

```

1 /*
2 <span id="gsh" data-title="GShell" data-author="sato@its-more.jp">
3 <meta charset="UTF-8">
4 <meta name="viewport" content="width=device-width, initial-scale=1.0">
5 <link rel="icon" id="GshFaviconURL" href=""><!-- place holder -->
6 <span id="GshVersion" hidden="">gsh-0.7.8-2020-11-01--SatoxITS</span>
7 <title id="GshTitle">GShell-0.7.8 by SatoxITS</title>
8
9 <div id="GshHeading">
10 <div id="GshTopbar" class="MetaWindow"></div>
11 <div id="GshPerfMon"></div>
12 <div id="GshSidebar" class="SbSidebar" style=""><div id="GshIndexer" style=""></div></div>
13 </div>
14 <div id="GshMain">
15 <div id="GshNames" height="100px" onclick="shiftBg();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.7.8 // 2020-11-01 // SatoxITS</note></div>
17 </div>
18 */
19
20
21 <!-- ===== Work { ===== -->
22 <span id="CascadedCanvasBook_WorkCodeSpan">
23 /*
24 <details open=""><summary>CascadedCanvasBook</summary>
25 </-- ===== CascadedCanvasBook // 2020-1031 SatoxITS { -->
26 <h2>Cascaded Canvas Book</h2>
27
28 <!--
29 <div id="CBPanel" class="CBPanel">
30 <input id="cb_new" type="button" value="NewCanvas">
31 </div>
32 -->
33 <br>
34
35 <h3>Undo / Redo / Replay</h3>
36
37 <div id="CanvasTool_UndoRedo">
38 <span id="DrawReplay" class="CanvasTool" draggable="true" contenteditable>
39 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
40 <input class="CV_Button" type="button" value="Redraw" value="Redraw">
41 From <input data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
42 To <input id="DrawingSernoView" data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
43 </span>
44 </div>
45
46 <script>
47 function OnWheelInt(){
48 event.preventDefault();
49 t = event.target;
50 n = t.nodeName;
51 i = t.id;
52 p = t.parentNode;
53 y = event.deltaY;
54 //console.log('OnWheelInt '+y+' '+n+'#'+i+' '+t.value);
55 if ( y < 0 ) { // scroll forward (up)
56 inc = -y;
57 }else{
58 inc = -y;
59 }
60 inc /= 6;
61 val = parseFloat(t.value) + inc;
62 t.value = val.toFixed(0);
63 return val;
64 }
65 var DrawingSerno = 0;
66 function saveDrawing(){
67 DrawingSerno += 1;
68 DrawingSernoView.value = DrawingSerno;
69 }
70 </script>
71
72 <h3>Color</h3>
73
74 <div id="CanvasColors">
75 <div id="CanvasTool_Color_0" class="CanvasTool" onchange="showColorSample()">
76 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
77 <input class="CV_Button" type="button" value="Trans">
78 <span data-name="I" class="ColorParam" type="text">C0-0</span>
79 BW<input data-name="BW" class="CanvasParam" type="text" value="0">
80 FC<input data-name="FC" class="ColorParam" type="text" value="#000000">
81 <input data-name="FO" class="CanvasParam" type="text" value="0.0">
82 <span data-name="FCS" class="ColorSample">xxx</span>
83 <span data-name="BCS" class="ColorSample">xxx</span>
84 BC<input data-name="BC" class="ColorParam" type="text" value="#000000">
85 <input data-name="BO" class="CanvasParam" type="text" value="0.0">
86 </div>
87 <div id="CanvasTool_Color_1" class="CanvasTool" onchange="showColorSample()">
88 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
89 <input class="CV_Button" type="button" value="Mono">
90 <span data-name="I" class="ColorParam" type="text">C0-1</span>
91 BW<input data-name="BW" class="CanvasParam" type="text" value="1">
92 FC<input data-name="FC" class="ColorParam" type="text" value="#000000">
93 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
94 <span data-name="FCS" class="ColorSample">xxx</span>
95 <span data-name="BCS" class="ColorSample">xxx</span>
96 BC<input data-name="BC" class="ColorParam" type="text" value="#d0d0d0">
97 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
98 </div>
99 <div id="CanvasTool_Color_2" class="CanvasTool" onchange="showColorSample()">
100 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
101 <input class="CV_Button" type="button" value="Red">
102 <span data-name="I" class="ColorParam" type="text">C0-2</span>
103 BW<input data-name="BW" class="CanvasParam" type="text" value="1">
104 FC<input data-name="FC" class="ColorParam" type="text" value="#82020">
105 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
106 <span data-name="FCS" class="ColorSample">xxx</span>
107 <span data-name="BCS" class="ColorSample">xxx</span>
108 BC<input data-name="BC" class="ColorParam" type="text" value="#####">
109 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
110 </div>
111 <div id="CanvasTool_Color_3" class="CanvasTool" onchange="showColorSample()">
112 <input class="CV_Button" type="button" value="Green">
113 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
114 <span data-name="I" class="ColorParam" type="text">C0-3</span>
115 BW<input data-name="BW" class="CanvasParam" type="text" value="1">
116 FC<input data-name="FC" class="ColorParam" type="text" value="#20c820">
117 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
118 <span data-name="FCS" class="ColorSample">xxx</span>
119 <span data-name="BCS" class="ColorSample">xxx</span>
120 BC<input data-name="BC" class="ColorParam" type="text" value="#####">
121 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
122 </div>
123 <div id="CanvasTool_Color_4" class="CanvasTool" onchange="showColorSample()">
124 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
125 <input class="CV_Button" type="button" value="Blue">
126 <span data-name="I" class="ColorParam" type="text">C0-4</span>
127 BW<input data-name="BW" class="CanvasParam" type="text" value="1">
128 FC<input data-name="FC" class="ColorParam" type="text" value="#6080f0">
129 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
130 <span data-name="FCS" class="ColorSample">xxx</span>
131 <span data-name="BCS" class="ColorSample">xxx</span>
132 BC<input data-name="BC" class="ColorParam" type="text" value="#c0c0c0">
133 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
134 </div>
135 <div id="CanvasTool_Color_5" class="CanvasTool" onchange="showColorSample()">
136 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
137 <input class="CV_Button" type="button" value="Yellow">
138 <span data-name="I" class="ColorParam" type="text">C0-5</span>
139 BW<input data-name="BW" class="CanvasParam" type="text" value="1">
140 FC<input data-name="FC" class="ColorParam" type="text" value="#fae60">
141 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
142 <span data-name="FCS" class="ColorSample">xxx</span>
143 <span data-name="BCS" class="ColorSample">xxx</span>
144 BC<input data-name="BC" class="ColorParam" type="text" value="#####">
145 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
146 </div>
147 </div>
148
149 <script>
150 function childByName(node,name){
151 for( let i = 0; i < node.children.length; i++ ){
152 ch = node.children[i];
153 name1 = ch.getAttribute('data-name');
154 if( name1 == name ){
155 return ch;
156 }
157 }
158 return null;
159 }
160 // https://developer.mozilla.org/en-US/docs/Web/CSS/color_value
161 function genCSSColorStyle(color,opa){
162 opa = parseFloat(opa);
163 opa *= 0xFF;
164 opa = opa.toFixed(0);
165 if( 0xFF < opa ) opa = 0xFF;
166 opa = parseInt(opa);
167 opa = opa.toString(16);
168 if( opa < 0x10 ) opa = '0' + opa;
169 color = opa;
170 return color;
171 }
172 function CV_GenColorStyle(cole,forFill){
173 if( forFill ){
174 col = childByName(cole, 'FC').value;
175 opa = childByName(cole, 'FO').value;
176 }else{

```

```

177     col = childByName(cole, 'BC').value;
178     opa = childByName(cole, 'BO').value;
179 }
180 color = genCSSColorStyle(col, opa);
181 return color;
182 }
183 function xxxshowColorSample(){
184     t = event.target;
185     p = t.parentNode;
186     alert('showColorSample '+event.target.nodeName+'/#'+p.id);
187 }
188 function showColorSample(){
189     var csv = CanvasColors.children;
190     //console.log('colors'+csv.length);
191     for( i = 0; i < csv.length; i++){
192         fc = childByName(csv[i], 'FC').value;
193         bc = childByName(csv[i], 'BC').value;
194         //console.log('colors'+csv[i].id+' fc='+fc+' bc='+bc);
195         fcs = childByName(csv[i], 'FCS');
196         fcs.style.color = fc;
197         fcs.style.borderColor = fc;
198         fcs.style.backgroundColor = bc;
199
200         bcs = childByName(csv[i], 'BCS');
201         bcs.style.color = bc;
202         bcs.style.borderColor = fc;
203         bcs.style.backgroundColor = fc;
204     }
205     CV_redrawParts();
206 }
207 </script>
208
209 <style>
210 .ColorSample {
211     color:#fff;
212     background-color:#ff0;
213     border:1px solid #000;
214     margin:0px;
215     padding:10px;
216     width:12pt !important;
217     height:12pt !important;
218 }
219 .ColorParam {
220     color:#000 !important;
221     font-family:Courier New !important;
222     font-size:9pt !important;
223     padding:2px !important;
224     line-height:1.1 !important;
225     height:14pt !important;
226     width:45pt !important;
227     text-align:left !important;
228     display:inline !important;
229     vertical-align:middle !important;
230 }
231 .CanvasParam {
232     color:#000 !important;
233     font-family:Courier New, Monospace !important;
234     font-size:9pt !important;
235     padding:2px !important;
236     line-height:1.1 !important;
237     height:14pt !important;
238     width:30pt !important;
239     text-align:right !important;
240     display:inline !important;
241     vertical-align:middle !important;
242 }
243 .CV_Button {
244     padding:2pt !important;
245     line-height:1.1 !important;
246     border:2px inset #bbb !important;
247     font-size:9pt !important;
248     font-weight:normal !important;
249     font-family:Georgia !important;
250     border-radius:3px !important;
251     color:#ddd; background-color:#66a !important;
252     width:50pt;
253 }
254 .xxHtmlCodeviewText {
255     font-size:9pt;
256     font-family:Courier New;
257     white-space:pre;
258 }
259 .xxCV_Button {
260     font-family:Arial, Monospace, Courier New;
261     color:#000;
262     font-size:9pt;
263     line-height:1.2;
264     width:50pt;
265 }
266 </style>
267
268 <h3>Parts</h3>
269 <div id="AppendToCanvas" class="CanvasTool" draggable="true">
270     Resize<input class="CanvasParam" type="text" value="100" onwheel="OnWheelResize()">
271     Zoom<input class="CanvasParam" type="text" value="100" onwheel="OnWheelZoom()">
272     <input id="RedrawImmediate" type="checkbox" value="Redraw" checked="checked" Redraw Immediate
273 </div>
274 <span id="CanvasTools" class="CanvasTools" draggable="true" contenteditable>
275 <div id="CanvasTool_Clear" class="CanvasTool">
276     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
277     <input data-name="rdr" class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
278     <span data-name="cvid" class="CanvasParam" type="text">CL-0</span>
279     <input type="checkbox" value="Fill" checked="checked">
280     <input data-name="X" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
281     <input data-name="Y" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
282     <input data-name="Z" class="CanvasParam" type="text" value="1" onwheel="OnWheelIntRedraw()">
283     <input data-name="W" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
284     <input data-name="H" class="CanvasParam" type="text" value="1000" onwheel="OnWheelIntRedraw()">
285     <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
286     <input data-name="C" class="CanvasParam" type="text" value="0">
287 </div>
288 <div id="CanvasTool_Rect" class="CanvasTool">
289     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
290     <input data-name="rdr" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()">
291     <span data-name="cvid" class="CanvasParam" type="text">RE-0</span>
292     <input data-name="F" type="checkbox" value="Fill" checked="checked">
293     <input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
294     <input data-name="Y" class="CanvasParam" type="text" value="60" onwheel="OnWheelIntRedraw()">
295     <input data-name="Z" class="CanvasParam" type="text" value="2" onwheel="OnWheelIntRedraw()">
296     <input data-name="W" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()">
297     <input data-name="H" class="CanvasParam" type="text" value="80" onwheel="OnWheelIntRedraw()">
298     <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
299     <input data-name="C" class="CanvasParam" type="text" value="1">
300 </div>
301 <div id="CanvasTool_Circle" class="CanvasTool">
302     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
303     <input data-name="rdr" class="CV_Button" type="button" value="Circle" onclick="CV_drawCircle()">
304     <span data-name="cvid" class="CanvasParam" type="text">CI-0</span>
305     <input data-name="F" type="checkbox" value="Fill" checked="checked">
306     <input data-name="X" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
307     <input data-name="Y" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
308     <input data-name="Z" class="CanvasParam" type="text" value="3" onwheel="OnWheelIntRedraw()">
309     <input data-name="R" class="CanvasParam" type="text" value="24" onwheel="OnWheelIntRedraw()">
310     <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
311     <input data-name="B" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()">
312     <input data-name="C" class="CanvasParam" type="text" value="2">
313 </div>
314 <div id="CanvasTool_Packman" class="CanvasTool">
315     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
316     <input data-name="rdr" class="CV_Button" type="button" value="Packman" onclick="CV_drawPackman()">
317     <span data-name="cvid" class="CanvasParam" type="text">PA-0</span>
318     <input data-name="F" type="checkbox" value="Fill" checked="checked">
319     <input data-name="X" class="CanvasParam" type="text" value="240" onwheel="OnWheelIntRedraw()">
320     <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
321     <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()">
322     <input data-name="W" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
323     <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
324     <input data-name="B" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
325     <input data-name="C" class="CanvasParam" type="text" value="5">
326 </div>
327 <div id="CanvasTool_Ellipse" class="CanvasTool">
328     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
329     <input data-name="rdr" class="CV_Button" type="button" value="Ellipse" onclick="CV_drawEllipse()">
330     <span data-name="cvid" class="CanvasParam" type="text">EL-0</span>
331     <input data-name="F" type="checkbox" value="Fill" checked="checked">
332     <input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
333     <input data-name="Y" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
334     <input data-name="Z" class="CanvasParam" type="text" value="4" onwheel="OnWheelIntRedraw()">
335     <input data-name="RX" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
336     <input data-name="RY" class="CanvasParam" type="text" value="20" onwheel="OnWheelIntRedraw()">
337     <input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
338     <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
339     <input data-name="B" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()">
340     <input data-name="C" class="CanvasParam" type="text" value="4">
341 </div>
342 <div id="CanvasTool_Baloon" class="CanvasTool">
343     <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
344     <input data-name="rdr" class="CV_Button" type="button" value="Baloon" onclick="CV_drawBaloon()">
345     <span data-name="cvid" class="CanvasParam" type="text">BA-0</span>
346     <input data-name="F" type="checkbox" value="Fill" checked="checked">
347     <input data-name="X" class="CanvasParam" type="text" value="280" onwheel="OnWheelIntRedraw()">
348     <input data-name="Y" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
349     <input data-name="Z" class="CanvasParam" type="text" value="5" onwheel="OnWheelIntRedraw()">
350     <input data-name="RX" class="CanvasParam" type="text" value="50" onwheel="OnWheelIntRedraw()">
351     <input data-name="RY" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
352     <input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">

```

```

354 <input data-name="S" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()">
355 <input data-name="B" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
356 <CO><input data-name="C" class="CanvasParam" type="text" value="4">
357 </div>
358
359 <div id="CanvasTool_Piechart" class="CanvasTool">
360 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
361 <input data-name="rdr" class="CV_Button" type="button" value="Piechart" onclick="CV_drawPiechart()">
362 <span data-name="cvId" class="CanvasParam" type="text" value="BA-0</span>
363 <input data-name="checked" type="checkbox" value="Fill" checked>
364 <input data-name="X" class="CanvasParam" type="text" value="350" onwheel="OnWheelIntRedraw()">
365 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
366 <input data-name="Z" class="CanvasParam" type="text" value="7" onwheel="OnWheelIntRedraw()">
367 <input data-name="R" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
368 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
369 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
370 <input data-name="RX" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
371 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
372 <CO><input data-name="C" class="CanvasParam" type="text" value="3">
373 </div>
374
375 <div id="CanvasTool_XArc1" class="CanvasTool">
376 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
377 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
378 <span data-name="cvId" class="CanvasParam" type="text" value="XA-0</span>
379 <input data-name="F" type="checkbox" value="Fill" checked>
380 <input data-name="X" class="CanvasParam" type="text" value="460" onwheel="OnWheelIntRedraw()">
381 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
382 <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()">
383 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
384 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
385 <input data-name="R0" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()">
386 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
387 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
388 <CO><input data-name="C" class="CanvasParam" type="text" value="4">
389 </div>
390
391 <div id="CanvasTool_XArc2" class="CanvasTool">
392 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
393 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
394 <span data-name="cvId" class="CanvasParam" type="text" value="XA-1</span>
395 <input data-name="checked" type="checkbox" value="Fill" checked>
396 <input data-name="X" class="CanvasParam" type="text" value="580" onwheel="OnWheelIntRedraw()">
397 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
398 <input data-name="Z" class="CanvasParam" type="text" value="8" onwheel="OnWheelIntRedraw()">
399 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
400 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
401 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
402 <input data-name="RX" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
403 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
404 <CO><input data-name="C" class="CanvasParam" type="text" value="2">
405 </div>
406
407 <canvas id="CV_partsCanvas" data-name="canvas" class="CS_Canvas" width="740" height="200"></canvas>
408 </span>
409 <script>
410 function CV_redrawParts(){
411 // search Z-Index and sort
412 var parts = CanvasTools.children;
413 //console.log("parts="+parts.length);
414 np = [];
415 for( i = 0; i < parts.length; i++){
416 p = parts[i];
417 z = childByName(p,'Z');
418 if( z != null ){
419 //console.log('P#'+p.id+' z'+z+'+z.value);
420 np.push([z.value,p]);
421 }
422 }
423 np.sort(function(np1,np2){ return np1[0] - np2[0]; });
424 CV_clearRect();
425 for( i = 0; i < np.length; i++){
426 p = np[i][1];
427 z = childByName(p,'rdr');
428 //console.log("Redraw z="+np[i][0]+' #'+np[i][1].id+' redraw'+redraw.onclick);
429 if( redraw != null ){
430 redraw.click();
431 }
432 }
433 }
434 function DrawingCanvas_Setup(){
435 showColorSample();
436 CV_redrawParts();
437 }
438 function OnWheelZoom(){
439 val = OnWheelInt();
440 canvas = CV_partsCanvas;
441 canvas.style.zoom = val + 's';
442 CV_redrawParts();
443 }
444 function OnWheelResize(){
445 val = OnWheelInt();
446 canvas = CV_partsCanvas;
447 if( !canvas.hasAttribute('data-width') ){
448 w = canvas.width;
449 h = canvas.height;
450 canvas.setAttribute('data-width',w);
451 canvas.setAttribute('data-height',h);
452 sw = canvas.getAttribute('data-width');
453 sh = canvas.getAttribute('data-height');
454 console.log("Zoom save original w="+w+',h='+h+' sw='+sw+', sh='+sh);
455 }
456 w = canvas.getAttribute('data-width');
457 h = canvas.getAttribute('data-height');
458 console.log("Zoom got original size w="+w+' h='+h);
459 mw = w * (val/100.0);
460 mh = h * (val/100.0);
461 //console.log("Zoom mw="+mw+' nh="+nh);
462 CV_partsCanvas.width = mw;
463 CV_partsCanvas.height = mh;
464 CV_redrawParts();
465 }
466 function OnWheelIntRedraw(){
467 OnWheelInt();
468 t = event.target;
469 n = t.nodeName;
470 i = t.id;
471 p = t.parentNode;
472 y = event.detail.toFixed(0);
473 //console.log("OnWheelIntRedraw '+y+' '+n+'#'+i+' '+t.value+' #'+p.id);
474
475 if( true ){
476 CV_redrawParts();
477 }else{
478 if( RedrawImmediate.checked ){
479 CV_clearRect();
480 //If( p.id == 'CanvasTool_Circle' ){ CV_drawCircle1(CanvasTool_Circle); }
481 //If( p.id == 'CanvasTool_Rect' ){ CV_drawRect(CanvasTool_Rect); }
482 CV_drawCircle1(CanvasTool_Circle);
483 CV_drawRect(CanvasTool_Rect);
484 }
485 }
486 }
487 }
488 }
489
490 function CV_setCtxStyle(ctx,p){
491 C = childByName(p,'C').value;
492 c = document.getElementById('CanvasTool_Color'+C);
493 ctx.fillStyle = CV_GenColorStyle(c,true);
494 ctx.strokeStyle = CV_GenColorStyle(c,false);
495 return ctx;
496 }
497 function CV_clearRect(){
498 cv = document.getElementById('CV_partsCanvas');
499 ctx = cv.getContext('2d');
500 ctx.clearRect(0,0,cv.width,cv.height);
501 }
502 function CV_drawRect(rect){
503 canvas = document.getElementById('CV_partsCanvas');
504 ctx = canvas.getContext('2d');
505 ctx = CV_setCtxStyle(ctx,p);
506
507 p = rect;
508 F = childByName(p,'F').checked;
509 X = childByName(p,'X').value;
510 Y = childByName(p,'Y').value;
511 Z = childByName(p,'Z').value;
512 p.style.zIndex = Z;
513 W = childByName(p,'W').value;
514 H = childByName(p,'H').value;
515
516 if( F ){
517 ctx.fillRect(X,Y,W,H);
518 }else{
519 ctx.strokeRect(X,Y,W,H);
520 }
521 saveDrawing();
522 }
523 function CV_drawRect(rect){
524 CV_drawRect1(CanvasTool_Rect);
525 }
526 function CV_drawCircle1(circle){
527 canvas = document.getElementById('CV_partsCanvas');
528 ctx = canvas.getContext('2d');
529 ctx = CV_setCtxStyle(ctx,p);
530

