsg));
204:                                         try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].errmsg});
205:                                     },
206:                                     .help => {
207:                                         try w.objectField(@tagName(fld.Efield.help));
208:                                         try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].help});
209:                                     },
210:                                     .procfunc => {
211:                                         try w.objectField(@tagName(fld.Efield.procfunc));
212:                                         try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].procfunc});
213:                                     },
214:                                     .proctask => {
215:                                         try w.objectField(@tagName(fld.Efield.proctask));
216:                                         try w.print("\"{s}\"", .{XPANEL.items[np].field.items[fp].proctask});
217:                                     },
218:                                 }
219:                             }
220:                             try w.endObject();
221:                         }
222:
223:                         try w.endArray();
224:                     },
225:                     else => {},
226:                 }
227:             }
228:             try w.endObject();
229:             try w.endArray();
230:         }
```

```
231:     try w.endObject();
232:
233:     fjson.close();
234: }
```