```

```

531 p = circle;
532 F = childByName(p, 'F').checked;
533 X = childByName(p, 'X').value;
534 Y = childByName(p, 'Y').value;
535 Z = childByName(p, 'Z').value;
536 p.style.zIndex = Z;
537 R = childByName(p, 'R').value;
538 S = childByName(p, 'S').value;
539 E = childByName(p, 'E').value;
540
541 //console.log('Circle'+X+', '+Y+' F='+F);
542 ctx.beginPath();
543 SA = (S / 180) * Math.PI;
544 EA = (E / 180) * Math.PI;
545 ctx.arc(X, Y, R, SA, EA);
546 if( F ){
547   ctx.fill();
548 }else{
549   ctx.stroke();
550 }
551 saveDrawing();
552 }
553 function CV_drawCircle(){
554   CV_drawCircle(CanvasTool_Circle);
555 }
556 function CV_drawEllipse(circle){
557   canvas = document.getElementById('CV_partsCanvas');
558   ctx = canvas.getContext('2d');
559   ctx = CV_setCtxStyle(ctx,p);
560
561   p = circle;
562   X = childByName(p, 'X').value;
563   Y = childByName(p, 'Y').value;
564   Z = childByName(p, 'Z').value;
565   RX = childByName(p, 'RX').value;
566   RY = childByName(p, 'RY').value;
567   p.style.zIndex = Z;
568   RO = childByName(p, 'RO').value;
569   S = childByName(p, 'S').value;
570   E = childByName(p, 'E').value;
571
572   ctx.beginPath();
573   SA = (S / 180) * Math.PI;
574   EA = (E / 180) * Math.PI;
575   ROA = (RO / 180) * Math.PI;
576   ctx.ellipse(X, Y, RX, RY, ROA, SA, EA);
577
578   F = childByName(p, 'F').checked;
579
580   if( F ){
581     ctx.fill();
582   }
583   // if( S ){
584   {
585     ctx.stroke();
586   }
587   saveDrawing();
588 }
589 function CV_drawEllipse(){
590   CV_drawEllipse(CanvasTool_Ellipse);
591 }
592 function CV_drawBaloon1(baloon){
593   canvas = document.getElementById('CV_partsCanvas');
594   ctx = canvas.getContext('2d');
595   ctx = CV_setCtxStyle(ctx,p);
596
597   p = baloon;
598   F = childByName(p, 'F').checked;
599   X = childByName(p, 'X').value;
600   Y = childByName(p, 'Y').value;
601   Z = childByName(p, 'Z').value;
602   RX = childByName(p, 'RX').value;
603   RY = childByName(p, 'RY').value;
604   p.style.zIndex = Z;
605   RO = childByName(p, 'RO').value;
606   S = childByName(p, 'S').value;
607   E = childByName(p, 'E').value;
608
609   //console.log('Ellipse'+X+', '+Y+' F='+F);
610   ctx.beginPath();
611
612   SA = (S / 180) * Math.PI;
613   EA = (E / 180) * Math.PI;
614   ROA = (RO / 180) * Math.PI;
615   ctx.ellipse(X, Y, RX, RY, ROA, SA, EA);
616
617   PX = parseInt(X);
618   PY = parseInt(Y);
619   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
620   PX -= 25;
621   PY += 40;
622   //console.log('Ellipse B '+PX+', '+PY+' F='+F);
623   ctx.lineTo(PX, PY);
624   ctx.closePath();
625
626   if( F ){
627     ctx.fill();
628   }else{
629     ctx.stroke();
630   }
631   saveDrawing();
632 }
633 function CV_drawBaloon(){
634   CV_drawBaloon1(CanvasTool_Baloon);
635 }
636 function CV_drawPackman1(circle){
637   canvas = document.getElementById('CV_partsCanvas');
638   ctx = canvas.getContext('2d');
639   ctx = CV_setCtxStyle(ctx,p);
640
641   p = circle;
642   F = childByName(p, 'F').checked;
643   X = childByName(p, 'X').value;
644   Y = childByName(p, 'Y').value;
645   Z = childByName(p, 'Z').value;
646   p.style.zIndex = Z;
647   R = childByName(p, 'R').value;
648   S = childByName(p, 'S').value;
649   E = childByName(p, 'E').value;
650
651   //console.log('Packman'+X+', '+Y+' F='+F);
652   ctx.beginPath();
653   SA = (S / 180) * Math.PI;
654   //SA += 0.15 * Math.PI;
655   EA = SA + Math.PI; // (E / 180) * Math.PI;
656   ctx.arc(X, Y, R, SA, EA);
657   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
658
659   ctx.beginPath();
660   E = 180 - E;
661   SA += (E/180) * Math.PI;
662   EA += (E/180) * Math.PI;
663   ctx.arc(X, Y, R, SA, EA);
664   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
665
666   saveDrawing();
667 }
668 function CV_drawPackman(){
669   CV_drawPackman1(CanvasTool_Packman);
670 }
671 function CV_drawPiechart1(baloon){
672   canvas = document.getElementById('CV_partsCanvas');
673   ctx = canvas.getContext('2d');
674   ctx = CV_setCtxStyle(ctx,p);
675
676   p = baloon;
677   F = childByName(p, 'F').checked;
678   X = childByName(p, 'X').value;
679   Y = childByName(p, 'Y').value;
680   Z = childByName(p, 'Z').value;
681   RX = childByName(p, 'RX').value;
682   RY = childByName(p, 'RY').value;
683   p.style.zIndex = Z;
684   RO = childByName(p, 'RO').value;
685   S = childByName(p, 'S').value;
686   E = childByName(p, 'E').value;
687
688   //console.log('Ellipse'+X+', '+Y+' F='+F);
689   ctx.beginPath();
690
691   SA = (S / 180) * Math.PI;
692   EA = (E / 180) * Math.PI;
693   ROA = (RO / 180) * Math.PI;
694   ctx.ellipse(X, Y, RX, RY, ROA, SA, EA);
695
696   PX = parseInt(X);
697   PY = parseInt(Y);
698   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
699   //PX -= 25;
700   //PY += 40;
701   //console.log('Ellipse B '+PX+', '+PY+' F='+F);
702   ctx.lineTo(PX, PY);
703   ctx.closePath();
704
705   if( F ){
706     ctx.fill();
707   }else{

```

```

708     ctx.stroke();
709 }
710 saveDrawing();
711 }
712 function CV_drawPiechart(){
713     CV_drawPiechart1(CanvasTool_Piechart);
714 }
715
716 function CV_drawXArc1(baloon){
717     canvas = document.getElementById('CV_partsCanvas');
718     ctx = canvas.getContext('2d');
719     ctx = CV_setCtxStyle(ctx,p);
720
721     p = baloon;
722     F = childByName(p,'F').checked;
723     X = childByName(p,'X').value;
724     Y = childByName(p,'Y').value;
725     Z = childByName(p,'Z').value;
726     RX = childByName(p,'RX').value;
727     RY = childByName(p,'RY').value;
728     p.style.zIndex = Z;
729     RO = childByName(p,'RO').value;
730     S = childByName(p,'S').value;
731     E = childByName(p,'E').value;
732
733     //console.log('Ellipse'+X+', '+Y+' F='+F);
734     ctx.beginPath();
735
736     if( true ){
737         d = new Date();
738         ms = d.getTime();
739         id = (ms % 200) / 10;
740         //S = parseFloat(E) + id/2;
741         E = parseFloat(E) - id;
742         xd = (ms % 10000) / 5;
743         if( 1000 < xd ){
744             xd = 2000 - xd;
745         }
746         xd *= 0.8;
747         X = parseFloat(X) + xd - 600;
748         //console.log('Ellipse S='+S+', E='+E+' id='+id);
749
750         SA = (S / 180) * Math.PI;
751         EA = (E / 180) * Math.PI;
752         ROA = (RO / 180) * Math.PI;
753         ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
754     }
755
756     PX = parseInt(X);
757     PY = parseInt(Y);
758
759     //console.log('Ellipse A '+PX+', '+PY+' F='+F);
760     //console.log('Ellipse B '+PX+', '+PY+' F='+F);
761
762     ctx.lineTo(PX,PY);
763     ctx.closePath();
764
765     if( F ){
766         ctx.fill();
767     }else{
768         ctx.stroke();
769     }
770     saveDrawing();
771 }
772 function CV_drawXArc(){
773     t = event.target;
774     CV_drawXArc1(t.parentNode);
775 }
776 var AnimeIntvl = window.setInterval(CV_redrawParts,30);
777
778 </script>
779
780 <h3>Animation</h3>
781 <span id="Animation" class="CanvasTool" draggable="true" contenteditable>
782 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
783 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
784 <span data-name="evid" class="CanvasParam" type="text">ANIMA-0</span>
785 <input data-name="r" class="ColorParam" type="text" value="rotate">
786 <input data-name="T" class="CanvasParam" type="text" value="30" onwheel="OnWheelInt()">ms
787 <input data-name="T" class="CanvasParam" type="text" value="10" onwheel="OnWheelInt()">s
788 </span>
789
790 <h3>Canvas</h3>
791 <span id="AppendToCanvas" class="CanvasTool" draggable="true">
792 <input class="CV_Button" type="button" value="Append"> the above part
793 </span>
794 <span id="CanvasWrapTemplate" class="CanvasWrap" draggable="true">
795 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
796 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
797 <span data-name="evid" class="CanvasParam" type="text">0</span>
798 <input data-name="width" class="CanvasParam" type="text" onchange="CS_setSize()" value="700">
799 <input data-name="height" class="CanvasParam" type="text" onchange="CS_setSize()" value="400">
800 <input data-name="zoom" class="CanvasParam" type="text" onchange="CS_setSize()" value="100" onwheel="OnWheelCanvasZoom()">
801 <input class="CV_Button" type="button" value="Remove" onclick="removeParent()"><br>
802 <canvas id="DrawingCanvas" data-name="canvas" class="CS_Canvas" width="700" height="400"></canvas>
803 </span>
804 <script>
805 function OnWheelCanvasZoom(){
806     val = OnWheelInt();
807     DrawingCanvas.style.zoom = val + '%';
808     //CanvasWrapTemplate.style.width = (700*(val/100.0)+20) + 'px';
809     CanvasWrapTemplate.style.height = (400*(val/100.0)+30) + 'px';
810 }
811 </script>
812
813 <h3>CanvasBook</h3>
814 <div id="CanvasBook"></div>
815 <br>
816 <style>
817 .CanvasBook {
818     overflow:scroll;
819 }
820 .CBPanel {
821 }
822 .CanvasTool {
823     font-size:9pt;
824     font-family:Courier New;
825     color:#000 !important;
826     xborder:1px solid #aaf;
827     background-color:rgba(127,127,127,0.5);
828     width:740px;
829     white-space:nowrap;
830     xxheight:30;
831     margin:4px;
832     padding-left:4px;
833     padding-right:4px;
834     overflow:auto;
835     display:inline-block;
836     resize:both;
837     vertical-align:top;
838     zoom:1.0;
839 }
840 .CanvasWrap {
841     font-size:9pt;
842     font-family:Courier New;
843     color:#000 !important;
844     border:1px solid #aaf;
845     background-color:rgba(200,200,200,0.2);
846     width:740px;
847     height:430px;
848     margin:4px;
849     padding-left:4px;
850     padding-right:4px;
851     overflow:auto;
852     display:inline-block;
853     resize:both;
854     vertical-align:top;
855     zoom:1.0;
856 }
857 .CS_Panel {
858     padding:3px;
859     vertical-align:middle;
860 }
861 .CS_Canvas {
862     border:1px dashed #fcc;
863     resize:both;
864     zoom:1.0;
865 }
866 </style>
867 <script>
868 var CanvasID = 0;
869 function removeParent(){
870     e = event.target;
871     p = e.parentNode;
872     if( p == CanvasWrapTemplate ){
873         return;
874     }
875     pp = p.parentNode;
876     alert('removeParent #' +pp.id+'/' +p.id+'/' +e.id);
877     p.parentNode.removeChild(p);
878 }
879 function cloneParent(){
880     b = event.target;
881     w = b.parentNode;
882     CanvasID += 1;
883     //pp = w.parentNode;
884     cw = w.cloneNode(true);

```

```

885   cw.id = 'Canvas_'+CanvasID;
886   childByName(cw, 'cvid').innerHTML = CanvasID;
887   childByName(cw, 'cvid').value = CanvasID;
888   CanvasBook.appendChild(cw);
889 }
890 function CS_resize(){
891   e = event.target;
892   p = e.parentNode;
893   console.log('resize '+e.nodeName+' '+p.nodeName);
894   c = childByName(p, 'canvas');
895   w = childByName(p, 'width').value;
896   h = childByName(p, 'height').value;
897   console.log('resize '+c.nodeName+' '+w+', '+h);
898   c.width = w;
899   c.height = h;
900   p.style.width = w + 'px';
901   p.style.height = h + 'px';
902   console.log("c="+c+' '+w+'/'+'+c.width+', '+h+'/'+'+c.height);
903 }
904 function CS_newFunc(){
905   cvt = CanvasWrapTemplate;
906   cw = CanvasWrapTemplate.cloneNode(true); //needs an argument, otherwise 'function not found'
907   CanvasID += 1;
908   cw.id = 'Canvas_' + CanvasID;
909   //childByName(cw, 'cvid').contentEditable = false;
910   childByName(cw, 'cvid').innerHTML = CanvasID;
911   childByName(cw, 'cvid').value = CanvasID;
912   childByName(cw, 'width').value = w = childByName(cvt, 'width').value;
913   childByName(cw, 'height').value = h = childByName(cvt, 'height').value;
914   cw.style.width = w + 'px';
915   //ncv = document.createElement('canvas');
916   ncv = childByName(cw, 'canvas');
917   //ncv.setAttribute('class', 'CS_Canvas');
918   //ncv.setAttribute('data-name', 'canvas');
919   ncv.width = w;
920   ncv.height = h;
921   //cw.replaceChild(ncv, childByName(cw, 'canvas'));
922   CanvasBook.appendChild(cw);
923 }
924 //CS_new.addEventListener('click', CS_newFunc);
925 </script>
926
927 <input id="CanvasBook_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
928 <input id="CanvasBook_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
929 <input id="CanvasBook_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
930 <span id="CanvasBook_WorkCodeView"></span>
931 <script id="CanvasBook_WorkScript">
932 function CanvasBook_openWorkCodeView(){
933   function CanvasBook_showWorkCode(){
934     showHtmlCode(CanvasBook_WorkCodeView, CascadedCanvasBook_WorkCodeSpan);
935   }
936   CanvasBook_WorkCodeViewOpen.addEventListener('click', CanvasBook_showWorkCode);
937 }
938 CanvasBook_openWorkCodeView(); // should be invoked by an event
939 </script>
940 </details>
941 <!-- CanvasBook_WorkCodeSpan -->
942 *//</span>
943 //<!-- Work } ----- -->
944
945
946
947
948
949 <!-- Work { ----- -->
950 <span id="Topbar_WorkCodeSpan">
951 /*
952 <details><summary>Topbar</summary>
953 <!-- Topbar // 2020-1008 SatoxITS -->
954 <h2>Topbar</h2>
955 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
956 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
957 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
958 <span id="Topbar_WorkCodeView"></span>
959 </details>
960
961 <style>
962 #GshHeading {
963   display:inline;
964   overflow:visible;
965 }
966 .ConfigIcon {
967   position:absolute;
968   top:-6px;
969   left:92%;
970   width:32px;
971   height:32px;
972 }
973 .MetaWindow {
974   z-index:1000;
975   position:relative;
976   display:block;
977   overflow:visible !important;
978   width:99.9%;
979   height:22px;
980   top:-22px;
981   border:1px solid #22a;
982   margin:0px;
983   left:0.0%;
984   line-height:1.0;
985   font-family:Georgia;
986   color:#ff;
987   font-size:12pt;
988   text-align:center;
989   vertical-align:middle;
990   padding:4px;
991   xbackground-color:rgba(0,8,170,0.8);
992   background-color:#3a4861;xxx-PBlue;
993   vertical-align:middle;
994 }
995 .MetaWindow:hover {
996   color:#000;
997   border:1px solid #22a;
998   background-color:rgba(255,255,255,1.0);
999 }
1000 #GshBanner {
1001   overflow:visible;
1002   display:block;
1003   width:100%;
1004   height:100px;
1005   left:inherit !important;
1006 }
1007 </style>
1008 <script>
1009 function Topbar_openWorkCodeView(){
1010   function Topbar_showWorkCode(){
1011     showHtmlCode(Topbar_WorkCodeView, Topbar_WorkCodeSpan);
1012   }
1013   Topbar_WorkCodeViewOpen.addEventListener('click', Topbar_showWorkCode);
1014 }
1015 Topbar_openWorkCodeView();
1016 function ConfigClick(){
1017   if( 0 <= AffView.style.zIndex ){
1018     AffView.style.saved_zIndex = AffView.style.zIndex;
1019     AffView.style.zIndex = -1000;
1020     GshSidebar.style.zIndex = -1;
1021     GshPerfMon.style.zIndex = -1;
1022   }else{
1023     //AffView.style.zIndex = AffView.style.saved_zIndex;
1024     AffView.style.zIndex = 1;
1025     GshSidebar.style.zIndex = 1;
1026     GshPerfMon.style.zIndex = 1;
1027     GMenu.style.zIndex = 10000000;
1028   }
1029   console.log('AffZidex="+AffView.style.zIndex);
1030 }
1031 function Gshell_initTopbar(){
1032   GshTopbar.innerHTML = GshTitle.innerHTML;
1033   <!--<img id="ConfigIcon" class="ConfigIcon">
1034   if( true ){
1035     cfig = document.createElement('img');
1036     cfig.id = 'ConfigIcon';
1037     cfig.setAttribute('class', 'ConfigIcon');
1038     GshTopbar.appendChild(cfig);
1039     cfig.src = ConfigICON_DATA;
1040
1041     //cfig.style.zIndex = 10000000000;
1042     //cfig.addEventListener('click', ConfigClick);
1043     GshTopbar.addEventListener('click', ConfigClick);
1044   }
1045 }
1046 </script>
1047 <!-- Topbar_WorkCodeSpan -->
1048 *//</span>
1049 //<!-- Work } ----- -->
1050
1051
1052 <!-- Work { ----- -->
1053 <span id="Indexer_WorkCodeSpan">
1054 /*
1055 <details><summary>Indexer</summary>
1056 <!-- Indexer // 2020-1007 SatoxITS -->
1057 <h2>Indexer</h2>
1058 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1059 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1060 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1061 <span id="Indexer_WorkCodeView"></span>

```

```

1062 </details>
1063 <style id="SidebarIndex">
1064 #gsh {
1065     display:block;
1066     xxoverflow:scroll !important;
1067 }
1068 #GshMain {
1069     z-index:1;
1070     position:relative;
1071     display:block;
1072     width:80% !important;
1073     left:19.5% !important;
1074 }
1075 #GshSidebar {
1076     z-index:0;
1077     position:relative !important;
1078     overflow:auto;
1079     resize:both !important;
1080     xxoverflow-y:hidden !important;
1081     xxheight:100px !important;
1082     xxdisplay:inline !important;
1083     left:0px;
1084     top:0px;
1085     width:19.5%;
1086     min-width:80px;
1087     xxheight:100% !important;
1088     height:0px;
1089     color:#f00;
1090     xxbackground-color:rgba(64,64,64,0.5);
1091     xxbackground-color:#DFE3EB;xxx-PBlue;
1092     background-color:#eeeeee;xxx-PBlue;
1093 }
1094 #GshPerfMon {
1095     position:relative;
1096     display:block;
1097     overflow:visible;
1098     z-index:0 !important;
1099     xxheight:12pt;
1100     font-family:monospace, Courier New !important;
1101     font-size:9pt !important;
1102     color:#f84;
1103     top:-20px;
1104 }
1105 #GshPerfMon:hover {
1106     z-index:3 !important;
1107 }
1108 #GshSidebar:hover {
1109     z-index:2;
1110     overflow:xxvisible !important;
1111     background-color:rgba(255,255,255,0.7);
1112     width:50%;
1113 }
1114 #GshIndexer {
1115     z-index:0;
1116     position:relative;
1117     resize:both !important;
1118     height:100%;
1119     left:0px;
1120     top:0px;
1121     scroll-behavior: overflow !important;
1122     padding-left:4pt;
1123     font-size:0.5em;
1124     white-space:nowrap;
1125     xxx-background-color:rgba(64,160,64,0.6) !important;
1126     color:#7794c6;xxx-PBlue;
1127     xxbackground-color:#DFE3EB;xxx-PBlue;
1128     background-color:#eeeeee;xxx-PBlue;
1129 }
1130 #GshIndexer:hover {
1131     z-index:1000000;
1132     overflow:xxvisible !important;
1133     color:#000000 !important;xxx-PBlue;
1134     xxxbackground-color:#FFFFFF;xxx-PBlue;
1135     background-color:rgba(255,255,255,0.7);
1136     padding-right:0px;
1137     width:80%;
1138 }
1139 #GshIndexer:select {
1140     color:#000000 !important;xxx-PBlue;
1141     background-color:#FFFFFF;xxx-PBlue;
1142 }
1143 .IndexLine {
1144     font-size:8pt !important;
1145     font-family:Georgia;
1146     display:block;
1147     xxx-color:#fff;
1148     xxx-color:#efffE5;xxx-PBlue;
1149     xxx-color:#41516d;xxx-PBlue;
1150     xxx-color:#7794c6;xxx-PBlue;
1151     padding-right:4pt;
1152 }
1153 .IndexLine:hover {
1154     font-size:10pt !important;
1155     xxx-color:#228;
1156     xxx-background-color:#fff;
1157     xxxcolor:#fff;xxx-PBlue;
1158     color:#516487;xxx-PBlue;
1159     background-color:rgba(220,220,255,1.0);xxx-PBlue;
1160     xxtext-shadow:1px 1px #e3f;
1161     text-shadow:1px 1px #eee;
1162     xxxbackground-color:#516487;xxx-PBlue;
1163     xxtext-decoration:underline !important;
1164 }
1165 </style>
1166
1167 <script id="Indexer WorkScript">
1168 function Indexer_openWorkCodeView(){
1169     function Indexer_showWorkCode(){
1170         showHtmlCode(Indexer_WorkCodeView, Indexer_WorkCodeSpan);
1171     }
1172     Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
1173 }
1174 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
1175 Indexer_openWorkCodeView();
1176
1177 var startPerfDate = new Date();
1178 var prevPerfDate = startPerfDate;
1179 function ShowResourceUsage(){
1180     d = new Date();
1181     perf = '';
1182     perf += '<'+font color="gray">UA:' + window.navigator.userAgent + '<'+font><br>\n';
1183     perf += DateShort(0(startPerfDate)) + '<br>\n';
1184     perf += DateShort() + '<br>\n';
1185     elps = d.getTime() - startPerfDate.getTime();
1186     itvl = d.getTime() - prevPerfDate.getTime();
1187     perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
1188     perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
1189     prevPerfDate = d;
1190
1191     if( performance.memory != undefined ){
1192         m0 = performance.memory;
1193         mem (m0.usedHeapSize / 1000000.0); // .toFixed(6);
1194         perf += 'Memory: '+mu0+' MB<br>\n';
1195     }
1196     perf += '<br>\n';
1197
1198     //GshSidebar.innerHTML = perf;
1199     GshPerfMon.innerHTML = perf;
1200     //GshIndexer.innerHTML = 'Memory: '+mu0+' MB';
1201     //console.log('-- PerfMon heap: '+mu0+'/' +m0.totalHeapSize+'/' +m0.jsHeapSizeLimit);
1202     if( true ){
1203         GshSidebar.style.zIndex = 1000;
1204         GshIndexer.style.zIndex = 0;
1205         GshPerfMon.style.zIndex = 1;
1206         //GshSidebar.appendChild(GshPerfMon);
1207         if (document.getElementById('primary') == null) { // not in WordPress
1208             GshPerfMon.style.position = 'absolute';
1209         }
1210         GshPerfMon.style.display = 'block';
1211         GshPerfMon.style.marginLeft = '4px';
1212         //GshPerfMon.style.top = '45px';
1213         GshPerfMon.style.position = 'relative';
1214         //GshPerfMon.style.position = 'absolute';
1215         //topy = GshTopbar.getBoudingClientRect().top;
1216         //topy = parseInt(topy) + 40;
1217         //GshPerfMon.style.top = topy + 'px';
1218         GshPerfMon.style.left = '0px';
1219
1220         GshMain.style.top = -GshPerfMon.getBoudingClientRect().height;
1221     }
1222 }
1223 function ResetPerfMon(){
1224     GshPerfMon.removeAttribute('style');
1225     GshSidebar.removeAttribute('style');
1226 }
1227
1228 var iserno = 0;
1229 var GeneratedId = 0;
1230 function generateIndex(ni,e,chnv,nch,ht){
1231     // https://developer.mozilla.org/en-US/docs/Web/API/Element
1232     c = '';
1233     if (e.classList != null) {
1234         c = e.classList.value;
1235     }
1236     //console.log('-- '<+e.nodeName+'> #' +e.id+' .' +c+' '+e.attributes);
1237     if (e.nodeName == '#text') { return ''; }
1238     if (e.nodeName == '#comment') { return ''; }

```

```

1239 if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
1240   id = e.innerHTML;
1241   Generatedid += 1;
1242   eid = 'GeneratedId-'+GeneratedId;
1243   e.id = eid;
1244 }else
1245 if( e.nodeName == 'SUMMARY' ){
1246   id = e.innerHTML;
1247   Generatedid += 1;
1248   eid = 'GeneratedId-'+GeneratedId;
1249   e.id = eid;
1250 }else
1251 if( and(e.nodeName == 'DIV',c='xxxxxxxxxxxxxxxxentry-content') ){
1252   console.log("-- DIV entry-content begin");
1253   id = e.innerHTML;
1254   Generatedid += 1;
1255   eid = 'GeneratedId-'+GeneratedId;
1256   e.id = eid;
1257   console.log("-- DIV entry-content end hash-child=",e.hasChildNodes());
1258 }else
1259 if( e.id == '' || e.id == 'undefined' ){
1260   return '';
1261 }else{
1262   id = '#'+e.id;
1263   e.id = e.id;
1264 }
1265 iserno += 1;
1266 ht = '<div id="GeneratedEref'+iserno+'" class="IndexLine" href="'+eid+'>'
1267   + iserno+' '+ni+' '+e.nodeName + ':' + id;
1268 if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+'/div>'; }
1269 if( !e.hasChildNodes() ){ return ht + '<'+'/div>'; }
1270 chv = e.childNodes;
1271 nch = e.childNodes.length;
1272 if( chv != null ){ nch = chv.length; }
1273 ht += ' ('+nch+')' + '<'+'/div>';
1274 for( let i = 0; i < chv.length; i++){
1275   sec = ni+'.'+i;
1276   if( ni == '' ){ sec = i; }
1277   ht += generateIndex(sec,chv[i],null,0);
1278 }
1279 return ht;
1280 }
1281 function onClickIndex(e){
1282   tid = e.target.id;
1283   tge = document.getElementById(tid);
1284   eid = tge.getAttribute( 'href' );
1285   rx = tge.getBoundingClientRect().left.toFixed(0)
1286   ry = tge.getBoundingClientRect().top.toFixed(0)
1287   if( false ){
1288     alert( 'index clicked mouse(x='+e.x+', y='+e.y+')'
1289       + '\ntid=#'+ tid + ' rx='+rx + ',ry='+ry
1290       + '\neid=' + eid + '\n'
1291       + '\nhtml='+ tge.outerHTML);
1292   }
1293   ee = document.getElementById(eid);
1294   sx = 'xah;
1295   sy = e.getBoundingClientRect().top;
1296   console.log('sx'+sx+',sy'+sy);
1297   ee.scrollIntoView()
1298   window.scrollTo(sx,sy)
1299   //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
1300 }
1301 function Indexer_afterLoaded(){
1302   sideindex = document.getElementById('GshIndexer');
1303   ht = '<'+h3>G-Index<'+/h3>';
1304   ht += generateIndex("",document.getElementById('gsh'),null,0,'');
1305   if( (pri = document.getElementById('primary')) != null ){
1306     ht += generateIndex("",pri,null,0,'');
1307   }
1308   ht += '<'+<br>';
1309   ht += '<'+<br>';
1310   ht += '<'+<br>';
1311   ht += '<'+<br>';
1312   sideindex.innerHTML = ht;
1313   sideindex.addEventListener('click',onClickIndex);
1314 }
1315 if( (pri = document.getElementById('primary')) != null ){
1316   console.log("-- Seems in WordPress");
1317   pri.style.zIndex = 2000;
1318 }
1319 GshSidebar.style.setProperty('position','relative','important');
1320 GshSidebar.style.top = '-140px';
1321 //GshSidebar.style.setProperty('position','absolute','important');
1322 //GshSidebar.style.top = '0px';
1323 }
1324 GshSidebar.style.setProperty('width','200px','important');
1325 GshSidebar.style.setProperty('overflow','scroll','important');
1326 GshSidebar.style.resize = 'both';
1327 }
1328 GshSidebar.style.left = '-100px';
1329 GshIndexer.style.left = '100px';
1330 GshIndexer.style.height = '140px';
1331 gsh.appendChild(GshSidebar); // change parent
1332 }else{
1333   console.log("-- Seems not in WordPress");
1334   GshSidebar.style.setProperty('position','fixed','important');
1335 }
1336 }
1337 //document.addEventListener('load',Indexer_afterLoaded);
1338 }
1339 DestroyIndexBar = function(){
1340   sideindex = document.getElementById('GshIndexer');
1341   sideindex.innerHTML = "";
1342   sideindex.style = "";
1343 }
1344 </script>
1345
1346 <!-- Indexer_WorkCodeSpan -->
1347 *! //<span>
1348 //<!-- ----- Work ----- -->
1349
1350
1351 /*
1352 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
1353 <p>
1354 <note>
1355 It is a shell for myself, by myself, of myself. --SatoxITS(^-^ )
1356 <a href="gsh-0.6.2.go.html">prev.</a>
1357 </note>
1358 </p>
1359 <div id="GJFactory_x"></div>
1360
1361 <div>
1362 <span id="GshMenu" class="GshMenu">
1363 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
1364 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
1365 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
1366 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
1367 <span id="gsh-WinId" onclick="win_jump('0.1');">></span>
1368 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
1369 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
1370 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
1371 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
1372 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
1373 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="">Source</span>
1374 <!-- /<span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span> -->
1375 </span>
1376 </div>
1377 *!
1378
1379 /*
1380 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
1381 <h3>Fun to create a shell</h3>
1382 <p>For a programmer, it must be far easy and fun to create his own simple shell
1383 rightly fitting to his favor and necessities, than learning existing shells with
1384 complex full features that he never use.
1385 I, as one of programmers, am writing this tiny shell for my own real needs,
1386 totally from scratch, with fun.
1387 </p><p>
1388 For a programmer, it is fun to learn new computer languages. For long years before
1389 writing this software, I had been specialized to C and early HTML2 :-).
1390 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
1391 on demand as a novice of these, with fun.
1392 </p><p>
1393 This single file "gsh.go", that is executable by Go, contains all of the code written
1394 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
1395 HTML file that works as the viewer of the code of itself, and as the "home page" of
1396 this software.
1397 </p><p>
1398 Because this HTML file is a Go program, you may run it as a real shell program
1399 on your computer.
1400 But you must be aware that this program is written under situation like above.
1401 Needless to say, there is no warranty for this program in any means.
1402 </p>
1403 <address>Aug 2020, SatoxITS (sato8bits-more.jp)</address>
1404 </details>
1405 *!
1406 /*
1407 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
1408 </p>
1409 <h3>Cross-browser communication</h3>
1410 <p>
1411 ... to be written ...
1412 </p>
1413 <h3>Vi compatible command line editor</h3>
1414 <p>
1415

```



```

1416 The command line of GShell can be edited with commands compatible with
1417 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
1418 As in vi you can enter <b>command modes</b> by <b>ESC</b> key,
1419 then move around in the history by <b>code>j k / ? n N</code></b>,
1420 or within the current line by <b>code>l h e w b o $ </code></b> or so.
1421 </p>
1422 </details>
1423 */
1424 */
1425 <details id="gsh-gindex">
1426 <summary>Index</summary><div class="gsh-src">
1427 Documents
1428 <span class="gsh-link" onclick="jumpTo_JavaScriptView();">Command summary</span>
1429 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
1430 Package structures
1431 <a href="#import">import</a>
1432 <a href="#struct">struct</a>
1433 Main functions
1434 <a href="#comexpansion">str-expansion</a> // macro processor
1435 <a href="#finder">finder</a> // builtin find + du
1436 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
1437 <a href="#plugin">plugin</a> // plugin commands
1438 <a href="#ex-commands">system</a> // external commands
1439 <a href="#builtin">builtin</a> // builtin commands
1440 <a href="#netio">network</a> // socket handler
1441 <a href="#remote-sh">remote-sh</a> // remote shell
1442 <a href="#redirect">redirect</a> // Stdin/Out redirection
1443 <a href="#history">history</a> // command history
1444 <a href="#rusage">rusage</a> // resource usage
1445 <a href="#encode">encode</a> // encode / decode
1446 <a href="#img">IMG</a> // command line IMG
1447 <a href="#editing">getline</a> // line editor
1448 <a href="#scanf">scanf</a> // string decomposer
1449 <a href="#interpreter">interpreter</a> // command interpreter
1450 <a href="#main">main</a>
1451 </span>
1452 JavaScript part
1453 <a href="#script-src-view" class="gsh-link" onclick="jumpTo_JavaScriptView();">Source</a>
1454 <a href="#gsh-data-frag" class="gsh-link" onclick="jumpTo_DataView();">Builtin data</a>
1455 CSS part
1456 <a href="#style-src-view" class="gsh-link" onclick="jumpTo_StyleView();">Source</a>
1457 References
1458 <a href="#" class="gsh-link" onclick="jumpTo_WholeView();">Internal</a>
1459 <a href="#gsh-reference" class="gsh-link" onclick="jumpTo_ReferenceView();">External</a>
1460 Whole parts
1461 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Source</a>
1462 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Download</a>
1463 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Dump</a>
1464 </div>
1465 </details>
1466 */
1467 </<details id="gsh-gocode">
1468 </<summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
1470 // gsh - Go lang based Shell
1471 // (c) 2020 ITS more Co., Ltd.
1472 // 2020-0807 created by SatoxITS (sato@its-more.jp)
1473
1474 package main // gsh main
1475
1476 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
1477 import (
1478     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
1479     "errors" // <a href="https://golang.org/pkg/errors/">errors</a>
1480     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
1481     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
1482     "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
1483     "time" // <a href="https://golang.org/pkg/time/">time</a>
1484     "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
1485     "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
1486     "os" // <a href="https://golang.org/pkg/os/">os</a>
1487     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
1488     "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
1489     "net" // <a href="https://golang.org/pkg/net/">net</a>
1490     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
1491     "html" // <a href="https://golang.org/pkg/html/">html</a>
1492     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
1493     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
1494     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
1495     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
1496     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
1497     "gobinary" // gshell's logo and source code
1498     "hash/crc32" // <a href="https://golang.org/pkg/hash/crc32/">crc32</a>
1499     "golang.org/x/net/websocket"
1500     "runtime"
1501 )
1502
1503 /*
1504 #include <stdio.h> // </stdio.h> to be closed as HTML tag :-p
1505 #ifdef _WIN32
1506 #include <windows.h> // </windows.h>
1507 // 2020-1022 added -- terminal mode on Windows
1508 // https://docs.microsoft.com/en-us/windows/console/setconsolemode
1509 // https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
1510 int setTermMode(){
1511     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
1512     DWORD tmode = 0;
1513     if( GetConsoleMode(hStdin,&tmode) ){
1514         DWORD xmode = tmode;
1515         xmode &= ~ENABLE_ECHO_INPUT;
1516         xmode &= ~ENABLE_LINE_INPUT;
1517         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
1518         if( SetConsoleMode(hStdin,xmode) ){
1519             return tmode;
1520         }
1521     }
1522     return 0;
1523 }
1524 #endif
1525 int setTermMode(int tmode){
1526     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
1527     SetConsoleMode(hStdin,tmode);
1528     return 0;
1529 }
1530 #else
1531 int setTermRaw(){
1532     return -1;
1533 }
1534 int setTermMode(int tmode){
1535     return 0;
1536 }
1537 #endif
1538 */
1539 import "C"
1540
1541 /*
1542 // // 2020-0906 added,
1543 // // <a href="https://golang.org/cmd/cgo/">CGo</a>
1544 // #include "poll.h" <poll.h> // </poll.h> to be closed as HTML tag :-p
1545 // typedef struct { struct pollfd fdv[8]; } pollFdv;
1546 // int pollx(pollFdv *fdv, int nfdv, int timeout){
1547 //     return poll(&fdv->fdv,nfdv,timeout);
1548 // }
1549 import "C"
1550
1551 // 2020-1021 replaced poll() with channel/select
1552 // // 2020-0906 added,
1553 func CFpollInl(fp*os.File, timeoutUs int)(ready uintptr){
1554     var fdv = C.pollFdv{
1555         var nfdv = 1
1556         var timeout = timeoutUs/1000
1557
1558         fdv.fdv[0].fd = C.int(fp.Fd())
1559         fdv.fdv[0].events = C.POLLIN
1560         if( 0 < EventRecvFd {
1561             fdv.fdv[1].fd = C.int(EventRecvFd)
1562             fdv.fdv[1].events = C.POLLIN
1563             nfdv += 1
1564         }
1565         r := C.pollx(&fdv,C.int(nfdv),C.int(timeout))
1566         if( r <= 0 ){
1567             return 0
1568         }
1569         if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
1570             //fprintf(stderr,"--De-- got Event\n");
1571             return uintptr(EventFdOffset + fdv.fdv[1].fd)
1572         }
1573         if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
1574             return uintptr(NormalFdOffset + fdv.fdv[0].fd)
1575         }
1576         return 0
1577     }
1578 */
1579
1580 const (
1581     NAME = "gsh"
1582     VERSION = "0.7.8"
1583     DATE = "2020-11-01"
1584     AUTHOR = "SatoxITS(~)://"
1585 )
1586 var (
1587     GSH_HOME = ".gsh" // under home directory
1588     GSH_PORT = 9999
1589     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
1590     PROMPT = ">"
1591     LINESIZE = (8*1024)
1592     PATHSEP = ":" // should be ";" in Windows

```

```

1593 DIRSEP = "/" // canbe \ in Windows
1594 OnWindows = false;
1595 }
1596 func initGshEnv(){
1597 if( runtime.GOOS == "windows" ){
1598     PARSEP = ";";
1599     DIRSEP = "\";
1600     OnWindows = true;
1601 }else{
1602 }
1603 }
1604
1605 // --X logging control
1606 // --A-- all
1607 // --I-- info.
1608 // --D-- debug
1609 // --T-- time and resource usage
1610 // --W-- warning
1611 // --E-- error
1612 // --F-- fatal error
1613 // --Xn- network
1614
1615 // <a name="struct">Structures</a>
1616
1617 // 2020-1022 Unix/Windows
1618 // -----
1619 //type aStat_t syscall.Stat_t;
1620 //type aStat_t struct { syscall.Stat_t }
1621 type aStat_t struct {
1622     Size      int64
1623     Mode      os.FileMode
1624     Rdev      int64
1625     Blocks    int64
1626     Nlink     int64
1627 }
1628 func aStat(path string, astat *aStat_t)(error){
1629 /*
1630 sstat := syscall.Stat_t{};
1631 err := syscall.Stat(path,&sstat);
1632 *astat = aStat_t(sstat);
1633 */
1634 fi,err := os.Stat(path);
1635 if( err == nil ){
1636     astat.Mode = fi.Mode();
1637     astat.Size = fi.Size();
1638 }
1639 return err;
1640 }
1641
1642 func aFStat(fd int, astat *aStat_t)(error){
1643 /*
1644 sstat := syscall.Stat_t{};
1645 err := syscall.FStat(fd,&sstat);
1646 *astat = aStat_t(sstat);
1647 */
1648 err := errors.New("NotImplemented-Fstat");
1649 //fmt.Printf("----E-- fatal(%v)(%v)\n",fd,err);
1650 return err;
1651 }
1652 func aAccess(path string, mode uint32)(error){
1653 //err := syscall.Access(path,mode);
1654 //err := errors.New("NotImplemented-Access");
1655 fi,err := os.Stat(path);
1656 //fmt.Printf("----Access(%v,%v)\n(%v)\n",path,mode,err);
1657 if( err == nil ){
1658     fmode := fi.Mode();
1659     if( fmode.IsRegular() ){
1660         perm := fmode.Perm();
1661         if( (uint32(perm) & mode) != 0 ){
1662             return nil;
1663 }
1664         return errors.New("NotAccessible");
1665 }
1666         return errors.New("NotRegularFile");
1667 }
1668 return err;
1669 }
1670 }
1671 // 2020-1022 Unix/Windows
1672 // -----
1673 type aRusage struct {
1674     syscall.Rusage
1675     Utime      time.Duration
1676     Stime      time.Duration
1677     //Sys      interface{}
1678 }
1679
1680 /*
1681 const aRUSAGE_SELF = syscall.RUSAGE_SELF
1682 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
1683 */
1684 const aRUSAGE_SELF = 0
1685 const aRUSAGE_CHILDREN = 1
1686 func aGetRusage(sel int, ru *aRusage){
1687 /*
1688 sysru := syscall.Rusage{};
1689 syscall.GetRusage(sel,&sysru);
1690 ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
1691 ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
1692 */
1693 }
1694 func aSetRusage(ru *aRusage, ps *os.ProcessState){
1695 ru.Utime = ps.UserTime();
1696 ru.Stime = ps.SystemTime();
1697 }
1698 func showRusage(what string,argv []string, ru *aRusage){
1699     fmt.Printf("%s:",what);
1700     //fmt.Printf("  User=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
1701     //fmt.Printf("  Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
1702     fmt.Printf("  User=%d.%06ds",ru.Utime/1000000000,(ru.Utime/1000)%10000000);
1703     fmt.Printf("  Sys=%d.%06ds",ru.Stime/1000000000,(ru.Stime/1000)%10000000);
1704 /*
1705     fmt.Printf("  Rss=%vB",ru.Maxrss)
1706     if isin("-l",argv) {
1707         fmt.Printf("  MinFlt=%v",ru.Minflt)
1708         fmt.Printf("  MajFlt=%v",ru.Majflt)
1709         fmt.Printf("  Ixrss=%vB",ru.Ixrss)
1710         fmt.Printf("  Idrss=%vB",ru.Idrss)
1711         fmt.Printf("  Nswap=%vB",ru.Nswap)
1712         fmt.Printf("  Read=%v",ru.Inblock)
1713         fmt.Printf("  Write=%v",ru.Outblock)
1714     }
1715     fmt.Printf("  Snd=%v",ru.Msgsnd)
1716     fmt.Printf("  Rcv=%v",ru.Msgrcv)
1717     //if isin("-l",argv) {
1718         fmt.Printf("  Sig=%v",ru.Nsignals)
1719     //}
1720 */
1721     fmt.Printf("\n");
1722 }
1723 }
1724 type GCommandHistory struct {
1725     StartAt      time.Time // command line execution started at
1726     EndAt        time.Time // command line execution ended at
1727     ResCode      int // exit code of (external command)
1728     CmdError     error // error string
1729     OutData      *os.File // output of the command
1730     FoundFile    []string // output - result of ufind
1731     Rusagev      [2]aRusage // Resource consumption, CPU time or so
1732     CmdId       int // maybe with identified with arguments or impact
1733     WorkDir     string // redirection commands should not be the CmdId
1734     WorkDirX    int // working directory at start
1735     CmdLine     string // index in GChdirHistory
1736 }
1737
1738 type GChdirHistory struct {
1739     Dir          string
1740     MovedAt    time.Time
1741     CmdIndex    int
1742 }
1743 type CmdMode struct {
1744     Background bool
1745 }
1746 type Event struct {
1747     when        time.Time
1748     event       int
1749     evarg       int64
1750     CmdIndex    int
1751 }
1752 var CmdIndex int
1753 var Events []Event
1754 type PluginInfo struct {
1755     Spec        *plugin.Plugin
1756     Addr        plugin.Symbol
1757     Name        string // maybe relative
1758     Path        string // this is in Plugin but hidden
1759 }
1760 type GServer struct {
1761     host        string
1762     port        string
1763 }
1764
1765 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
1766 const { // SumType
1767     SUM_TYPES = 0x000001 // items count
1768     SUM_SIZE = 0x000002 // data length (simply added)
1769     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)

```

```

1770 SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
1771 // also envelope attributes like time stamp can be a part of digest
1772 // hashed value of sizes or mod-date of files will be useful to detect changes
1773
1774 SUM_WORDS = 0x000010 // word count is a kind of digest
1775 SUM_LINES = 0x000020 // line count is a kind of digest
1776 SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
1777
1778 SUM_SUM32_BITS = 0x000100 // the number of true bits
1779 SUM_SUM32_2BYTE = 0x000200 // 16bits words
1780 SUM_SUM32_4BYTE = 0x000400 // 32bits words
1781 SUM_SUM32_8BYTE = 0x000800 // 64bits words
1782
1783 SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
1784 SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
1785 SUM_UNIXFILE = 0x004000
1786 SUM_CRCIEEE = 0x008000
1787 }
1788 type CheckSum struct {
1789     Files      int64 // the number of files (or data)
1790     Size       int64 // content size
1791     Words      int64 // word count
1792     Lines      int64 // line count
1793     SumType    int
1794     Sum64      uint64
1795     Crc32Table crc32.Table
1796     Crc32Val   uint32
1797     Sum16      int
1798     Ctime      time.Time
1799     Atime      time.Time
1800     Mtime      time.Time
1801     Start      time.Time
1802     Done       time.Time
1803     RusgAtStart [2]aRusage
1804     RusgAtEnd   [2]aRusage
1805 }
1806 type ValueStack [][]string
1807 type GshContext struct {
1808     StartDir string // the current directory at the start
1809     GetLine  string // gsh-getline command as a input line editor
1810     ChdirHistory []GchdirHistory // the 1st entry is wd at the start
1811     //gshPA      syscall.ProcAttr
1812     gshPA      os.ProcAttr
1813     CommandHistory []GCommandHistory
1814     CmdCurrent    GCommandHistory
1815     Background bool
1816     BackgroundJobs []os.ProcessState; [][]int
1817     LastRusage      aRusage
1818     GshHomeDir      string
1819     TerminalId      int
1820     CmdTrace        bool // should be [map]
1821     CmdTime         bool // should be [map]
1822     PluginFuncs    []PluginInfo
1823     iValues         []string
1824     iDelimiter      string // field separator of print out
1825     iFormat         string // default print format (of integer)
1826     iValStack       ValueStack
1827     LastServer      GServer
1828     RSERVER         string // [gsh://host[:port]
1829     RWD             string // remote (target, there) working directory
1830     lastCheckSum   CheckSum
1831 }
1832
1833 func nsleep(ns time.Duration){
1834     time.Sleep(ns)
1835 }
1836 func usleep(ns time.Duration){
1837     nsleep(ns*1000)
1838 }
1839 func msleep(ns time.Duration){
1840     nsleep(ns*1000000)
1841 }
1842 func sleep(ns time.Duration){
1843     nsleep(ns*1000000000)
1844 }
1845
1846 func strBegins(str, pat string)(bool){
1847     if len(pat) <= len(str){
1848         yes := str[0:len(pat)] == pat
1849         //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat, yes)
1850         return yes
1851     }
1852     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
1853     return false
1854 }
1855 func isin(what string, list []string) bool {
1856     for _, v := range list {
1857         if v == what {
1858             return true
1859         }
1860     }
1861     return false
1862 }
1863 func isinX(what string, list []string)(int){
1864     for i, v := range list {
1865         if v == what {
1866             return i
1867         }
1868     }
1869     return -1
1870 }
1871
1872 func env(opts []string) {
1873     env := os.Environ()
1874     if isin("-s", opts){
1875         sort.Slice(env, func(i, j int) bool {
1876             return env[i] < env[j]
1877         })
1878     }
1879     for _, v := range env {
1880         fmt.Printf("%v\n",v)
1881     }
1882 }
1883
1884 // - rewriting should be context dependent
1885 // - should postpone until the real point of evaluation
1886 // - should rewrite only known notation of symbol
1887 func scanInt(str string)(val int, leng int){
1888     leng = -1
1889     for i, ch := range str {
1890         if '0' <= ch && ch <= '9' {
1891             leng = i+1
1892         }else{
1893             break
1894         }
1895     }
1896     if 0 < leng {
1897         ival, _ := strconv.Atoi(str[0:leng])
1898         return ival, leng
1899     }else{
1900         return 0, 0
1901     }
1902 }
1903 func substHistory(gshCtx *GshContext, str string, i int, rstr string)(leng int, rst string){
1904     if len(str[i+1:]) == 0 {
1905         return 0, rstr
1906     }
1907     hi := 0
1908     histlen := len(gshCtx.CommandHistory)
1909     if str[i+1] == '!' {
1910         hi = histlen - 1
1911         leng = 1
1912     }else{
1913         hi, leng = scanInt(str[i+1:])
1914         if leng == 0 {
1915             return 0, rstr
1916         }
1917         if hi < 0 {
1918             hi = histlen + hi
1919         }
1920     }
1921     if 0 <= hi && hi < histlen {
1922         var ext byte
1923         if 1 < len(str[i+leng:]){
1924             ext = str[i+leng:][1]
1925         }
1926         //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng:])
1927         if ext == 'f' {
1928             leng = 1
1929             xlist := []string{}
1930             list := gshCtx.CommandHistory[hi].FoundFile
1931             for _, v := range list {
1932                 //list[i] = escapeWhiteSP(v)
1933                 xlist = append(xlist, escapeWhiteSP(v))
1934             }
1935             //rstr += strings.Join(list, " ")
1936             rstr += strings.Join(xlist, " ")
1937         }else{
1938             if ext == '0' || ext == 'd' {
1939                 // !NE . workdir at the start of the command
1940                 leng = 1
1941                 rstr += gshCtx.CommandHistory[hi].WorkDir
1942             }else{
1943                 rstr += gshCtx.CommandHistory[hi].CmdLine
1944             }
1945         }else{
1946             leng = 0

```

```

1947     }
1948     return leng,rstr
1949 }
1950 func escapeWhiteSP(str string)(string){
1951     if len(str) == 0 {
1952         return "\\z" // empty, to be ignored
1953     }
1954     rstr := ""
1955     for _,ch := range str {
1956         switch ch {
1957             case '\\': rstr += "\\\\"
1958             case ' ': rstr += "\\s"
1959             case '\t': rstr += "\\t"
1960             case '\r': rstr += "\\r"
1961             case '\n': rstr += "\\n"
1962             default: rstr += string(ch)
1963         }
1964     }
1965     return rstr
1966 }
1967 func unescapeWhiteSP(str string)(string){ // strip original escapes
1968     rstr := ""
1969     for i := 0; i < len(str); i++ {
1970         ch := str[i]
1971         if ch == '\\' {
1972             if i+1 < len(str) {
1973                 switch str[i+1] {
1974                     case 'z':
1975                         continue;
1976                 }
1977             }
1978         }
1979         rstr += string(ch)
1980     }
1981     return rstr
1982 }
1983 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
1984     ustrv := []string{}
1985     for _,v := range strv {
1986         ustrv = append(ustrv,unescapeWhiteSP(v))
1987     }
1988     return ustrv
1989 }
1990
1991 // <a name="comexpansion">str-expansion</a>
1992 // - this should be a macro processor
1993 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
1994     rbuff := []byte{}
1995     if false {
1996         //@@U Unicode should be cared as a character
1997         return str
1998     }
1999     //rstr := ""
2000     inEsc := 0 // escape characer mode
2001     for i := 0; i < len(str); i++ {
2002         //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
2003         ch := str[i]
2004         if inEsc == 0 {
2005             if ch == '\\' {
2006                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
2007                 leng,rs := substHistory(gshCtx,str,i,"")
2008                 if 0 < leng {
2009                     //_,rs := substHistory(gshCtx,str,i,"")
2010                     rbuff = append(rbuff,[]byte(rs)...)
2011                     i += leng
2012                     //rstr = xrstr
2013                     continue
2014                 }
2015             }
2016             switch ch {
2017                 case '\\': inEsc = '\\'; continue
2018                 //case '$': inEsc = '$'; continue
2019                 case '$':
2020             }
2021         }
2022         switch inEsc {
2023             case '\\':
2024                 switch ch {
2025                     case '\\': ch = '\\'
2026                     case 's': ch = ' '
2027                     case 't': ch = '\t'
2028                     case 'r': ch = '\r'
2029                     case 'n': ch = '\n'
2030                     case 'z': inEsc = 0; continue // empty, to be ignored
2031                 }
2032             case '$':
2033                 switch {
2034                     case ch == '$': ch = '$'
2035                     case ch == 'T':
2036                         //rstr = rstr + time.Now().Format(time.Stamp)
2037                     case ch == 't':
2038                         rs := time.Now().Format(time.Stamp)
2039                         rbuff = append(rbuff,[]byte(rs)...)
2040                         inEsc = 0
2041                         continue;
2042                     default:
2043                         // postpone the interpretation
2044                         //rstr = rstr + "$" + string(ch)
2045                         rbuff = append(rbuff,ch)
2046                         inEsc = 0
2047                         continue;
2048                 }
2049             case 0
2050         }
2051         //rstr = rstr + string(ch)
2052         rbuff = append(rbuff,ch)
2053     }
2054     //fmt.Printf("--D--subst($s)\n",str,string(rbuff))
2055     return string(rbuff)
2056     //return rstr
2057 }
2058 func showFileInfo(path string, opts []string) {
2059     if isin("-l",opts) || isin("-ls",opts) {
2060         fi, err := os.Stat(path)
2061         if err != nil {
2062             fmt.Printf("----- ((%v))",err)
2063         }else{
2064             mod := fi.ModTime()
2065             date := mod.Format(time.Stamp)
2066             fmt.Printf("%v %8v %s ",fi.Mode(),fi.Size(),date)
2067         }
2068     }
2069     fmt.Printf("%s",path)
2070     if isin("-sp",opts) {
2071         fmt.Printf(" ")
2072     }else
2073     if ! isin("-n",opts) {
2074         fmt.Printf("\n")
2075     }
2076 }
2077 func userHomeDir()(string,bool){
2078     /*
2079     homedir, _ = os.UserHomeDir() // not implemented in older Golang
2080     */
2081     homedir,found := os.LookupEnv("HOME")
2082     //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
2083     if !found {
2084         return "/tmp",found
2085     }
2086     return homedir,found
2087 }
2088
2089 func toFullpath(path string) (fullpath string) {
2090     if path[0] == '/' {
2091         return path
2092     }
2093     pathv := strings.Split(path,DIRSEP)
2094     switch {
2095     case pathv[0] == ".":
2096     case pathv[0],_ = os.Getwd()
2097     case pathv[0] == "..": // all ones should be interpreted
2098         cwd,_ := os.Getwd()
2099         ppathv := strings.Split(cwd,DIRSEP)
2100         pathv[0] = strings.Join(ppathv,DIRSEP)
2101     case pathv[0] == "~":
2102         pathv[0],_ = userHomeDir()
2103     default:
2104         cwd,_ := os.Getwd()
2105         pathv[0] = cwd + DIRSEP + pathv[0]
2106     }
2107     return strings.Join(pathv,DIRSEP)
2108 }
2109
2110 func IsRegFile(path string)(bool){
2111     fi, err := os.Stat(path)
2112     if err == nil {
2113         fm := fi.Mode()
2114         return fm.IsRegular();
2115     }
2116     return false
2117 }
2118
2119 // <a name="encode">Encode / Decode</a>
2120 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
2121 func (gshCtx *GshContext)Enc(argv[]string){
2122     file := os.Stdin
2123     buff := make([]byte,LINESIZE)

```

```

2124     li := 0
2125     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
2126     for li = 0; li++ {
2127         count, err := file.Read(buff)
2128         if count <= 0 {
2129             break
2130         }
2131         if err != nil {
2132             break
2133         }
2134         encoder.Write(buff[0:count])
2135     }
2136     encoder.Close()
2137 }
2138 func (gshCtx *GshContext)Dec(argv[]string){
2139     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
2140     li := 0
2141     buff := make([]byte,LINESIZE)
2142     for li = 0; li++ {
2143         count, err := decoder.Read(buff)
2144         if count <= 0 {
2145             break
2146         }
2147         if err != nil {
2148             break
2149         }
2150         os.Stdout.Write(buff[0:count])
2151     }
2152 }
2153 // lnsip [N] [-crlf][-C \\\]
2154 func (gshCtx *GshContext)SplitLine(argv[]string){
2155     strRep := isin("-str",argv) // " "
2156     reader := bufio.NewReaderSize(os.Stdin,64*1024)
2157     ni := 0
2158     toi := 0
2159     for ni = 0; ni++ {
2160         line, err := reader.ReadString('\n')
2161         if len(line) <= 0 {
2162             if err != nil {
2163                 fmt.Fprintf(os.Stderr,"--I-- lnsip %d to %d (%v)\n",ni,toi,err)
2164                 break
2165             }
2166         }
2167         off := 0
2168         ilen := len(line)
2169         remlen := len(line)
2170         if strRep { os.Stdout.Write([]byte("\n")) }
2171         for oi = 0; 0 < remlen; oi++ {
2172             olen := remlen
2173             addnl := false
2174             if 72 < olen {
2175                 olen = 72
2176                 addnl = true
2177             }
2178             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d/%d\n",
2179                 toi,ni,oi,off,olen,remlen,ilen)
2180             toi += 1
2181             os.Stdout.Write([]byte(line[0:olen]))
2182             if addnl {
2183                 if strRep {
2184                     os.Stdout.Write([]byte("\n\n"))
2185                 }else{
2186                     //os.Stdout.Write([]byte("\r\n"))
2187                     os.Stdout.Write([]byte("\n"))
2188                     os.Stdout.Write([]byte("\n"))
2189                 }
2190             }
2191             line = line[olen:]
2192             off += olen
2193             remlen -= olen
2194         }
2195         if strRep { os.Stdout.Write([]byte("\n")) }
2196     }
2197     fmt.Fprintf(os.Stderr,"--I-- lnsip %d to %d\n",ni,toi)
2198 }
2199
2200 // CRC32 <a href="http://gojlang.jp/pkg/hash-crc32">crc32</a>
2201 // 1 000 0100 1100 0001 1101 1011 0111
2202 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
2203 var CRC32IEEE uint32 = uint32(0xEDB88320)
2204 func byteCRC32add(crc uint32,str[]byte,len uint64)(uint32){
2205     var oi uint64
2206     for oi = 0; oi < len; oi++ {
2207         var oct = str[oi]
2208         for bi = 0; bi < 8; bi++ {
2209             //fprintf(stderr,"--CRC32 %d %X (%d.%d)\n",crc,oct,oi,bi)
2210             ovf1 := (crc & 0x80000000) != 0
2211             ovf2 := (oct & 0x80) != 0
2212             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
2213             oct <<= 1
2214             crc <<= 1
2215             if ovf { crc ^= CRC32UNIX }
2216         }
2217     }
2218     //fprintf(stderr,"--CRC32 return %d %d\n",crc,len)
2219     return crc
2220 }
2221 func byteCRC32end(crc uint32, len uint64)(uint32){
2222     var slen = make([]byte,4)
2223     var li = 0
2224     for li = 0; li < 4; {
2225         slen[li] = byte(len)
2226     }
2227     li += 1
2228     len >= 8
2229     if (len == 0 ){
2230         break
2231     }
2232     crc = byteCRC32add(crc,slen,uint64(li))
2233     crc ^= 0xFFFFFFFF
2234     return crc
2235 }
2236 func strCRC32(str string,len uint64)(crc uint32){
2237     crc = byteCRC32add(0,[]byte(str),len)
2238     crc = byteCRC32end(crc,len)
2239     //fprintf(stderr,"--CRC32 %d %d\n",crc,len)
2240     return crc
2241 }
2242 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
2243     var slen = make([]byte,4)
2244     var li = 0
2245     for li = 0; li < 4; {
2246         slen[li] = byte(len & 0xFF)
2247     }
2248     li += 1
2249     len >= 8
2250     if (len == 0 ){
2251         break
2252     }
2253     crc = crc32.Update(crc,table,slen)
2254     crc ^= 0xFFFFFFFF
2255     return crc
2256 }
2257
2258 func (gsh*GshContext)xcksum(path string,argv[]string, sum*Checksum)(int64){
2259     if isin("-type/f",argv) && !IsRegFile(path){
2260         return 0
2261     }
2262     if isin("-type/d",argv) && !IsRegFile(path){
2263         return 0
2264     }
2265     file, err := os.OpenFile(path,os.O_RDONLY,0)
2266     if err != nil {
2267         fmt.Printf("--E-- cksum %v (%v)\n",path,err)
2268         return -1
2269     }
2270     defer file.Close()
2271     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n",path,argv) }
2272
2273     bi := 0
2274     var buff = make([]byte,32*1024)
2275     var total int64 = 0
2276     var initTime = time.Time()
2277     if sum.Start == initTime {
2278         sum.Start = time.Now()
2279     }
2280     for bi = 0; bi++ {
2281         count,err := file.Read(buff)
2282         if count <= 0 || err != nil {
2283             break
2284         }
2285         if (sum.SumType & SUM_SUM64) != 0 {
2286             s := sum.Sum64
2287             for _c := range buff[0:count] {
2288                 s += uint64(c)
2289             }
2290             sum.Sum64 = s
2291         }
2292         if (sum.SumType & SUM_UNIXFILE) != 0 {
2293             sum.Crc32Val = byteCRC32add(sum.Crc32Val,buff,uint64(count))
2294         }
2295         if (sum.SumType & SUM_CRCIEEE) != 0 {
2296             sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
2297         }
2298         // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
2299         if (sum.SumType & SUM_SUM16_BSD) != 0 {
2300             s := sum.Sum16

```

```

2301     for _,c := range buff[0:count] {
2302         s = (s >> 1) + ((s & 1) << 15)
2303         s += int(c)
2304         s &= 0xFFFF
2305         //fmt.Printf("BSData: %d[%d] %d\n",sum.Size+int64(i),i,s)
2306     }
2307     sum.Sum16 = s
2308 }
2309 if (sum.SumType & SUM_SUM16_SYSV) != 0 {
2310     for bj := 0; bj < count; bj++ {
2311         sum.Sum16 += int(buff[bj])
2312     }
2313     total += int64(count)
2314 }
2315 sum.Done = time.Now()
2316 sum.Files += 1
2317 sum.Size += total
2318 if !isin("-s",argv) {
2319     fmt.Printf("%v ",total)
2320 }
2321 return 0
2322 }
2323 }
2324
2325 // <a name="grep">grep</a>
2326 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
2327 // a*,lab,c, ... sequential combination of patterns
2328 // what "LINE" is should be definable
2329 // generic line-by-line processing
2330 // grep [-v]
2331 // cat -n -v
2332 // uniq [-c]
2333 // tail -f
2334 // sed s/x/y/ or awk
2335 // grep with line count like wc
2336 // rewrite contents if specified
2337 func (gsh*GshContext)XGrep(path string, rexp[]string)(int){
2338     file, err := os.OpenFile(path,os.O_RDONLY,0)
2339     if err != nil {
2340         fmt.Printf("--E-- grep %v (%v)\n",path,err)
2341         return -1
2342     }
2343     defer file.Close()
2344     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path, rexp) }
2345     //reader := bufio.NewReaderSize(file,LINESIZE)
2346     reader := bufio.NewReaderSize(file,80)
2347     li := 0
2348     found = 0
2349     for li = 0; li++ {
2350         line, err := reader.ReadString('\n')
2351         if len(line) <= 0 {
2352             break
2353         }
2354         if 150 < len(line) {
2355             // maybe binary
2356             break;
2357         }
2358         if err != nil {
2359             break
2360         }
2361         if 0 <= strings.Index(string(line),rexp[0]) {
2362             found += 1
2363             fmt.Printf("%s:%d: %s",path,li,line)
2364         }
2365     }
2366     //fmt.Printf("total %d lines %s\n",li,path)
2367     //if( 0 <= found ){ fmt.Printf("(found %d lines %s)\n",found,path); }
2368     return found
2369 }
2370
2371 // <a name="finder">Pinder</a>
2372 // finding files with it name and contents
2373 // file names are OReD
2374 // show the content with %x fmt list
2375 // ls -R
2376 // tar command by adding output
2377 type fileSum struct {
2378     Err int64 // access error or so
2379     Size int64 // content size
2380     DupSize int64 // content size from hard links
2381     Blocks int64 // number of blocks (of 512 bytes)
2382     DupBlocks int64 // Blocks pointed from hard links
2383     HLinks int64 // hard links
2384     Words int64
2385     Lines int64
2386     Files int64
2387     Dirs int64 // the num. of directories
2388     Symlink int64
2389     Flats int64 // the num. of flat files
2390     MaxDepth int64
2391     MaxNamlen int64 // max. name length
2392     nextRepo time.Time
2393 }
2394 func showFusage(dir string,fusage *fileSum){
2395     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
2396     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
2397
2398     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
2399         dir,
2400         fusage.Files,
2401         fusage.Dirs,
2402         fusage.Symlink,
2403         fusage.HLinks,
2404         float64(fusage.Size)/1000000.0,bsume);
2405 }
2406 const {
2407     S_IFMT = 0170000
2408     S_IFCHR = 0020000
2409     S_IFDIR = 0040000
2410     S_IFREG = 0100000
2411     S_IFLNK = 0120000
2412     S_IFSOCK = 0140000
2413 }
2414 func cumPinfo(fsum *fileSum, path string, stater error, fstat aStat_t, argv[]string,verb bool)(*fileSum){
2415     now := time.Now()
2416     if time.Second <= now.Sub(fsum.nextRepo) {
2417         if !fsum.nextRepo.IsZero(){
2418             tstamp := now.Format(time.Stamp)
2419             showFusage(tstamp, fsum)
2420         }
2421         fsum.nextRepo = now.Add(time.Second)
2422     }
2423     if stater != nil {
2424         fsum.Err += 1
2425         return fsum
2426     }
2427     fsum.Files += 1
2428     if 1 < fstat.Nlink {
2429         // must count only once...
2430         // at least ignore ones in the same directory
2431         //if !finfo.Mode().IsRegular() {
2432             if (fstat.Mode & S_IFMT) == S_IFREG {
2433                 fsum.HLinks += 1
2434                 fsum.DupBlocks += int64(fstat.Blocks)
2435                 //fmt.Printf("---Dup HardLink %v %s\n",fstat.Nlink,path)
2436             }
2437         }
2438         //fsum.Size += finfo.Size()
2439         fsum.Size += fstat.Size
2440         fsum.Blocks += int64(fstat.Blocks)
2441         //if verb { fmt.Printf("%8dBk %s",fstat.Blocks/2,path) }
2442         //if !is("-ls",argv){
2443             //if verb { fmt.Printf("%4d %8d ",fstat.Blksize,fstat.Blocks) }
2444             // fmt.Printf("%d\t",fstat.Blocks/2)
2445         }
2446         //if finfo.IsDir()
2447         if (fstat.Mode & S_IFMT) == S_IFDIR {
2448             fsum.Dirs += 1
2449         }
2450         //if (finfo.Mode() & os.ModeSymlink) != 0
2451         if (fstat.Mode & S_IFMT) == S_IFLNK {
2452             //if verb { fmt.Printf("symlink(%v,%s)\n",fstat.Mode,finfo.Name()) }
2453             //if !fsum.Symlink.IsZero(){
2454                 //fmt.Printf("symlink(%s,%s)\n",fstat.Mode,finfo.Name())
2455             }
2456             fsum.Symlink += 1
2457         }
2458         return fsum
2459     }
2460 }
2461 func (gsh*GshContext)xxFindEntv(depth int,total *fileSum,dir string, dstat aStat_t, ei int, entv []string,npatv[]string,argv[]string)(*fileSum){
2462     nois := isin("-grep",argv)
2463     // sort entv
2464     /*
2465     if isin("-t",argv){
2466         sort.Slice(filev, func(i,j int) bool {
2467             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
2468         })
2469     }
2470     */
2471     if isin("-u",argv){
2472         sort.Slice(filev, func(i,j int) bool {
2473             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
2474         })
2475     }
2476     if isin("-U",argv){
2477         sort.Slice(filev, func(i,j int) bool {
2478             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
2479         })
2480     }

```

```

2478     }
2479     /*
2480     if isin("-s",argv){
2481         sort.Slice(filev, func(i,j int) bool {
2482             return filev[j].Size() < filev[i].Size()
2483         })
2484     }
2485     */
2486     for _,filename := range entv {
2487         for _,npat := range npatv {
2488             match := true
2489             if npat == "*" {
2490                 match = true
2491             } else {
2492                 match, _ = filepath.Match(npat,filename)
2493             }
2494             path := dir + DIRSEP + filename
2495             if !match {
2496                 continue
2497             }
2498             var fstat aStat_t
2499             staterr := aLstat(path,&fstat)
2500             if staterr != nil {
2501                 if !isin("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
2502                 continue;
2503             }
2504             if isin("-du",argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
2505                 // should not show size of directory in "-du" mode ...
2506             } else {
2507                 if !inols && !isin("-s",argv) && (!isin("-du",argv) || !isin("-a",argv)) {
2508                     if !isin("-du",argv) {
2509                         fmt.Printf("%d\t",fstat.Blocks/2)
2510                     }
2511                     showFileInfo(path,argv)
2512                 }
2513             }
2514             if true { // && isin("-du",argv)
2515                 total = cumFinfo(total,path,staterr,fstat,argv,false)
2516             }
2517             /*
2518             if isin("-wc",argv) {
2519                 */
2520             if gsh.lastCheckSum.SumType != 0 {
2521                 gsh.xCksum(path,argv,&gsh.lastCheckSum);
2522             }
2523             x := isinX("-grep",argv); // -grep will be convenient like -ls
2524             if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
2525                 if !isRegFile(path) {
2526                     found := gsh.xGrep(path,argv[x+1:])
2527                     if 0 < found {
2528                         foundv := gsh.CmdCurrent.FoundFile
2529                         if len(foundv) < 10 {
2530                             gsh.CmdCurrent.FoundFile =
2531                                 append(gsh.CmdCurrent.FoundFile,path)
2532                         }
2533                     }
2534                 }
2535             }
2536             if !isin("-r0",argv) { // -d 0 in du, -depth n in find
2537                 //total.Depth += 1
2538                 if (fstat.Mode & S_IFMT) == S_IFLNK {
2539                     continue
2540                 }
2541                 if dstat.Rdev != fstat.Rdev {
2542                     fmt.Printf("--I-- don't follow differnet device %v(%v) %v(%v)\n",
2543                         dir,dstat.Rdev,path,fstat.Rdev)
2544                 }
2545                 if (fstat.Mode & S_IFMT) == S_IFDIR {
2546                     total = gsh.xxFind(depth+1,total,path,npatv,argv)
2547                 }
2548             }
2549         }
2550     }
2551     }
2552     return total
2553 }
2554 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum){
2555     nols := isin("-grep",argv)
2556     dirfile,oerr := os.OpenFile(dir,os.O_RDONLY,0)
2557     if oerr == nil {
2558         //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
2559         defer dirfile.Close()
2560     } else {
2561     }
2562     prev := *total
2563     var dstat aStat_t
2564     staterr := aLstat(dir,&dstat) // should be flstat
2565     if staterr != nil {
2566         if !isin("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
2567         return total
2568     }
2569     //filev,err := ioutil.ReadDir(dir)
2570     //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
2571     /*
2572     if err != nil {
2573         if !isin("-w",argv){fmt.Printf("ufind: %v\n",err) }
2574         return total
2575     }
2576     */
2577     if depth == 0 {
2578         total = cumFinfo(total,dir,staterr,dstat,argv,true)
2579         if !inols && !isin("-s",argv) && (!isin("-du",argv) || !isin("-a",argv)) {
2580             showFileInfo(dir,argv)
2581         }
2582     }
2583     // it is not a directory, just scan it and finish
2584     for ei := 0; ; ei++ {
2585         entv,rdrerr := dirfile.Readdirnames(8*1024)
2586         if len(entv) == 0 || rdrerr != nil {
2587             //if rdrerr != nil {fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rdrerr) }
2588             break
2589         }
2590         if 0 < ei {
2591             fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
2592         }
2593         total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
2594     }
2595     if isin("-du",argv) {
2596         // if in "du" mode
2597         fmt.Printf("%d\t%s\n",total.Blocks-prev.Blocks)/2,dir)
2598     }
2599     return total
2600 }
2601 //
2602 // (ufind[ful]e) [Files] [// Names] [-- Expressions]
2603 // Files is "." by default
2604 // Names is "*" by default
2605 // Expressions is "-print" by default for "ufind", or -du for "fu" command
2606 func (gsh*GshContext)xxFind(argv[]string){
2607     if 0 < len(argv) && strBegins(argv[0],"?"){
2608         showFound(gsh,argv)
2609         return
2610     }
2611     if isin("-cksum",argv) || isin("-sum",argv) {
2612         gsh.lastCheckSum = CheckSum{}
2613         if isin("-sum",argv) && isin("-add",argv) {
2614             gsh.lastCheckSum.SumType |= SUM_SUM64
2615         } else {
2616             if isin("-sum",argv) && isin("-size",argv) {
2617                 gsh.lastCheckSum.SumType |= SUM_SIZE
2618             } else {
2619                 if isin("-sum",argv) && isin("-bsd",argv) {
2620                     gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
2621                 } else {
2622                     if isin("-sum",argv) && isin("-sysv",argv) {
2623                         gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
2624                     } else {
2625                         if isin("-sum",argv) {
2626                             gsh.lastCheckSum.SumType |= SUM_SUM64
2627                         }
2628                     }
2629                 }
2630             }
2631             if isin("-unix",argv) {
2632                 gsh.lastCheckSum.SumType |= SUM_UNIXFILE
2633                 gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32UNIX)
2634             }
2635             if isin("-ieee",argv){
2636                 gsh.lastCheckSum.SumType |= SUM_CRCIEEE
2637                 gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32IEEE)
2638             }
2639             gsh.lastCheckSum.RusgAtStart = Getrusagev()
2640         }
2641     }
2642     var total = fileSum{}
2643     npats := []string{}
2644     for _,v := range argv {
2645         if 0 < len(v) && v[0] != '-' {
2646             npats = append(npats,v)
2647         }
2648         if v == "/" { break }
2649         if v == "--" { break }
2650         if v == "-grep" { break }
2651         if v == "-ls" { break }
2652     }
2653     if len(npats) == 0 {
2654         npats = []string{"*"}
2655     }
2656 }

```

```

2655 cwd := "."
2656 // if to be fullpath :: cwd, := os.Getwd()
2657 if len(npats) == 0 { npats = []string{"*"} }
2658 fusage := gsh.xxFind(0,&total,cwd,npats,argv)
2659 if gsh.lastCheckSum.SumType != 0 {
2660     var sumi uint64 = 0
2661     sum := gsh.lastCheckSum
2662     if (sum.SumType & SUM_SIZE) != 0 {
2663         sumi = uint64(sum.Size)
2664     }
2665     if (sum.SumType & SUM_SUM64) != 0 {
2666         sumi = sum.Sum64
2667     }
2668     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
2669         s := uint32(sum.Sum16)
2670         r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
2671         sum = (r & 0xFFFF) + (r >> 16)
2672         sum.Crc32Val = uint32(s)
2673         sumi = uint64(s)
2674     }
2675     if (sum.SumType & SUM_SUM16_BSD) != 0 {
2676         sum.Crc32Val = uint32(sum.Sum16)
2677         sumi = uint64(sum.Sum16)
2678     }
2679     if (sum.SumType & SUM_UNIXFILE) != 0 {
2680         sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
2681         sumi = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
2682     }
2683     if 1 < sum.Files {
2684         fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
2685             sumi,sum.Size,
2686             absSize(sum.Size),sum.Files,
2687             absSize(sum.Size/sum.Files))
2688     }else{
2689         fmt.Printf("%v %v\n",
2690             sumi,sum.Size,npats[0])
2691     }
2692 }
2693 if !isin("-grep",argv) {
2694     showFusage("total",fusage)
2695 }
2696 if !isin("-s",argv){
2697     hits := len(gsh.CmdCurrent.FoundFile)
2698     if 0 < hits {
2699         fmt.Printf("--I-- %d files hits // can be refered with !&df\n",
2700             hits,len(gsh.CommandHistory))
2701     }
2702 }
2703 if gsh.lastCheckSum.SumType != 0 {
2704     if !isin("-ru",argv) {
2705         sum := gsh.lastCheckSum
2706         sum.Done = time.Now()
2707         gsh.lastCheckSum.RusqAtEnd = Cetrusagev()
2708         elps := sum.Done.Sub(sum.Start)
2709         fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
2710             sum.Size,absSize(sum.Size),sum.Files,absSize(sum.Size/sum.Files))
2711         nanos := int64(elps)
2712         dnanos := time.Duration(nanos)
2713         fmt.Printf("--cksum-time: %v/total, %v/file, %i files/s, %v/r\n",
2714             abftime(dnanos),
2715             abftime(time.Duration(nanos/sum.Files)),
2716             (float64(sum.Files)*100000000.0)/float64(nanos),
2717             abbspeed(sum.Size,nanos))
2718         diff := RusageSub(sum.RusqAtEnd,sum.RusqAtStart)
2719         fmt.Printf("--cksum-rusg: %v\n",sRusagef("",argv,diff))
2720     }
2721 }
2722 return
2723 }
2724
2725 func showFiles(files[]string){
2726     sp := ""
2727     for i,file := range files {
2728         if 0 < i { sp = " " } else { sp = "" }
2729         fmt.Printf(sp"%s",escapeWhiteSP(file))
2730     }
2731 }
2732 func showFound(gshCtx *GshContext, argv[]string){
2733     for i,v := range gshCtx.CommandHistory {
2734         if 0 < len(v.FoundFile) {
2735             fmt.Printf("%d (%d) %i,%i\n",i,len(v.FoundFile))
2736             if !isin("-s",argv){
2737                 fmt.Printf("\n")
2738                 for _,file := range v.FoundFile {
2739                     fmt.Printf("%i //sub number?
2740                         showFileInfo(file,argv)
2741                 }
2742             }else{
2743                 showFiles(v.FoundFile)
2744                 fmt.Printf("\n")
2745             }
2746         }
2747     }
2748 }
2749
2750 func showMatchFile(filev []os.FileInfo, npat,dir string, argv[]string)(string,bool){
2751     fname := ""
2752     found := false
2753     for _,v := range filev {
2754         match, := filepath.Match(npat,(v.Name()))
2755         if match {
2756             fname = v.Name()
2757             found = true
2758             //fmt.Printf("%d (%d) %i,%i\n",i,len(v.FoundFile))
2759             showIfExecutable(fname,dir,argv)
2760         }
2761     }
2762     return fname,found
2763 }
2764 func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
2765     var fullpath string
2766     if strBegins(name,DIRSEP){
2767         fullpath = name
2768     }else{
2769         if( len(dir) == 0 ){
2770             fullpath = name;
2771         }else{
2772             fullpath = dir + DIRSEP + name
2773         }
2774     }
2775     fi, err := os.Stat(fullpath)
2776     //fmt.Printf("--Dp-- %v\n",fullpath,err);
2777     if err != nil {
2778         fullpath += ".exe";
2779         fi, err = os.Stat(fullpath)
2780     }
2781     if err != nil {
2782         fullpath = dir + DIRSEP + name + ".go"
2783         fi, err = os.Stat(fullpath)
2784     }
2785     if err == nil {
2786         fm := fi.Mode()
2787         if fm.IsRegular() {
2788             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
2789             if aAccess(fullpath,5) == nil {
2790                 ffullpath = fullpath
2791                 ffound = true
2792                 if ! isin("-s", argv) {
2793                     showFileInfo(fullpath,argv)
2794                 }
2795             }
2796         }
2797     }
2798     return ffullpath, ffound
2799 }
2800 func which(list string, argv []string) (fullpathv []string, itis bool){
2801     if len(argv) == 1 {
2802         fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
2803         return []string{"", false}
2804     }
2805     path := argv[1]
2806     if strBegins(path,"/") {
2807         // should check if executable?
2808         ,exOK := showIfExecutable(path,"",argv)
2809         fmt.Printf("--D- %v exOK=%v\n",path,exOK)
2810         return []string{path},exOK
2811     }
2812     pathenv, efound := os.LookupEnv(list)
2813     if ! efound {
2814         fmt.Printf("--E- which: no \"%s\" environment\n",list)
2815         return []string{"", false}
2816     }
2817     //fmt.Printf("PATH=%v\n",pathenv);
2818     showall := isin("-a",argv) || 0 < strings.Index(path,"*")
2819     dirv := strings.Split(pathenv,PATHSEP)
2820     ffound := false
2821     ffullpath := path
2822     for dir := range dirv {
2823         if 0 < strings.Index(path,"*") { // by wild-card
2824             list, := ioutil.ReadDir(dir)
2825             ffullpath, ffound = showMatchFile(list,path,dir,argv)
2826         }else{
2827             ffullpath, ffound = showIfExecutable(path,dir,argv)
2828         }
2829         //if ffound && !isin("-a", argv) {
2830             break;
2831         }
2832     }
2833 }

```



```

2832 }
2833 return []string{ffullpath}, ffound
2834 }
2835
2836 func stripLeadingWSParg(argv []string) ([]string){
2837     for i, 0 < len(argv); i++ {
2838         if len(argv[i]) == 0 {
2839             argv = argv[1:]
2840         }else{
2841             break
2842         }
2843     }
2844     return argv
2845 }
2846 func xEval(argv []string, nleend bool){
2847     argv = stripLeadingWSParg(argv)
2848     if len(argv) == 0 {
2849         fmt.Printf("eval [%sformat] [Go-expression]\n")
2850         return
2851     }
2852     pfmt := "%v"
2853     if argv[0][0] == '%' {
2854         pfmt = argv[0]
2855         argv = argv[1:]
2856     }
2857     if len(argv) == 0 {
2858         return
2859     }
2860     gocode := strings.Join(argv, " ");
2861     //fmt.Printf("eval [%v] [%v]\n", pfmt, gocode)
2862     fset := token.NewFileSet()
2863     rval, _ := types.Eval(fset, nil, token.NoPos, gocode)
2864     fmt.Printf(pfmt, rval.Value)
2865     if nleend { fmt.Println() }
2866 }
2867
2868 func getval(name string) (found bool, val int) {
2869     /* should expand the name here */
2870     if name == "gsh.pid" {
2871         return true, os.Getpid()
2872     }else{
2873         if name == "gsh.ppid" {
2874             return true, os.Getppid()
2875         }
2876         return false, 0
2877     }
2878 }
2879 func echo(argv []string, nleend bool){
2880     for ai := 1; ai < len(argv); ai++ {
2881         if ai < ai {
2882             fmt.Printf(" ");
2883         }
2884         arg := argv[ai]
2885         found, val := getval(arg)
2886         if found {
2887             fmt.Printf("%d", val)
2888         }else{
2889             fmt.Printf("%s", arg)
2890         }
2891     }
2892     if nleend {
2893         fmt.Println();
2894     }
2895 }
2896
2897 func resfile() string {
2898     return "gsh.tmp"
2899 }
2900 //var resF *File
2901 func resmap() {
2902     //_, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
2903     //https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
2904     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
2905     if err != nil {
2906         fmt.Printf("refF could not open: %s\n", err)
2907     }else{
2908         fmt.Printf("refF opened\n")
2909     }
2910 }
2911
2912 // ##2020-0821
2913 func gshScanArg(str string, strip int)(argv []string){
2914     var si = 0
2915     var sb = 0
2916     var inBracket = 0
2917     var argl = make([]byte, LINESIZE)
2918     var ax = 0
2919     debug := false
2920
2921     for ; si < len(str); si++ {
2922         if str[si] != ' ' {
2923             break
2924         }
2925     }
2926     sb = si
2927     for ; si < len(str); si++ {
2928         if sb <= si {
2929             if debug {
2930                 fmt.Printf("--Da- +%d %d-%d %s ... %s\n",
2931                     inBracket, sb, si, str[0:ax], str[si:])
2932             }
2933             ch := str[si]
2934             if ch == '{' {
2935                 inBracket += 1
2936                 if 0 < strip && inBracket <= strip {
2937                     //fmt.Printf("stripLEV %d <= %d\n", inBracket, strip)
2938                     continue
2939                 }
2940             }
2941             if 0 < inBracket {
2942                 if ch == '}' {
2943                     inBracket -= 1
2944                     if 0 < strip && inBracket < strip {
2945                         //fmt.Printf("stripLEV %d < %d?\n", inBracket, strip)
2946                         continue
2947                     }
2948                 }
2949             }
2950             argl[ax] = ch
2951             ax += 1
2952             continue
2953         }
2954         if str[si] == ' ' {
2955             argv = append(argv, string(argl[0:ax]))
2956             if debug {
2957                 fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2958                     -1+len(argv), sb, si, string(argl[0:ax]), string(str[si:]))
2959             }
2960             sb = si+1
2961             ax = 0
2962             continue
2963         }
2964         argl[ax] = ch
2965         ax += 1
2966     }
2967     if sb < si {
2968         argv = append(argv, string(argl[0:ax]))
2969         if debug {
2970             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2971                 -1+len(argv), sb, si, string(argl[0:ax]), string(str[si:]))
2972         }
2973     }
2974     if debug {
2975         fmt.Printf("--Da- %d [%s] => [%d]\v\n", strip, str, len(argv), argv)
2976     }
2977     return argv
2978 }
2979
2980 // should get stderr (into tmpfile ?) and return
2981 func (gsh*GshContext)Dopen(name, mode string)(pin*os.File, pout*os.File, err bool){
2982     //var pv = []int{-1, -1}
2983     //syscall.Pipe(pv)
2984
2985     xarg := gshScanArg(name, 1)
2986     name = strings.Join(xarg, " ")
2987
2988     //pin = os.NewFile(uintptr(pv[0]), "StdoutOf-"+name+"")
2989     //pout = os.NewFile(uintptr(pv[1]), "StdinOf-"+name+"")
2990     pin, pout, _ = os.Pipe();
2991
2992     fdix := 0
2993     dir := "r"
2994     if mode == "r" {
2995         dir = "r<"
2996     }else{
2997         fdix = 1 // read from the stdout of the process
2998     }
2999     dir = ">"
3000     fdix = 0 // write to the stdin of the process
3001
3002     gshPA := gsh.gshPA
3003     savfd := gshPA.Files[fdix]
3004
3005     var fd uintptr = 0
3006     if mode == "r" {
3007         //fd = pout.Fd()
3008         //gshPA.Files[fdix] = pout.Fd()
3009         gshPA.Files[fdix] = pout;

```

```

3009 }else{
3010 //fd = pin.Fd()
3011 //gshPA.Files[fdix] = pin.Fd()
3012 gshPA.Files[fdix] = pin;
3013 }
3014 // should do this by Goroutine?
3015 if false {
3016     fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
3017     fmt.Printf("--RED1 [%d,%d,%d]->[%d,%d,%d]\n",
3018         os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
3019         pin.Fd(),pout.Fd(),pout.Fd())
3020 }
3021 savi := os.Stdin
3022 sawo := os.Stdout
3023 save := os.Stderr
3024 os.Stdin = pin
3025 os.Stdout = pout
3026 os.Stderr = pout
3027 gsh.BackGround = true
3028 gsh.gshellh(name)
3029 gsh.BackGround = false
3030 os.Stdin = savi
3031 os.Stdout = sawo
3032 os.Stderr = save
3033
3034 gshPA.Files[fdix] = savfd
3035 return pin,pout,false
3036 }
3037
3038 // <a name="ex-commands">External commands</a>
3039 func (gsh*GshContext)execcommand(exec bool, argv []string) (notf bool,exit bool) {
3040     if gsh.CmdTrace { fmt.Printf("--I-- execcommand[%v](%v)\n",exec,argv) }
3041
3042     gshPA := gsh.gshPA
3043     fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
3044     if itis == false {
3045         return true,false
3046     }
3047     fullpath := fullpathv[0]
3048     argv = unescapeWhiteSP(argv)
3049     if 0 < strings.Index(fullpath,".go") {
3050         nargv := argv // []string{}
3051         gofullpath, itis := which("PATH",[]string{"which","go","-s"})
3052         if itis == false {
3053             fmt.Printf("--P-- Go not found\n")
3054             return false,true
3055         }
3056         gofullpath := gofullpathv[0]
3057         nargv = []string{gofullpath,"run",fullpath}
3058         fmt.Printf("--I-- %s (%s %s)\n",gofullpath,
3059             nargv[0],nargv[1],nargv[2])
3060         if exec {
3061             syscall.Exec(gofullpath,nargv,os.Environ())
3062         }else{
3063             //pid, _ := syscall.ForkExec(gofullpath,nargv,gshPA)
3064             proc, _ := os.StartProcess(gofullpath,nargv,gshPA);
3065             pstat, _ := proc.Wait();
3066             pid := pstat.Pid();
3067             if gsh.BackGround {
3068                 fmt.Printf(stderr,"--Ip- in Background pid[%d]%(%v)\n",pid,len(argv),argv)
3069                 //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
3070                 gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
3071             }else{
3072                 /*
3073                 rusage := aRusage {}
3074                 syscall.Wait4(pid,nil,0,*rusage)
3075                 gsh.LastRusage = rusage
3076                 gsh.CmdCurrent.Rusagev[1] = rusage
3077                 */
3078             /*
3079             gsh.LastRusage = *pstat.SysUsage().(*aRusage);
3080             gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
3081             */
3082             aSetrusage(&gsh.LastRusage,pstat);
3083             gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
3084             }
3085         }
3086     }else{
3087         if exec {
3088             syscall.Exec(fullpath,argv,os.Environ())
3089         }else{
3090             //pid, _ := syscall.ForkExec(fullpath,argv,gshPA)
3091             proc, _ := os.StartProcess(fullpath,argv,gshPA);
3092             pstat, _ := proc.Wait();
3093             pid := pstat.Pid();
3094             //fmt.Printf("%d\n",pid); // '%d' to be background
3095         }if false {
3096             fmt.Printf("Sys=%v\n",gshPA.Sys);
3097         }if (gshPA.Sys != nil) {
3098             //fmt.Printf("inFG=%v\n",gshPA.Sys.Foreground);
3099         }
3100     }
3101     if gsh.BackGround {
3102         fmt.Printf(stderr,"--Ip- in Background pid[%d]%(%v)\n",pid,len(argv),argv)
3103         //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
3104         gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
3105     }else{
3106         /*
3107         rusage := aRusage {}
3108         syscall.Wait4(pid,nil,0,*rusage);
3109         gsh.LastRusage = rusage
3110         gsh.CmdCurrent.Rusagev[1] = rusage
3111         */
3112     /*
3113     gsh.LastRusage = *pstat.SysUsage().(*aRusage);
3114     gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
3115     */
3116     aSetrusage(&gsh.LastRusage,pstat);
3117     gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
3118     }
3119     }
3120     }
3121     return false,false
3122 }
3123
3124 // <a name="builtin">Builtin Commands</a>
3125 func (gshCtx *GshContext) sleep(argv []string) {
3126     if len(argv) < 2 {
3127         fmt.Printf("Sleep 100ms, 100us, 100ns, ...n")
3128         return
3129     }
3130     duration := argv[1];
3131     d, err := time.ParseDuration(duration)
3132     if err != nil {
3133         d, err = time.ParseDuration(duration+"s")
3134         if err != nil {
3135             fmt.Printf("duration ? %s (%s)\n",duration,err)
3136         }
3137     }
3138     time.Sleep(d)
3139     //fmt.Printf("Sleep %v\n",duration)
3140     if 0 < len(argv[2:]) {
3141         gshCtx.gshellv(argv[2:])
3142     }
3143 }
3144
3145 func (gshCtx *GshContext)repeat(argv []string) {
3146     if len(argv) < 2 {
3147         return
3148     }
3149     start0 := time.Now()
3150     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
3151         if 0 < len(argv[2:]) {
3152             //start := time.Now()
3153             gshCtx.gshellv(argv[2:])
3154             end := time.Now()
3155             elps := end.Sub(start0);
3156             if( 1000000000 < elps ){
3157                 fmt.Printf("(repeat%d %v)\n",ri,elps);
3158             }
3159         }
3160     }
3161 }
3162
3163 func (gshCtx *GshContext)gen(argv []string) {
3164     gshPA := gshCtx.gshPA
3165     if len(argv) < 2 {
3166         fmt.Printf("Usage: %s N\n",argv[0])
3167         return
3168     }
3169     // should br repeated by "repeat" command
3170     count, _ := strconv.Atoi(argv[1])
3171     //fd := gshPA.Files[1] // Stdout
3172     //file := os.NewFile(fd,"internalStdout")
3173     file := gshPA.Files[1]; // Stdout
3174     fmt.Printf("--I-- Gen. Count=%d to [%d]\n",count,file.Fd())
3175     //buf := []byte{}
3176     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
3177     for gi := 0; gi < count; gi++ {
3178         file.WriteString(outdata)
3179     }
3180     //file.WriteString("\n")
3181     //fmt.Printf("\n(%d B)\n",count*len(outdata));
3182     //file.Close()
3183 }
3184
3185

```

```

3186 // <a name="rexec">Remote Execution</a> // 2020-0820
3187 func Elapsed(from time.Time)(string){
3188     elps := time.Now().Sub(from)
3189     if 1000000000 < elps {
3190         return fmt.Sprintf("%5d.%02ds]", elps/1000000000, (elps%1000000000)/10000000)
3191     }else
3192     if 1000000 < elps {
3193         return fmt.Sprintf("[%3d.%03dms]", elps/1000000, (elps%1000000)/1000)
3194     }else{
3195         return fmt.Sprintf("[%3d.%03dus]", elps/1000, (elps%1000))
3196     }
3197 }
3198 //func abbtTime(nanos int64)(string){
3199 func abbtTime(nanos time.Duration)(string){
3200     if 1000000000 < nanos {
3201         return fmt.Sprintf("%d.%02ds", nanos/1000000000, (nanos%1000000000)/10000000)
3202     }else
3203     if 1000000 < nanos {
3204         return fmt.Sprintf("%d.%03dms", nanos/1000000, (nanos%1000000)/1000)
3205     }else{
3206         return fmt.Sprintf("%d.%03dus", nanos/1000, (nanos%1000))
3207     }
3208 }
3209 func absSize(size int64)(string){
3210     fsize := float64(size)
3211     if 1024*1024*1024 < size {
3212         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
3213     }else
3214     if 1024*1024 < size {
3215         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
3216     }else{
3217         return fmt.Sprintf("%.3fKiB", fsize/1024)
3218     }
3219 }
3220 func absSize(size int64)(string){
3221     fsize := float64(size)
3222     if 1024*1024*1024 < size {
3223         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
3224     }else
3225     if 1024*1024 < size {
3226         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
3227     }else{
3228         return fmt.Sprintf("%.3fKiB", fsize/1024)
3229     }
3230 }
3231 func absSpeed(totalB int64, ns int64)(string){
3232     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
3233     if 1000 <= MBs {
3234         return fmt.Sprintf("%6.3fGB/s", MBs/1000)
3235     }
3236     if 1 <= MBs {
3237         return fmt.Sprintf("%6.3fMB/s", MBs)
3238     }else{
3239         return fmt.Sprintf("%6.3fKB/s", MBs*1000)
3240     }
3241 }
3242 func absSpeed(totalB int64, ns time.Duration)(string){
3243     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
3244     if 1000 <= MBs {
3245         return fmt.Sprintf("%6.3fGBps", MBs/1000)
3246     }
3247     if 1 <= MBs {
3248         return fmt.Sprintf("%6.3fMBps", MBs)
3249     }else{
3250         return fmt.Sprintf("%6.3fKBps", MBs*1000)
3251     }
3252 }
3253 func fileRelay(what string, in*os.File, out*os.File, size int64, bsiz int)(wcount int64){
3254     Start := time.Now()
3255     buff := make([]byte, bsiz)
3256     var total int64 = 0
3257     var rem int64 = size
3258     nio := 0
3259     Prev := time.Now()
3260     var PrevSize int64 = 0
3261 }
3262 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
3263     what, absSize(total), size, nio)
3264 }
3265 for i:= 0; ; i++ {
3266     var len = bsiz
3267     if int(rem) < len {
3268         len = int(rem)
3269     }
3270     Now := time.Now()
3271     Elps := Now.Sub(Prev);
3272     if 1000000000 < Now.Sub(Prev) {
3273         fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
3274             what, absSize(total), size, nio,
3275             absSpeed((total-PrevSize), Elps))
3276         Prev = Now;
3277         PrevSize = total
3278     }
3279     rlen := len
3280     if in != nil {
3281         // should watch the disconnection of out
3282         rcc, err := in.Read(buff[0:rlen])
3283         if err != nil {
3284             fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<v\n",
3285                 what, rcc, err, in.Name())
3286             break
3287         }
3288         rlen = rcc
3289         if string(buff[0:10]) == "(SoftEOF " {
3290             var ecc int64 = 0
3291             fmt.Sprintf(string(buff), "(SoftEOF %v", &ecc)
3292             fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))&v\n",
3293                 what, ecc, total)
3294             if ecc == total {
3295                 break
3296             }
3297         }
3298     }
3299 }
3300 wlen := rlen
3301 if out != nil {
3302     wcc, err := out.Write(buff[0:rlen])
3303     if err != nil {
3304         fmt.Printf(Elapsed(Start)+"--En-- X: %s write(%v,%v)>v\n",
3305             what, wcc, err, out.Name())
3306         break
3307     }
3308     wlen = wcc
3309 }
3310 if wlen < rlen {
3311     fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
3312         what, wlen, rlen)
3313     break;
3314 }
3315 nio += 1
3316 total += int64(rlen)
3317 rem -= int64(rlen)
3318 if rem <= 0 {
3319     break
3320 }
3321 }
3322 Done := time.Now()
3323 Elps := float64(Done.Sub(Start))/1000000000 //Seconds
3324 TotalMB := float64(total)/1000000 //MB
3325 MBps := TotalMB / Elps
3326 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %v %.3fMB/s\n",
3327     what, total, size, nio, absSize(total), MBps)
3328 return total
3329 }
3330 func tcpPush(clnt *os.File){
3331     // shrink socket buffer and recover
3332     usleep(100);
3333 }
3334 }
3335 func (gsh*GshContext)RexecServer(argv[string]){
3336     debug := true
3337     Start0 := time.Now()
3338     Start := Start0
3339     // if local == ":" { local = "0.0.0.0:9999" }
3340     local := "0.0.0.0:9999"
3341 }
3342 if 0 < len(argv) {
3343     if argv[0] == "-s" {
3344         debug = false
3345         argv = argv[1:]
3346     }
3347 }
3348 if 0 < len(argv) {
3349     argv = argv[1:]
3350 }
3351 port, err := net.ResolveTCPAddr("tcp", local);
3352 if err != nil {
3353     fmt.Printf("--En- S: Address error: %s (%s)\n", local, err)
3354     return
3355 }
3356 fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n", local);
3357 sconn, err := net.ListenTCP("tcp", port)
3358 if err != nil {
3359     fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n", local, err)
3360     return
3361 }
3362 }

```

```

3363 reqbuf := make([]byte,LINESIZE)
3364 res := ""
3365 for {
3366     fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
3367     acconn, err := sconn.AcceptTCP()
3368     Start = time.Now()
3369     if err != nil {
3370         fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
3371         return
3372     }
3373     clnt, _ := acconn.File()
3374     fd := clnt.Fd()
3375     ar := acconn.RemoteAddr()
3376     if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
3377         local,fd,ar) }
3378     res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
3379     fmt.Printf(clnt, "%s",res)
3380     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
3381     count, err := clnt.Read(reqbuf)
3382     if err != nil {
3383         fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
3384             count,err,string(reqbuf))
3385     }
3386     req := string(reqbuf[:count])
3387     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
3388     reqv := strings.Split(string(req),"\r")
3389     cmdv := gshScanArg(reqv[0],0)
3390     //cmdv := strings.Split(reqv[0], " ")
3391     switch cmdv[0] {
3392     case "HELO":
3393         res = fmt.Sprintf("250 %v",req)
3394     case "GET":
3395         // download {remotefile|-zN} [localfile]
3396         var dsz int64 = 32*1024*1024
3397         var bsize int = 64*1024
3398         var fname string = ""
3399         var in *os.File = nil
3400         var pseudoEOF = false
3401         if 1 < len(cmdv) {
3402             fname = cmdv[1]
3403             if strBegins(fname,"-z") {
3404                 fmt.Sscanf(fname[2:], "%d",&dsz)
3405             }else{
3406                 if strBegins(fname,"(") {
3407                     xin,xout,err := gsh.Popen(fname,"r")
3408                     if err {
3409                         }else{
3410                             xout.Close()
3411                             defer xin.Close()
3412                             in = xin
3413                             dsz = MaxStreamSize
3414                             pseudoEOF = true
3415                         }
3416                     }else{
3417                         xin,err := os.Open(fname)
3418                         if err != nil {
3419                             fmt.Printf("--En- GET (%v)\n",err)
3420                         }else{
3421                             defer xin.Close()
3422                             fi, _ := xin.Stat()
3423                             dsz = fi.Size()
3424                         }
3425                     }
3426                 }
3427             }
3428             //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dsz,bsize)
3429             res = fmt.Sprintf("200 %v\r\n",dsz)
3430             fmt.Fprintf(clnt, "%v",res)
3431             tcpPush(clnt); // should be separated as line in receiver
3432             fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
3433             wcount := fileRelay("SendGET",in,clnt,dsz,bsize)
3434             if pseudoEOF {
3435                 in.Close() // pipe from the command
3436                 // show end of stream data (its size) by OOB?
3437                 SoftEOF := fmt.Sprintf("((SoftEOF %v))",wcount)
3438                 fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
3439             }
3440             tcpPush(clnt); // to let SoftEOF data appear at the top of received data
3441             fmt.Fprintf(clnt, "%v\r\n",SoftEOF)
3442             tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
3443             // with client generated random?
3444             //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
3445         }
3446         res = fmt.Sprintf("200 GET done\r\n")
3447     case "PUT":
3448         // upload {srcfile|-zN} [dstfile]
3449         var dsz int64 = 32*1024*1024
3450         var bsize int = 64*1024
3451         var fname string = ""
3452         var out *os.File = nil
3453         if 1 < len(cmdv) { // localfile
3454             fmt.Sscanf(cmdv[1], "%d",&dsz)
3455         }
3456         if 2 < len(cmdv) {
3457             fname = cmdv[2]
3458             if fname == "-" {
3459                 // nul dev
3460             }else{
3461                 if strBegins(fname,"(") {
3462                     xin,xout,err := gsh.Popen(fname,"w")
3463                     if err {
3464                         }else{
3465                             xin.Close()
3466                             defer xout.Close()
3467                             out = xout
3468                         }
3469                     }else{
3470                         // should write to temporary file
3471                         // should suppress "C" on tty
3472                         xout, err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
3473                         //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
3474                         if err != nil {
3475                             fmt.Printf("--En- PUT (%v)\n",err)
3476                         }else{
3477                             out = xout
3478                         }
3479                     }
3480                 }
3481             }
3482             fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
3483                 fname,local,err)
3484         }
3485         fmt.Printf(Elapsed(Start)+"--In- PUT %v ((%v)\n",dsz,bsize)
3486         fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n",dsz)
3487         fmt.Fprintf(clnt, "200 %v OK\r\n",dsz)
3488         fileRelay("RecvPUT",clnt,out,dsz,bsize)
3489         res = fmt.Sprintf("200 PUT done\r\n")
3490     default:
3491         res = fmt.Sprintf("400 What? %v",req)
3492     }
3493     swcc,serr := clnt.Write([]byte(res))
3494     if serr != nil {
3495         fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,serr,res)
3496     }else{
3497         fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
3498     }
3499     acconn.Close();
3500     clnt.Close();
3501 }
3502 sconn.Close();
3503 }
3504 func (gsh+GshContext)RexecClient(argv[]string)(int,string){
3505     debug := true
3506     Start := time.Now()
3507     if len(argv) == 1 {
3508         return -1, "EmptyARG"
3509     }
3510     argv = argv[1:]
3511     if argv[0] == "-serv" {
3512         gsh.RexecServer(argv[1:])
3513         return 0, "Server"
3514     }
3515     remote := "0.0.0.0:9999"
3516     if argv[0][0] == '#' {
3517         remote = argv[0][1:]
3518         argv = argv[1:]
3519     }
3520     if argv[0] == "-s" {
3521         debug = false
3522         argv = argv[1:]
3523     }
3524     dport, err := net.ResolveTCPAddr("tcp",remote);
3525     if err != nil {
3526         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
3527         return -1, "AddressError"
3528     }
3529     }
3530     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
3531     serv, err := net.DialTCP("tcp",nil,dport)
3532     if err != nil {
3533         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
3534         return -1, "CannotConnect"
3535     }
3536     if debug {
3537         al := serv.LocalAddr()
3538         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
3539     }
3540     req := ""
3541     res := make([]byte,LINESIZE)

```

```

3540 count,err := serv.Read(res)
3541 if err != nil {
3542     fmt.Printf("--En- S: (%d,%v) %v",count,err,string(res))
3543 }
3544 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }
3545
3546 if argv[0] == "GET" {
3547     savPA := gsh.gshPA
3548     var bsize int = 64*1024
3549     req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
3550     fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
3551     fmt.Fprintf(serv,req)
3552     count,err = serv.Read(res)
3553     if err != nil {
3554     }else{
3555         var dsize int64 = 0
3556         var out *os.File = nil
3557         var out_tobeclosed *os.File = nil
3558         var fname string = ""
3559         var rcode int = 0
3560         var pid int = -1
3561         fmt.Sscanf(string(res),"%d %d",&rcode,&dsize)
3562         fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
3563         if 3 <= len(argv) {
3564             fname = argv[2]
3565             if strBegins(fname,"{") {
3566                 xin,xout,err := gsh.Popen(fname,"w")
3567                 if err {
3568                     }else{
3569                         xin.Close()
3570                         defer xout.Close()
3571                         out = xout
3572                         out_tobeclosed = xout
3573                         pid = 0 // should be its pid
3574                     }
3575                 }else{
3576                     // should write to temporary file
3577                     // should suppress "C on tty
3578                     xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
3579                     if err != nil {
3580                         fmt.Printf("--En- %v\n",err)
3581                     }
3582                     out = xout
3583                     //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
3584                 }
3585             }
3586             in_ := serv.File()
3587             fileRelay("RecvGET",in,out,dsize,bsize)
3588             if 0 <= pid {
3589                 gsh.gshPA = savPA // recovery of Fd(), and more?
3590                 fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
3591                 out_tobeclosed.Close()
3592                 //syscall.Wait4(pid,nil,0,nil) //##
3593             }
3594         }
3595     }else
3596     if argv[0] == "PUT" {
3597         remote_ := serv.File()
3598         var local *os.File = nil
3599         var dsize int64 = 32*1024*1024
3600         var bsize int = 64*1024
3601         var ofile string = ""
3602         //fmt.Printf("--I-- Rex %v\n",argv)
3603         if 1 < len(argv) {
3604             fname := argv[1]
3605             if strBegins(fname,"-z") {
3606                 fmt.Sscanf(fname[2:], "%d",&dsize)
3607             }else
3608             if strBegins(fname,"{") {
3609                 xin,xout,err := gsh.Popen(fname,"r")
3610                 if err {
3611                     }else{
3612                         xout.Close()
3613                         defer xin.Close()
3614                         //in = xin
3615                         local = xin
3616                         fmt.Printf("--In- {&d} < Upload output of %v\n",
3617                             local.Fd(),fname)
3618                         ofile = "-from."+fname
3619                         dsize = MaxStreamSize
3620                     }
3621                 }else{
3622                     xlocal,err := os.Open(fname)
3623                     if err != nil {
3624                         fmt.Printf("--En- (%s)\n",err)
3625                         local = nil
3626                     }else{
3627                         local = xlocal
3628                         fi := local.Stat()
3629                         dsize = fi.Size()
3630                         defer local.Close()
3631                         //fmt.Printf("--I-- Rex in(%v / %v)\n",ofile,dsize)
3632                     }
3633                     ofile = fname
3634                     fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
3635                         fname,dsize,local,err)
3636                 }
3637             }
3638             if 2 < len(argv) && argv[2] != "" {
3639                 ofile = argv[2]
3640                 //fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
3641             }
3642             //fmt.Printf(Elapsed(Start)+"--I-- Rex out(%v)\n",ofile)
3643             fmt.Printf(Elapsed(Start)+"--In- PUT %v (%/%)\n",dsize,bsize)
3644             req = fmt.Sprintf("PUT %v %v \r\n",dsize,ofile)
3645             if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
3646             fmt.Fprintf(serv,"%v",req)
3647             count,err = serv.Read(res)
3648             if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
3649             fileRelay("SendPUT",local,remote,dsize,bsize)
3650         }else{
3651             req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
3652             if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
3653             fmt.Fprintf(serv,"%v",req)
3654             //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
3655         }
3656         //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
3657         count,err = serv.Read(res)
3658         res := ""
3659         if count == 0 {
3660             res = "(nil)\r\n"
3661         }else{
3662             res = string(res[:count])
3663         }
3664         if err != nil {
3665             fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
3666         }else{
3667             fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
3668         }
3669         serv.Close()
3670         //conn.Close()
3671
3672         var stat string
3673         var rcode int
3674         fmt.Sscanf(res,"%d %s",&rcode,&stat)
3675         //fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
3676         return rcode,ress
3677     }
3678
3679 // <a name="remote-sh">Remote Shell</a>
3680 // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
3681 func (gsh*GshContext)FileCopy(argv[]string){
3682     var host = ""
3683     var port = ""
3684     var upload = false
3685     var download = false
3686     var xargv = []string{"rex-gcp"}
3687     var srcv = []string{}
3688     var dstv = []string{}
3689     argv = argv[1:]
3690
3691     for _,v := range argv {
3692         //
3693         if v[0] == '-' { // might be a pseudo file (generated date)
3694             continue
3695         }
3696         //
3697         obj := strings.Split(v,":")
3698         //fmt.Printf("%d %v %v\n",len(obj),v,obj)
3699         if 1 < len(obj) {
3700             host = obj[0]
3701             file := ""
3702             if 0 < len(host) {
3703                 gsh.LastServer.host = host
3704             }else{
3705                 host = gsh.LastServer.host
3706                 port = gsh.LastServer.port
3707             }
3708             if 2 < len(obj) {
3709                 port = obj[1]
3710                 if 0 < len(port) {
3711                     gsh.LastServer.port = port
3712                 }else{
3713                     port = gsh.LastServer.port
3714                 }
3715                 file = obj[2]
3716             }else{

```

```

3717         file = obj[1]
3718     }
3719     if len(srcv) == 0 {
3720         download = true
3721         srcv = append(srcv, file)
3722         continue
3723     }
3724     upload = true
3725     dstv = append(dstv, file)
3726     continue
3727 }
3728 /*
3729 idx := strings.Index(v, ":")
3730 if 0 <= idx {
3731     remote = v[0:idx]
3732     if len(srcv) == 0 {
3733         download = true
3734         srcv = append(srcv, v[idx+1:])
3735         continue
3736     }
3737     upload = true
3738     dstv = append(dstv, v[idx+1:])
3739     continue
3740 }
3741 }
3742 if download {
3743     dstv = append(dstv, v)
3744 }else{
3745     srcv = append(srcv, v)
3746 }
3747 }
3748 hostport := "@" + host + ":" + port
3749 if upload {
3750     if host != "" { xargv = append(xargv, hostport) }
3751     xargv = append(xargv, "PUT")
3752     xargv = append(xargv, srcv[0:]...)
3753     xargv = append(xargv, dstv[0:]...)
3754     //fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v // %v\n", hostport, dstv, srcv, xargv)
3755     fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v\n", hostport, dstv, srcv)
3756     gsh.RexecClient(xargv)
3757 }else
3758 if download {
3759     if host != "" { xargv = append(xargv, hostport) }
3760     xargv = append(xargv, "GET")
3761     xargv = append(xargv, srcv[0:]...)
3762     xargv = append(xargv, dstv[0:]...)
3763     //fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v // %v\n", hostport, srcv, dstv, xargv)
3764     fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v\n", hostport, srcv, dstv)
3765     gsh.RexecClient(xargv)
3766 }else{
3767 }
3768 }
3769 }
3770 // target
3771 func (gsh*GshContext)Trelpath(rloc string)(string){
3772     cwd, _ := os.Getwd()
3773     os.Chdir(gsh.RWD)
3774     os.Chdir(rloc)
3775     twd, _ := os.Getwd()
3776     os.Chdir(cwd)
3777 }
3778 tpath := twd + "/" + rloc
3779 return tpath
3780 }
3781 // join to remote GShell - [user@]host[:port] or cd host[:port]:path
3782 func (gsh*GshContext)Rjoin(argv []string){
3783     if len(argv) <= 1 {
3784         fmt.Printf("--I-- current server = %v\n", gsh.RSERV)
3785         return
3786     }
3787     serv := argv[1]
3788     servv := strings.Split(serv, ":")
3789     if 1 <= len(servv) {
3790         if servv[0] == "lo" {
3791             servv[0] = "localhost"
3792         }
3793     }
3794     switch len(servv) {
3795     case 1:
3796         //if strings.Index(serv, ":") < 0 {
3797             serv = servv[0] + ":" + fmt.Sprintf("%d", GSH_PORT)
3798         //}
3799     case 2: // host:port
3800         serv = strings.Join(servv, ":")
3801     }
3802     xargv := []string{"rex-join", "@"+serv, "HELO"}
3803     rcode, stat := gsh.RexecClient(xargv)
3804     if (rcode / 100) == 2 {
3805         fmt.Printf("--I-- OK Joined (%v) %v\n", rcode, stat)
3806         gsh.RSERV = serv
3807     }else{
3808         fmt.Printf("--I-- NG, could not joined (%v) %v\n", rcode, stat)
3809     }
3810 }
3811 func (gsh*GshContext)Rexec(argv []string){
3812     if len(argv) <= 1 {
3813         fmt.Printf("--I-- rexec command [ | {command} ]\n", gsh.RSERV)
3814         return
3815     }
3816     /*
3817     nargv := gsh.ScanArg(strings.Join(argv, " "), 0)
3818     fmt.Printf("--D-- nargc=%d (%v)\n", len(nargv), nargv)
3819     if nargv[1][0] != '{' {
3820         nargv[1] = "{" + nargv[1] + "}"
3821         fmt.Printf("--D-- nargc=%d (%v)\n", len(nargv), nargv)
3822     }
3823     argv = nargv
3824     */
3825     nargv := []string{}
3826     nargv = append(nargv, "{"+strings.Join(argv[1:], " ")+"}")
3827     fmt.Printf("--D-- nargc=%d (%v)\n", len(nargv), nargv)
3828     argv = nargv
3829 }
3830 xargv := []string{"rex-exec", "@"+gsh.RSERV, "GET"}
3831 xargv = append(xargv, argv...)
3832 xargv = append(xargv, "dev/tty")
3833 rcode, stat := gsh.RexecClient(xargv)
3834 if (rcode / 100) == 2 {
3835     fmt.Printf("--I-- OK Rexec (%v) %v\n", rcode, stat)
3836 }else{
3837     fmt.Printf("--I-- NG Rexec (%v) %v\n", rcode, stat)
3838 }
3839 }
3840 func (gsh*GshContext)Rchdir(argv []string){
3841     if len(argv) <= 1 {
3842         return
3843     }
3844     cwd, _ := os.Getwd()
3845     os.Chdir(gsh.RWD)
3846     os.Chdir(argv[1])
3847     twd, _ := os.Getwd()
3848     gsh.RWD = twd
3849     fmt.Printf("--I-- JWD=%v\n", twd)
3850     os.Chdir(cwd)
3851 }
3852 func (gsh*GshContext)Rpwd(argv []string){
3853     fmt.Printf("%v\n", gsh.RWD)
3854 }
3855 func (gsh*GshContext)Rls(argv []string){
3856     cwd, _ := os.Getwd()
3857     os.Chdir(gsh.RWD)
3858     argv[0] = "-ls"
3859     gsh.XFind(argv)
3860     os.Chdir(cwd)
3861 }
3862 func (gsh*GshContext)Rput(argv []string){
3863     var local string = ""
3864     var remote string = ""
3865     if 1 < len(argv) {
3866         local = argv[1]
3867         remote = local // base name
3868     }
3869     if 2 < len(argv) {
3870         remote = argv[2]
3871     }
3872     fmt.Printf("--I-- jput from=%v to=%v\n", local, gsh.Trelpath(remote))
3873 }
3874 func (gsh*GshContext)Rget(argv []string){
3875     var remote string = ""
3876     var local string = ""
3877     if 1 < len(argv) {
3878         remote = argv[1]
3879         local = remote // base name
3880     }
3881     if 2 < len(argv) {
3882         local = argv[2]
3883     }
3884     fmt.Printf("--I-- jget from=%v to=%v\n", gsh.Trelpath(remote), local)
3885 }
3886 }
3887 // <a name="network">network</a>
3888 // -s, -si, -so // bi-directional, source, sync (maybe socket)
3889 func (gsh*GshContext)Rconnect(inTCP bool, argv []string) {
3890     gshA := gshCtx.gshA
3891     if len(argv) < 2 {
3892         fmt.Printf("Usage: -s [host]:[port[:udp]]\n")
3893     }

```

```

3894     return
3895 }
3896 remote := argv[1]
3897 if remote == "" { remote = "0.0.0.0:9999" }
3898
3899 if inTCP { // TCP
3900     dport, err := net.ResolveTCPAddr("tcp",remote);
3901     if err != nil {
3902         fmt.Printf("Address error: %s (%s)\n",remote,err)
3903         return
3904     }
3905     conn, err := net.DialTCP("tcp",nil,dport)
3906     if err != nil {
3907         fmt.Printf("Connection error: %s (%s)\n",remote,err)
3908         return
3909     }
3910     file, _ := conn.File();
3911     //fd := file.Fd()
3912     //fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)
3913     fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,file.Fd())
3914
3915     savfd := gshPA.Files[1]
3916     //gshPA.Files[1] = fd;
3917     gshPA.Files[1] = file;
3918     gshCtx.gshellv(argv[2:])
3919     gshPA.Files[1] = savfd
3920     file.Close()
3921     conn.Close()
3922 }else{
3923     //dport, err := net.ResolveUDPAddr("udp4",remote);
3924     dport, err := net.ResolveUDPAddr("udp",remote);
3925     if err != nil {
3926         fmt.Printf("Address error: %s (%s)\n",remote,err)
3927         return
3928     }
3929     //conn, err := net.DialUDP("udp4",nil,dport)
3930     conn, err := net.DialUDP("udp",nil,dport)
3931     if err != nil {
3932         fmt.Printf("Connection error: %s (%s)\n",remote,err)
3933         return
3934     }
3935     file, _ := conn.File();
3936     //fd := file.Fd()
3937
3938     ar := conn.RemoteAddr()
3939     //al := conn.LocalAddr()
3940     fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
3941         //remote,ar.String(),fd)
3942         remote,ar.String(),file.Fd())
3943
3944     savfd := gshPA.Files[1]
3945     //gshPA.Files[1] = fd;
3946     gshPA.Files[1] = file;
3947     gshCtx.gshellv(argv[2:])
3948     gshPA.Files[1] = savfd
3949     file.Close()
3950     conn.Close()
3951 }
3952 }
3953 func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
3954     gshPA := gshCtx.gshPA
3955     if len(argv) < 2 {
3956         fmt.Printf("Usage: -ac [host]:[port.[udp]]\n")
3957         return
3958     }
3959     local := argv[1]
3960     if local == "" { local = "0.0.0.0:9999" }
3961     if inTCP { // TCP
3962         port, err := net.ResolveTCPAddr("tcp",local);
3963         if err != nil {
3964             fmt.Printf("Address error: %s (%s)\n",local,err)
3965             return
3966         }
3967         //fmt.Printf("Listen at %s...\n",local);
3968         sconn, err := net.ListenTCP("tcp", port)
3969         if err != nil {
3970             fmt.Printf("Listen error: %s (%s)\n",local,err)
3971             return
3972         }
3973         //fmt.Printf("Accepting at %s...\n",local);
3974         aconn, err := sconn.AcceptTCP()
3975         if err != nil {
3976             fmt.Printf("Accept error: %s (%s)\n",local,err)
3977             return
3978         }
3979         file, _ := aconn.File()
3980         //fd := file.Fd()
3981         //fmt.Printf("Accepted TCP at %s [%d]\n",local,fd)
3982         fmt.Printf("Accepted TCP at %s [%d]\n",local,file.Fd())
3983
3984         savfd := gshPA.Files[0]
3985         //gshPA.Files[0] = fd;
3986         gshPA.Files[0] = file;
3987         gshCtx.gshellv(argv[2:])
3988         gshPA.Files[0] = savfd
3989
3990         sconn.Close();
3991         aconn.Close();
3992         file.Close();
3993     }else{
3994         //port, err := net.ResolveUDPAddr("udp4",local);
3995         port, err := net.ResolveUDPAddr("udp",local);
3996         if err != nil {
3997             fmt.Printf("Address error: %s (%s)\n",local,err)
3998             return
3999         }
4000         //fmt.Printf("Listen UDP at %s...\n",local);
4001         //uconn, err := net.ListenUDP("udp4", port)
4002         uconn, err := net.ListenUDP("udp", port)
4003         if err != nil {
4004             fmt.Printf("Listen error: %s (%s)\n",local,err)
4005             return
4006         }
4007         file, _ := uconn.File()
4008         //fd := file.Fd()
4009         ar := uconn.RemoteAddr()
4010         remote := ""
4011         if ar != nil { remote = ar.String() }
4012         if remote == "" { remote = "?" }
4013
4014         // not yet received
4015         //fmt.Printf("Accepted at %s [%d] <- %s\n",local,fd,")
4016
4017         savfd := gshPA.Files[0]
4018         //gshPA.Files[0] = fd;
4019         gshPA.Files[0] = file;
4020         savenv := gshPA.Env
4021         gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
4022         gshCtx.gshellv(argv[2:])
4023         gshPA.Env = savenv
4024         gshPA.Files[0] = savfd
4025
4026         uconn.Close();
4027         file.Close();
4028     }
4029 }
4030
4031 // empty line command
4032 func (gshCtx*GshContext)xPwd(argv []string){
4033     // execute context command, pwd + date
4034     // context notation, representation scheme, to be resumed at re-login
4035     cwd, _ := os.Getwd()
4036     switch {
4037     case isin("-a",argv):
4038         gshCtx.ShowChdirHistory(argv)
4039     case isin("-ls",argv):
4040         showFileInfo(cwd,argv)
4041     default:
4042         fmt.Printf("%s\n",cwd)
4043     case isin("-v",argv): // obsolete empty command
4044         t := time.Now()
4045         date := t.Format(time.UnixDate)
4046         exe, _ := os.Executable()
4047         host, _ := os.Hostname()
4048         fmt.Printf("{PWD=\"%s\"}\n",cwd)
4049         fmt.Printf("HOST=\"%s\"}\n",host)
4050         fmt.Printf("DATE=\"%s\"}\n",date)
4051         fmt.Printf("TIME=\"%s\"}\n",t.String())
4052         fmt.Printf("PID=\"%d\"}\n",os.Getpid())
4053         fmt.Printf("EXE=\"%s\"}\n",exe)
4054         fmt.Printf("}\n")
4055     }
4056 }
4057
4058 // <a name="history">History</a>
4059 // these should be browsed and edited by HTTP browser
4060 // show the time of command with -t and direcotry with -ls
4061 // openfile-history, sort by -a -m -c
4062 // sort by elapsed time by -t -s
4063 // search by "more" like interface
4064 // edit history
4065 // sort history, and wc or uniq
4066 // CPU and other resource consumptions
4067 // limit showing range (by time or so)
4068 // export / import history
4069 func (gshCtx *GshContext)xHistory(argv []string){
4070     atWorkDirX := -1

```

```

4071 if l < len(argv) && strBegins(argv[l],"@") {
4072     atWorkDirX,_ = strconv.Atoi(argv[l][1:])
4073 }
4074 //fmt.Printf("--D-- showHistory(%v)\n",argv)
4075 for i, v := range gshCtx.CommandHistory {
4076     // exclude commands not to be listed by default
4077     // internal commands may be suppressed by default
4078     if v.CmdLine == "" && !isin("-a",argv) {
4079         continue;
4080     }
4081     if 0 <= atWorkDirX {
4082         if v.WorkDirX != atWorkDirX {
4083             continue
4084         }
4085     }
4086     if !isin("-n",argv){ // like "fc"
4087         fmt.Printf("%1%-2d ",i)
4088     }
4089     if !isin("-v",argv){
4090         fmt.Println(v) // should be with it date
4091     }else{
4092         if !isin("-l",argv) || !isin("-l0",argv) {
4093             elps := v.EndAt.Sub(v.StartAt);
4094             start := v.StartAt.Format(time.Stamp)
4095             fmt.Printf("%d ",v.WorkDirX)
4096             fmt.Printf("[%v] %11v/t ",start,elps)
4097         }
4098         if !isin("-l",argv) && !isin("-l0",argv){
4099             fmt.Printf("%v",Rusagef("%t %u/t// %s",argv,v.Rusagev))
4100         }
4101         if !isin("-at",argv) { // !isin("-ls",argv){
4102             dhi := v.WorkDirX // workdir history index
4103             fmt.Printf("%d %s\t",dhi,v.WorkDir)
4104             // show the FileInfo of the output command??
4105         }
4106         fmt.Printf("%s",v.CmdLine)
4107         fmt.Printf("\n")
4108     }
4109 }
4110 }
4111 // ln - history index
4112 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
4113     if gline[0] == '!' {
4114         hix, err := strconv.Atoi(gline[1:])
4115         if err != nil {
4116             fmt.Printf("--E-- (%s : range)\n",hix)
4117             return "", false, true
4118         }
4119         if hix < 0 || len(gshCtx.CommandHistory) <= hix {
4120             fmt.Printf("--E-- (%d : out of range)\n",hix)
4121             return "", false, true
4122         }
4123         return gshCtx.CommandHistory[hix].CmdLine, false, false
4124     }
4125     // search
4126     //for l, v := range gshCtx.CommandHistory {
4127     //}
4128     return gline, false, false
4129 }
4130 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
4131     if 0 <= hix && hix < len(gsh.CommandHistory) {
4132         return gsh.CommandHistory[hix].CmdLine,true
4133     }
4134     return "",false
4135 }
4136 // temporary adding to PATH environment
4137 // cd name -lib for LD_LIBRARY_PATH
4138 // chdir with directory history (date + full-path)
4139 // -s for sort option (by visit date or so)
4140 func (gsh*GshContext)ShowChdirHistory(i int,v GChdirHistory, argv []string){
4141     fmt.Printf("%1%-2d ",v.CmdIndex) // the first command at this WorkDir
4142     fmt.Printf("%d ",i)
4143     fmt.Printf("[%v]",v.MovedAt.Format(time.Stamp))
4144     showFileInfo(v.Dir,argv)
4145 }
4146 func (gsh*GshContext)ShowChdirHistory(argv []string){
4147     for i, v := range gsh.ChdirHistory {
4148         gsh.ShowChdirHistory(i,v,argv)
4149     }
4150 }
4151 }
4152 func skipOpts(argv[]string)(int){
4153     for i,v := range argv {
4154         if strBegins(v,"-") {
4155             return i
4156         }
4157     }
4158     return -1
4159 }
4160 }
4161 func (gshCtx*GshContext)xChdir(argv []string){
4162     cdhist := gshCtx.ChdirHistory
4163     if !isin("-?",argv) || !isin("-l",argv) || !isin("-a",argv) {
4164         gshCtx.ShowChdirHistory(argv)
4165         return
4166     }
4167     pwd, _ := os.Getwd()
4168     dir := ""
4169     if len(argv) <= 1 {
4170         dir = toFullpath("-")
4171     }else{
4172         i := skipOpts(argv[1:])
4173         if i < 0 {
4174             dir = toFullpath("-")
4175         }else{
4176             dir = argv[i+1]
4177         }
4178     }
4179     if strBegins(dir,"@") {
4180         if dir == "@0" { // obsolete
4181             dir = gshCtx.StartDir
4182         }else
4183         if dir == "@1" {
4184             index := len(cdhist) - 1
4185             if 0 < index { index -= 1 }
4186             dir = cdhist[index].Dir
4187         }else{
4188             index, err := strconv.Atoi(dir[1:])
4189             if err != nil {
4190                 fmt.Printf("--E-- xChdir(%v)\n",err)
4191                 dir = "?"
4192             }else
4193             if len(gshCtx.ChdirHistory) <= index {
4194                 fmt.Printf("--E-- xChdir(history range error)\n")
4195                 dir = "?"
4196             }else{
4197                 dir = cdhist[index].Dir
4198             }
4199         }
4200     }
4201     if dir != "?" {
4202         err := os.Chdir(dir)
4203         if err != nil {
4204             fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
4205         }else{
4206             cwd, _ := os.Getwd()
4207             if cwd != pwd {
4208                 hist1 := GChdirHistory { }
4209                 hist1.Dir = cwd
4210                 hist1.MovedAt = time.Now()
4211                 hist1.CmdIndex = len(gshCtx.CommandHistory)+1
4212                 gshCtx.ChdirHistory = append(cdhist,hist1)
4213                 if !isin("-s",argv){
4214                     //cwd, _ := os.Getwd()
4215                     //fmt.Printf("%s\n",cwd)
4216                     ix := len(gshCtx.ChdirHistory)-1
4217                     gshCtx.ShowChdirHistoryl(ix,hist1,argv)
4218                 }
4219             }
4220         }
4221     }
4222     if !isin("-ls",argv){
4223         cwd, _ := os.Getwd()
4224         showFileInfo(cwd,argv);
4225     }
4226 }
4227 }
4228 func TimeValSub(tv1 *time.Duration, tv2 *time.Duration){
4229     //tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
4230     *tv1 -= tv2;
4231 }
4232 }
4233 func RusageSub(rv1, rv2 [2]aRusage){[2]aRusage){
4234     TimeValSub(&rv1[0].Utime,&rv2[0].Utime)
4235     TimeValSub(&rv1[1].Stime,&rv2[1].Stime)
4236     return rv1
4237 }
4238 }
4239 func TimeValAdd(tv1 time.Duration, tv2 time.Duration)(time.Duration){
4240     //tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
4241     tvs := tv1 + tv2;
4242     return tvs;
4243 }
4244 }
4245 func RusageAdd(rv1, rv2 [2]aRusage){[2]aRusage){
4246     TimeValAdd(&rv1[0].Utime,&rv2[0].Utime)
4247     TimeValAdd(&rv1[1].Stime,&rv2[1].Stime)
4248     return rv1
4249 }

```



```

4248 TimeValAdd(ru[1].Stime,ru2[1].Stime)
4249 return ru1
4250 }
4251 */
4252 // <a name="rusage">Resource Usage</a>
4253 func sRusage(fmts string, argv []string, ru [2]aRusage)(string){
4254 // ru[0] self , ru[1] children
4255 ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
4256 st := TimeValAdd(ru[0].Stime,ru[1].Stime)
4257 //uu := (int64(ut.Sec)*1000000 + int64(ut.Usec)) * 1000
4258 //su := (int64(st.Sec)*1000000 + int64(st.Usec)) * 1000
4259 uu := ut // in nano sec
4260 su := st // in nano sec
4261 tu := uu + su
4262 ret := fmt.Sprintf("%v/sum",abftime(tu))
4263 ret := fmt.Sprintf("%v/uss", abftime(uss))
4264 ret += fmt.Sprintf("%v/sys",abftime(su))
4265 return ret
4266 }
4267 func Rusage(fmts string, argv []string, ru [2]aRusage)(string){
4268 ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
4269 st := TimeValAdd(ru[0].Stime,ru[1].Stime)
4270 //fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
4271 //fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
4272 //fmt.Printf("%d.%06ds/s ",ut/1000000000,(ut/1000000000));
4273 //fmt.Printf("%d.%06ds/s ",st/1000000000,(st/1000000000));
4274 return ""
4275 }
4276 }
4277 func Getrusage()(2]aRusage){
4278 var ruv = [2]aRusage{}
4279 aGetrusage(aRUSAGE_SELF,&ruv[0])
4280 aGetrusage(aRUSAGE_CHILDREN,&ruv[1])
4281 return ruv
4282 }
4283 func (gshCtx *GshContext)xTime(argv []string)(bool){
4284 if 2 < len(argv){
4285 gshCtx.LastRusage = aRusage{}
4286 rusagev1 := Getrusagev()
4287 fin := gshCtx.gshellv(argv[1:])
4288 rusagev2 := Getrusagev()
4289 showRusage(argv[1],argv,&gshCtx.LastRusage)
4290 rusagev := RusageSubv(rusagev2,rusagev1)
4291 showRusage("self",argv,&rusagev[0])
4292 showRusage("child",argv,&rusagev[1])
4293 return fin
4294 }else{
4295 rusage:= aRusage {}
4296 aGetrusage(aRUSAGE_SELF,&rusage)
4297 showRusage("self",argv, &rusage)
4298 aGetrusage(aRUSAGE_CHILDREN,&rusage)
4299 showRusage("child",argv, &rusage)
4300 return false
4301 }
4302 }
4303 func (gshCtx *GshContext)xJobs(argv []string){
4304 fmt.Printf("%d Jobs\n",len(gshCtx.BackGroundJobs))
4305 for j, pid := range gshCtx.BackGroundJobs {
4306 //wstat := syscall.WaitStatus {}
4307 rusage := aRusage {}
4308 //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
4309 //wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
4310 }
4311 wpid := pid.Pid();
4312 err := errors.New("stab_NoError");
4313 }
4314 if err != nil {
4315 fmt.Printf("--E-- %d [%d] (%v)\n",j,wpid,err)
4316 }else{
4317 fmt.Printf("###[%d]\n",j,wpid)
4318 showRusage("child",argv,&rusage)
4319 }
4320 }
4321 }
4322 }
4323 func (gsh*GshContext)inBackground(argv []string)(bool){
4324 if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n", argv) }
4325 gsh.BackGround = true // set background option
4326 xfin := false
4327 xfin = gsh.gshellv(argv)
4328 gsh.BackGround = false
4329 return xfin
4330 }
4331 // -o file without command means just opening it and refer by #N
4332 // should be listed by "files" command
4333 func (gshCtx*GshContext)xOpen(argv []string){
4334 //var pv = []int{-1,-1}
4335 //err := syscall.Pipe(pv)
4336 //fmt.Printf("--I-- pipe()=[%d,%d](%v)\n",pv[0],pv[1],err)
4337 pin,pout,err := os.Pipe();
4338 fmt.Printf("--I-- pipe()=[%d,%d](%v)\n",pin.Fd(),pout.Fd(),err)
4339 }
4340 func (gshCtx*GshContext)fromPipe(argv []string){
4341 }
4342 func (gshCtx*GshContext)xClose(argv []string){
4343 }
4344 }
4345 // <a name="redirect">redirect</a>
4346 func (gshCtx*GshContext)redirect(argv []string)(bool){
4347 if len(argv) < 2 {
4348 return false
4349 }
4350 cmd := argv[0]
4351 fname := argv[1]
4352 var file *os.File = nil
4353 fdix := 0
4354 mode := os.O_RDONLY
4355 switch {
4356 case cmd == "-i" || cmd == "<":
4357 fdix = 0
4358 mode = os.O_RDONLY
4359 case cmd == "-o" || cmd == ">":
4360 fdix = 1
4361 mode = os.O_RDWR | os.O_CREATE
4362 case cmd == "-a" || cmd == ">>":
4363 fdix = 1
4364 mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
4365 }
4366 if fname[0] == '#' {
4367 fd, err := strconv.Atoi(fname[1:])
4368 if err != nil {
4369 fmt.Printf("--E-- (%v)\n",err)
4370 return false
4371 }
4372 file = os.NewFile(uintptr(fd),"MaybePipe")
4373 }else{
4374 xfile, err := os.OpenFile(argv[1], mode, 0600)
4375 if err != nil {
4376 fmt.Printf("--E-- (%s)\n",err)
4377 return false
4378 }
4379 file = xfile
4380 }
4381 gshPA := gshCtx.gshPA
4382 savfd := gshPA.Files[fdix]
4383 //gshPA.Files[fdix] = file.Fd()
4384 gshPA.Files[fdix] = file;
4385 fmt.Printf("--I-- Opened [%d] %s\n",file.Fd(),argv[1])
4386 gshCtx.gshellv(argv[2:])
4387 gshPA.Files[fdix] = savfd
4388 return false
4389 }
4390 }
4391 }
4392 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
4393 func httpHandler(res http.ResponseWriter, req *http.Request){
4394 path := req.URL.Path
4395 fmt.Printf("--I-- Got HTTP Request(%s)\n",path)
4396 {
4397 gshCtxBuf, _ := setupGshContext()
4398 gshCtx := &gshCtxBuf
4399 fmt.Printf("--I-- %s\n",path[1:])
4400 gshCtx.tgshell(path[1:])
4401 }
4402 fmt.Fprintf(res, "Hello(^-^)/\n%s\n",path)
4403 }
4404 func (gshCtx *GshContext) httpServer(argv []string){
4405 http.HandleFunc("/", httpHandler)
4406 accept := "localhost:9999"
4407 fmt.Printf("--I-- HTTP Server Start at (%s)\n",accept)
4408 http.ListenAndServe(accept,nil)
4409 }
4410 func (gshCtx *GshContext)xGo(argv []string){
4411 go gshCtx.gshellv(argv[1:]);
4412 }
4413 func (gshCtx *GshContext) xPs(argv []string){}
4414 }
4415 // <a name="plugin">Plugin</a>
4416 // plugin [-ls [names]] to list plugins
4417 // References <a href="https://github.com/sec/plugin/">https://github.com/sec/plugin/</a> source code
4418 func (gshCtx *GshContext) whichPlugin(name string,argv []string)(pi *PluginInfo){
4419 pi = nil
4420 for _,p := range gshCtx.PluginFuncs {

```

```

4425     if p.Name == name && pi == nil {
4426         pi = &p
4427     }
4428     if !isin("-s",argv){
4429         //fmt.Printf("%v %v ",i,p)
4430         if !isin("-ls",argv){
4431             showFileInfo(p.Path,argv)
4432         }else{
4433             fmt.Printf("%s\n",p.Name)
4434         }
4435     }
4436 }
4437 return pi
4438 }
4439 func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
4440     if len(argv) == 0 || argv[0] == "-ls" {
4441         gshCtx.whichPlugin("",argv)
4442         return nil
4443     }
4444     name := argv[0]
4445     pin := gshCtx.whichPlugin(name,[]string{"-s"})
4446     if pin != nil {
4447         os.Args = argv // should be recovered?
4448         pin.Addr.(func())()
4449         return nil
4450     }
4451     sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)
4452
4453     p, err := plugin.Open(sofile)
4454     if err != nil {
4455         fmt.Printf("--E-- plugin.Open(%s) (%v)\n",sofile,err)
4456         return err
4457     }
4458     fname := "Main"
4459     f, err := p.Lookup(fname)
4460     if( err != nil ){
4461         fmt.Printf("--E-- plugin.Lookup(%s) (%v)\n",fname,err)
4462         return err
4463     }
4464     pin := PluginInfo {p,f,name,sofile}
4465     gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
4466     fmt.Printf("--I-- added (%d)\n",len(gshCtx.PluginFuncs))
4467
4468     //fmt.Printf("--I-- first call(%s:%s)\n",sofile,fname)
4469     os.Args = argv
4470     f.(func())()
4471     return err
4472 }
4473 func (gshCtx*GshContext)Args(argv[]string){
4474     for i,v := range os.Args {
4475         fmt.Printf("[%v] %v\n",i,v)
4476     }
4477 }
4478 func (gshCtx *GshContext) showVersion(argv[]string){
4479     if !isin("-l",argv) {
4480         fmt.Printf("%v/%v (%v)",NAME,VERSION,DATE);
4481     }else{
4482         fmt.Printf("%v",VERSION);
4483     }
4484     if !isin("-a",argv) {
4485         fmt.Printf("%s",AUTHOR)
4486     }
4487     if !isin("-n",argv) {
4488         fmt.Printf("\n")
4489     }
4490 }
4491
4492 // <a name="scanf">Scanf</a> // string decomposer
4493 // scanf [format] [input]
4494 func scanf(sstr string)(sstr[]string){
4495     strv = strings.Split(sstr, " ")
4496     return strv
4497 }
4498 func scanUntil(src,end string)(rstr string,leng int){
4499     idx := strings.Index(src,end)
4500     if 0 <= idx {
4501         rstr = src[0:idx]
4502         return rstr,idx+leng(end)
4503     }
4504     return src,0
4505 }
4506
4507 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
4508 func (gsh*GshContext)printVal(fmts string, vstr string, optv[]string){
4509     //vstr_err := strconv.Atoi(vstr)
4510     var ival int64 = 0
4511     n := 0
4512     err := error(nil)
4513     if strBegins(vstr, "-") {
4514         vx, _ := strconv.Atoi(vstr[1:])
4515         if vx < len(gsh.iValues) {
4516             vstr = gsh.iValues[vx]
4517         }else{
4518         }
4519     }
4520     // should use Eval()
4521     if strBegins(vstr,"0x") {
4522         n,err = fmt.Sscanf(vstr[2:], "%x",&ival)
4523     }else{
4524         n,err = fmt.Sscanf(vstr,"%d",&ival)
4525     }//fmt.Printf("--D-- n=%d err=(%v) (%s)=%v\n",n,err,vstr, ival)
4526     if n == 1 && err == nil {
4527         //fmt.Printf("--D-- formatn(%v) ival(%v)\n",fmts,ival)
4528         fmt.Printf("%s"+fmts,ival)
4529     }else{
4530         if !isin("-bn",optv){
4531             fmt.Printf("%s"+fmts,filepath.Base(vstr))
4532         }else{
4533             fmt.Printf("%s"+fmts,vstr)
4534         }
4535     }
4536 }
4537 }
4538 func (gsh*GshContext)printfv(fmts,div string,argv[]string,optv[]string,list[]string){
4539     //fmt.Printf("%d",len(list))
4540     //curfmt := "v"
4541     outlen = 0
4542     curfmt := gsh.iFormat
4543
4544     if 0 < len(fmts) {
4545         for xi := 0; xi < len(fmts); xi++ {
4546             fch := fmts[xi]
4547             if fch == '%' {
4548                 if xi+1 < len(fmts) {
4549                     curfmt = string(fmts[xi+1])
4550                     gsh.iFormat = curfmt
4551                     xi += 1
4552                 }
4553                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
4554                     vals,leng := scanUntil(fmts[xi+2:],")")
4555                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
4556                     gsh.printVal(curfmt,vals,optv)
4557                     xi += 2+leng-1
4558                     outlen += 1
4559                 }
4560                 continue
4561             }
4562             if fch == '.' {
4563                 hi,leng := scanInt(fmts[xi+1:])
4564                 if 0 < leng {
4565                     if hi < len(gsh.iValues) {
4566                         gsh.printVal(curfmt,gsh.iValues[hi],optv)
4567                         outlen += 1 // should be the real length
4568                     }else{
4569                         fmt.Printf("((out-range))")
4570                     }
4571                     xi += leng
4572                     continue;
4573                 }
4574             }
4575             fmt.Printf("%c",fch)
4576             outlen += 1
4577         }
4578     }else{
4579         //fmt.Printf("--D-- print (%s)\n")
4580         for l,v := range list {
4581             if 0 < l {
4582                 fmt.Printf(div)
4583             }
4584             gsh.printVal(curfmt,v,optv)
4585             outlen += 1
4586         }
4587     }
4588     if 0 < outlen {
4589         fmt.Printf("\n")
4590     }
4591 }
4592 }
4593 func (gsh*GshContext)Scanv(argv[]string){
4594     //fmt.Printf("--D-- Scanv(%v)\n",argv)
4595     if len(argv) == 1 {
4596         return
4597     }
4598     argv = argv[1:]
4599     fmts := ""
4600     if strBegins(argv[0],"-F") {
4601         fmts = argv[0]
4602         gsh.lDelimiter = fmts

```

```

4602     argv = argv[1:]
4603 }
4604 input := strings.Join(argv, " ")
4605 if fmts == "" { // simple decomposition
4606     v := scanv(input)
4607     gsh.iValues = v
4608     //fmt.Printf("%v\n", strings.Join(v, ","))
4609 }else{
4610     v := make([]string, 8)
4611     n, err := fmt.Sscanf(input, fmts, &v[0], &v[1], &v[2], &v[3])
4612     fmt.Printf("--D-- Scanf ->(%v) n=%d err=(%v)\n", v, n, err)
4613     gsh.iValues = v
4614 }
4615 }
4616 func (gsh*GshContext)Printv(argv []string){
4617     if false { //##0
4618         fmt.Printf("%v\n", strings.Join(argv[1:], " "))
4619         return
4620     }
4621     //fmt.Printf("--D-- Printv(%v)\n", argv)
4622     //fmt.Printf("%v\n", strings.Join(gsh.iValues, ","))
4623     div := gsh.idelimiter
4624     fmts := " "
4625     argv = argv[1:]
4626     if 0 < len(argv) {
4627         if strBegins(argv[0], "-F") {
4628             div = argv[0][2:]
4629             argv = argv[1:]
4630         }
4631     }
4632 }
4633 optv := []string{}
4634 for _, v := range argv {
4635     if strBegins(v, "-"){
4636         optv = append(optv, v)
4637         argv = argv[1:]
4638     }else{
4639         break;
4640     }
4641 }
4642 if 0 < len(argv) {
4643     fmts = strings.Join(argv, " ")
4644 }
4645 gsh.printf(fmts, div, argv, optv, gsh.iValues)
4646 }
4647 func (gsh*GshContext)Basename(argv []string){
4648     for i, v := range gsh.iValues {
4649         gsh.iValues[i] = filepath.Base(v)
4650     }
4651 }
4652 func (gsh*GshContext)Sortv(argv []string){
4653     sv := gsh.iValues
4654     sort.Slice(sv, func(i, j int) bool {
4655         return sv[i] < sv[j]
4656     })
4657 }
4658 func (gsh*GshContext)Shiftv(argv []string){
4659     vi := len(gsh.iValues)
4660     if 0 < vi {
4661         if isin("-", argv) {
4662             top := gsh.iValues[0]
4663             gsh.iValues = append(gsh.iValues[1:], top)
4664         }else{
4665             gsh.iValues = gsh.iValues[1:]
4666         }
4667     }
4668 }
4669 }
4670 func (gsh*GshContext)Enq(argv []string){
4671 }
4672 func (gsh*GshContext)Deg(argv []string){
4673 }
4674 func (gsh*GshContext)Push(argv []string){
4675     gsh.iValStack = append(gsh.iValStack, argv[1:])
4676     fmt.Printf("depth=%d\n", len(gsh.iValStack))
4677 }
4678 func (gsh*GshContext)Dump(argv []string){
4679     for i, v := range gsh.iValStack {
4680         fmt.Printf("%d %v\n", i, v)
4681     }
4682 }
4683 }
4684 func (gsh*GshContext)Pop(argv []string){
4685     depth := len(gsh.iValStack)
4686     if 0 < depth {
4687         v := gsh.iValStack[depth-1]
4688         if isin("-cat", argv){
4689             gsh.iValues = append(gsh.iValues, v...)
4690         }else{
4691             gsh.iValues = v
4692         }
4693         gsh.iValStack = gsh.iValStack[:depth-1]
4694         fmt.Printf("depth=%d %s\n", len(gsh.iValStack), gsh.iValues)
4695     }else{
4696         fmt.Printf("depth=%d\n", depth)
4697     }
4698 }
4699 }
4700 // <a name="interpreter">Command Interpreter</a>
4701 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
4702     fin = false
4703     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr, "--I-- gshellv(%d)\n", len(argv)) }
4704     if len(argv) == 0 {
4705         return false
4706     }
4707     xargv := []string{}
4708     for ai := 0; ai < len(argv); ai++ {
4709         xargv = append(xargv, strsubst(gshCtx, argv[ai], false))
4710     }
4711     argv = xargv
4712     if false {
4713         for ai := 0; ai < len(argv); ai++ {
4714             fmt.Printf("[%d] %s [%d]P\n",
4715                 ai, argv[ai], len(argv[ai]), argv[ai])
4716         }
4717     }
4718     cmd := argv[0]
4719     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr, "--I-- gshellv(%d)%v\n", len(argv), argv) }
4720     switch { // https://tour.golang.org/flowcontrol/11
4721     case cmd == "":
4722         gshCtx.xPadd([]string{}); // empty command
4723     case cmd == "-x":
4724         gshCtx.CmdTrace = ! gshCtx.CmdTrace
4725     case cmd == "-xt":
4726         gshCtx.CmdTime = ! gshCtx.CmdTime
4727     case cmd == "-ot":
4728         gshCtx.sconnect(true, argv)
4729     case cmd == "-ou":
4730         gshCtx.sconnect(false, argv)
4731     case cmd == "-it":
4732         gshCtx.saccept(true, argv)
4733     case cmd == "-iu":
4734         gshCtx.saccept(false, argv)
4735     case cmd == "-l" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
4736         gshCtx.redirect(argv)
4737     case cmd == "l":
4738         gshCtx.fromPipe(argv)
4739     case cmd == "args":
4740         gshCtx.Args(argv)
4741     case cmd == "bg" || cmd == "-bg":
4742         rfin := gshCtx.inBackground(argv[1:])
4743         return rfin
4744     case cmd == "-bn":
4745         gshCtx.Basename(argv)
4746     case cmd == "call":
4747         _, _ = gshCtx.execCommand(false, argv[1:])
4748     case cmd == "cd" || cmd == "chdir":
4749         gshCtx.xChdir(argv);
4750     case cmd == "-ksum":
4751         gshCtx.xFind(argv)
4752     case cmd == "-sum":
4753         gshCtx.xFind(argv)
4754     case cmd == "-sumtest":
4755         str := ""
4756         if 1 < len(argv) { str = argv[1] }
4757         crc := strCRC32(str, uint64(len(str)))
4758         fprintf(stderr, "%v %v\n", crc, len(str))
4759     case cmd == "close":
4760         gshCtx.xClose(argv)
4761     case cmd == "gop":
4762         gshCtx.FileCopy(argv)
4763     case cmd == "dec" || cmd == "decode":
4764         gshCtx.Dec(argv)
4765     case cmd == "#define":
4766     case cmd == "dic" || cmd == "d":
4767         xDic(argv)
4768     case cmd == "dump":
4769         gshCtx.Dump(argv)
4770     case cmd == "echo" || cmd == "e":
4771         echo(argv, true)
4772     case cmd == "enc" || cmd == "encode":
4773         gshCtx.Enc(argv)
4774     case cmd == "env":
4775         env(argv)
4776     case cmd == "eval":
4777         xEval(argv[1:], true)
4778     case cmd == "ev" || cmd == "events":

```

```

4779     dumpEvents(argv)
4780 case cmd == "exec":
4781     = gshCtx.execCommand(true,argv[1:])
4782     // should not return here
4783 case cmd == "exit" || cmd == "quit":
4784     // write Result code EXIT to 3>
4785     return true
4786 case cmd == "fds":
4787     // dump the attributes of fds (of other process)
4788 case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
4789     gshCtx.xFind(argv[1:])
4790 case cmd == "fu":
4791     gshCtx.xFind(argv[1:])
4792 case cmd == "fork":
4793     // mainly for a server
4794 case cmd == "-gen":
4795     gshCtx.gen(argv)
4796 case cmd == "-go":
4797     gshCtx.xGo(argv)
4798 case cmd == "-grep":
4799     gshCtx.xFind(argv)
4800 case cmd == "gdeg":
4801     gshCtx.Deg(argv)
4802 case cmd == "genq":
4803     gshCtx.Enq(argv)
4804 case cmd == "gpop":
4805     gshCtx.Pop(argv)
4806 case cmd == "gpush":
4807     gshCtx.Push(argv)
4808 case cmd == "history" || cmd == "hi": // hi should be alias
4809     gshCtx.xHistory(argv)
4810 case cmd == "jobs":
4811     gshCtx.xJobs(argv)
4812 case cmd == "lnsp" || cmd == "nlspr":
4813     gshCtx.SplitLine(argv)
4814 case cmd == "-le":
4815     gshCtx.xFind(argv)
4816 case cmd == "nop":
4817     // do nothing
4818 case cmd == "pipe":
4819     gshCtx.xOpen(argv)
4820 case cmd == "plug" || cmd == "plugin" || cmd == "pin":
4821     gshCtx.xPlugin(argv[1:])
4822 case cmd == "print" || cmd == "-pr":
4823     // output internal slice // also sprintf should be
4824     gshCtx.Print(argv)
4825 case cmd == "ps":
4826     gshCtx.xPs(argv)
4827 case cmd == "psitle":
4828     // to be gsh.title
4829 case cmd == "rexeed" || cmd == "rexd":
4830     gshCtx.RexeServer(argv)
4831 case cmd == "rexec" || cmd == "rex":
4832     gshCtx.RexeClient(argv)
4833 case cmd == "repeat" || cmd == "rep": // repeat cond command
4834     gshCtx.repeat(argv)
4835 case cmd == "replay":
4836     gshCtx.xReplay(argv)
4837 case cmd == "scan":
4838     // scan input (or so in fscanf) to internal slice (like Files or map)
4839     gshCtx.Scanv(argv)
4840 case cmd == "set":
4841     // set name ...
4842 case cmd == "serv":
4843     gshCtx.httpServer(argv)
4844 case cmd == "shift":
4845     gshCtx.Shiftv(argv)
4846 case cmd == "sleep":
4847     gshCtx.sleep(argv)
4848 case cmd == "-sort":
4849     gshCtx.Sortv(argv)
4850 case cmd == "j": // cmd == "join":
4851     gshCtx.Rjcin(argv)
4852 case cmd == "a": // cmd == "alpa":
4853     gshCtx.Rxec(argv)
4854 case cmd == "jcd" || cmd == "jchdir":
4855     gshCtx.Rchdir(argv)
4856 case cmd == "jget":
4857     gshCtx.Rget(argv)
4858 case cmd == "jls":
4859     gshCtx.Rls(argv)
4860 case cmd == "jput":
4861     gshCtx.Rput(argv)
4862 case cmd == "jpwd":
4863     gshCtx.Rpwd(argv)
4864 case cmd == "time":
4865     fin = gshCtx.xTime(argv)
4866 case cmd == "ungets":
4867     if 1 < len(argv) {
4868         ungets(argv[1]+"\\n")
4869     }
4870 case cmd == "pwd":
4871     gshCtx.xPwd(argv)
4872 case cmd == "ver" || cmd == "-ver" || cmd == "version":
4873     gshCtx.showVersion(argv)
4874 case cmd == "where":
4875     // data file or so?
4876 case cmd == "which":
4877     which("PATH", argv)
4878 case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
4879     go gj_server(argv[1:]);
4880 case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
4881     go gj_server(argv[1:]);
4882 case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
4883     go gj_client(argv[1:]);
4884 case cmd == "gj":
4885     jsend(argv);
4886 case cmd == "jsend":
4887     jsend(argv);
4888 default:
4889     if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
4890         gshCtx.xPlugin(argv)
4891     } else {
4892         notfound := gshCtx.execCommand(false,argv)
4893         if notfound {
4894             fmt.Printf("E-- command not found (%v)\n",cmd)
4895         }
4896     }
4897 }
4898 }
4899 }
4900 }
4901 }
4902 }
4903 }
4904 func (gsh+GshContext)gshell(gline string) (rfin bool) {
4905     argv := strings.Split(string(gline), " ")
4906     fin := gsh.gshellv(argv)
4907     return fin
4908 }
4909 func (gsh+GshContext)tgshell(gline string)(xfin bool){
4910     start := time.Now()
4911     fin := gsh.gshell(gline)
4912     end := time.Now()
4913     elps := end.Sub(start);
4914     if gsh.CmdTime {
4915         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + " (%d.%09ds)\n",
4916             elps/100000000,elps%100000000)
4917     }
4918     return fin
4919 }
4920 }
4921 func Ttyid() (int) {
4922     fi, err := os.Stdin.Stat()
4923     if err != nil {
4924         return 0;
4925     }
4926     //fmt.Printf("Stdin: %v Dev=%d\n",
4927     // fi.Mode(),fi.Mode()&os.ModeDevice)
4928     if (fi.Mode() & os.ModeDevice) != 0 {
4929         stat := aStat{};
4930         err := aFstat(0,&stat)
4931         if err != nil {
4932             //fmt.Printf("I-- Stdin: (%v)\n",err)
4933         } else {
4934             //fmt.Printf("I-- Stdin: rdev=%d %d\n",
4935             // stat.Rdev&0xFF,stat.Rdev);
4936             //fmt.Printf("I-- Stdin: tty%d\n",stat.Rdev&0xFF);
4937             return int(stat.Rdev & 0xFF)
4938         }
4939     }
4940     return 0
4941 }
4942 func (gshCtx +GshContext) ttyfile() string {
4943     //fmt.Printf("I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
4944     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tyt" +
4945         fmt.Sprintf("%02d",gshCtx.TerminalId)
4946     //strconv.Itoa(gshCtx.TerminalId)
4947     //fmt.Printf("I-- ttyfile=%s\n",ttyfile)
4948     return ttyfile
4949 }
4950 func (gshCtx +GshContext) ttyline>(*os.File){
4951     file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4952     if err != nil {
4953         fmt.Printf("I-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
4954         return file;
4955     }
4956     return file
4957 }

```

```

4956 }
4957 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
4958     if( skipping ){
4959         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4960         line, _ := reader.ReadLine()
4961         return string(line)
4962     }else
4963     if true {
4964         return xgetline(hix,prevline,gshCtx)
4965     }
4966     /*
4967     else
4968     if( with_exgetline && gshCtx.GetLine != "" ){
4969         //var xhix int64 = int64(hix); // cast
4970         newenv := os.Environ()
4971         newenv = append(newenv, "GSH_LINESIZE="+strconv.FormatInt(int64(hix),10) )
4972
4973         tty := gshCtx.ttyline()
4974         tty.WriteString(prevline)
4975         Pa := os.ProcAttr {
4976             "", // start dir
4977             newenv, //os.Environ(),
4978             []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
4979             nil,
4980         }
4981         //fmt.Printf("--I-- getline=%s // %s\n",gsh_getline[0],gshCtx.GetLine)
4982         proc, err := os.StartProcess(gsh_getline[0],[]string{"getline","getline"},&Pa)
4983         if err != nil {
4984             fmt.Printf("--F-- getline process error (%v)\n",err)
4985             // for ; ; { }
4986             return "exit (getline program failed)"
4987         }
4988         //stat, err := proc.Wait()
4989         proc.Wait()
4990         buff := make([]byte,LINESIZE)
4991         count, err := tty.Read(buff)
4992         //_, err = tty.Read(buff)
4993         //fmt.Printf("--D-- getline (%d)\n",count)
4994         if err != nil {
4995             if ! (count == 0) { // && err.String() == "EOF" } {
4996                 fmt.Printf("--E-- getline error (%s)\n",err)
4997             }
4998         }else{
4999             //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
5000         }
5001         tty.Close()
5002         gline := string(buff[0:count])
5003         return gline
5004     }else
5005     /*
5006     {
5007         // if isatty {
5008             fmt.Printf("%d",hix)
5009             fmt.Print(PROMPT)
5010             // }
5011             reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
5012             line, _ := reader.ReadLine()
5013             return string(line)
5014         }
5015     }
5016 }
5017 //== begin ===== getline
5018 /*
5019 * getline.c
5020 * 2020-0819 extracted from dog.c
5021 * getline.go
5022 * 2020-0822 ported to Go
5023 */
5024 /*
5025 package main // getline main
5026 import (
5027     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
5028     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
5029     "os" // <a href="https://golang.org/pkg/os/">os</a>
5030     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
5031     //"bytes" // <a href="https://golang.org/pkg/os/">os</a>
5032     //"os/exec" // <a href="https://golang.org/pkg/os/">os</a>
5033 )
5034 */
5035 // C language compatibility functions
5036 var errno = 0
5037 var stdin *os.File = os.Stdin
5038 var stdout *os.File = os.Stdout
5039 var stderr *os.File = os.Stderr
5040 var EOF = -1
5041 var NULL = 0
5042 type FILE os.File
5043 type StrBuff [1]byte
5044 var NULL_FP *os.File = nil
5045 var NULLSP = 0
5046 //var LINESIZE = 1024
5047
5048 func system(cmdstr string)(int){
5049     //PA := syscall.ProcAttr {
5050     PA := os.ProcAttr {
5051         "", // the starting directory
5052         os.Environ(),
5053         //[]uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
5054         []*os.File{os.Stdin,os.Stdout,os.Stderr},
5055         nil,
5056     }
5057     argv := strings.Split(cmdstr, " ")
5058     //pid,err := syscall.ForkExec(argv[0],argv,&PA)
5059     proc,err := os.StartProcess(argv[0],argv,&PA);
5060     if( err != nil ){
5061         //fmt.Printf("--Es-- system(%v)\n(%v)\n",cmdstr,err);
5062         return -1;
5063     }
5064     pstat, _ := proc.Wait();
5065     pid := pstat.Pid();
5066     if( err != nil ){
5067         fmt.Printf("--E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
5068     }
5069     //syscall.Wait4(pid,nil,0,nil)
5070     //fmt.Printf("====E== pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
5071     //
5072     /*
5073     argv := strings.Split(cmdstr, " ")
5074     fmt.Fprintf(os.Stderr, "--I-- system(%v)\n",argv)
5075     //cmd := exec.Command(argv[0],...)
5076     cmd := exec.Command(argv[0],argv[1],argv[2])
5077     cmd.Stdin = strings.NewReader("output of system")
5078     var out bytes.Buffer
5079     cmd.Stdout = &out
5080     var serr bytes.Buffer
5081     cmd.Stderr = &serr
5082     err := cmd.Run()
5083     if err != nil {
5084         fmt.Fprintf(os.Stderr, "--E-- system(%v)err(%v)\n",argv,err)
5085         fmt.Printf("ERR:%s\n",serr.String())
5086     }else{
5087         fmt.Printf("%s",out.String())
5088     }
5089     */
5090     return 0
5091 }
5092
5093 func atoi(str string)(ret int){
5094     ret,err := fmt.Sscanf(str,"%d",&ret)
5095     if err == nil {
5096         return ret
5097     }else{
5098         // should set errno
5099         return 0
5100     }
5101 }
5102
5103 func getenv(name string)(string){
5104     val,got := os.LookupEnv(name)
5105     if got {
5106         return val
5107     }else{
5108         return "?"
5109     }
5110 }
5111
5112 func strcpy(dst StrBuff, src string){
5113     var i int
5114     srcb := []byte(src)
5115     for i = 0; i < len(src) && srcb[i] != 0; i++ {
5116         dst[i] = srcb[i]
5117     }
5118     dst[i] = 0
5119 }
5120
5121 func xstrcpy(dst StrBuff, src StrBuff){
5122     dst = src
5123 }
5124
5125 func strcat(dst StrBuff, src StrBuff){
5126     dst = append(dst,src...)
5127 }
5128
5129 func strdup(str StrBuff)(string){
5130     return string(str[:strlen(str)])
5131 }
5132
5133 func strlen(str string)(int){
5134     return len(str)
5135 }
5136
5137 func strlen(str StrBuff)(int){
5138     var i int
5139     for i = 0; i < len(str) && str[i] != 0; i++ {

```

```

5133 }
5134 return i
5135 }
5136 func sizeof(data StrBuff)(int){
5137 return len(data)
5138 }
5139 func isatty(fd int)(ret int){
5140 return 1
5141 }
5142 }
5143 func fopen(file string,mode string)(fp*os.File){
5144 if mode == "r" {
5145 fp,err := os.Open(file)
5146 if (err != nil) {
5147 fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
5148 return NULL_FP;
5149 }
5150 return fp;
5151 }else{
5152 fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
5153 if (err != nil) {
5154 return NULL_FP;
5155 }
5156 return fp;
5157 }
5158 }
5159 func fclose(fp*os.File){
5160 fp.Close()
5161 }
5162 func fflush(fp *os.File)(int){
5163 return 0
5164 }
5165 func fgetc(fp*os.File)(int){
5166 var buf [1]byte
5167 err := fp.Read(buf[0:1])
5168 if (err != nil) {
5169 return EOF;
5170 }else{
5171 return int(buf[0])
5172 }
5173 }
5174 func sfgets(str*string, size int, fp*os.File)(int){
5175 buf := make(StrBuff,size)
5176 var ch int
5177 var i int
5178 for i = 0; i < len(buf)-1; i++ {
5179 ch = fgetc(fp)
5180 //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
5181 if (ch == EOF) {
5182 break;
5183 }
5184 buf[i] = byte(ch);
5185 if (ch == '\n') {
5186 break;
5187 }
5188 }
5189 buf[i] = 0
5190 //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
5191 return i
5192 }
5193 func fgets(buf StrBuff, size int, fp*os.File)(int){
5194 var ch int
5195 var i int
5196 for i = 0; i < len(buf)-1; i++ {
5197 ch = fgetc(fp)
5198 //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
5199 if (ch == EOF) {
5200 break;
5201 }
5202 buf[i] = byte(ch);
5203 if (ch == '\n') {
5204 break;
5205 }
5206 }
5207 buf[i] = 0
5208 //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
5209 return i
5210 }
5211 func fputc(ch int , fp*os.File)(int){
5212 var buf [1]byte
5213 buf[0] = byte(ch)
5214 fp.Write(buf[0:1])
5215 return 0
5216 }
5217 func fputs(buf StrBuff, fp*os.File)(int){
5218 fp.Write(buf)
5219 return 0
5220 }
5221 func xfputs(str string, fp*os.File)(int){
5222 return fputs([]byte(str),fp)
5223 }
5224 func sscanf(str StrBuff,fmts string, params ...interface{})(int){
5225 fmt.Sscanf(string(str[0:strlen(str)]),fmts,params...)
5226 return 0
5227 }
5228 func fprintf(fp*os.File,fmts string, params ...interface{})(int){
5229 fmt.Fprintf(fp,fmts,params...)
5230 return 0
5231 }
5232 }
5233 // <a name="IME">Command Line IME</a>
5234 //----- MYIME
5235 var MYIMEVER = "MYIME/0.0.2";
5236 type RomKana struct {
5237 dic string // dictionary ID
5238 pat string // input pattern
5239 out string // output pattern
5240 hit int64 // count of hit and used
5241 }
5242 var dicents = 0
5243 var romkana [1024]RomKana
5244 var Romkan []RomKana
5245 }
5246 func isinDic(str string)(int){
5247 for i,v := range Romkan {
5248 if v.pat == str {
5249 return i
5250 }
5251 }
5252 return -1
5253 }
5254 const (
5255 DIC_COM_LOAD = "im"
5256 DIC_COM_DUMP = "e"
5257 DIC_COM_LIST = "ls"
5258 DIC_COM_ENA = "en"
5259 DIC_COM_DIS = "di"
5260 )
5261 func helpDic(argv []string){
5262 out := stderr
5263 cmd := ""
5264 if 0 < len(argv) { cmd = argv[0] }
5265 fprintf(out,"-- %v Usage\n",cmd)
5266 fprintf(out,"... Commands\n")
5267 fprintf(out,"... %v %v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
5268 fprintf(out,"... %v %v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
5269 fprintf(out,"... %v %v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
5270 fprintf(out,"... %v %v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
5271 fprintf(out,"... %v %v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
5272 fprintf(out,"... Keys ... %v\n","ESC can be used for '\\\"")
5273 fprintf(out,"... \\c -- Reverse the case of the last character\n",)
5274 fprintf(out,"... \\i -- Replace input with translated text\n",)
5275 fprintf(out,"... \\j -- On/Off translation mode\n",)
5276 fprintf(out,"... \\l -- Force Lower Case\n",)
5277 fprintf(out,"... \\u -- Force Upper Case (software CapsLock)\n",)
5278 fprintf(out,"... \\v -- Show translation actions\n",)
5279 fprintf(out,"... \\x -- Replace the last input character with it Hexa-Decimal\n",)
5280 }
5281 func mbic(argv[]string){
5282 if len(argv) <= 1 {
5283 helpDic(argv)
5284 return
5285 }
5286 argv = argv[1:]
5287 var debug = false
5288 var info = false
5289 var silent = false
5290 var dump = false
5291 var builtin = false
5292 cmd := argv[0]
5293 argv = argv[1:]
5294 opt := ""
5295 arg := ""
5296 }
5297 if 0 < len(argv) {
5298 arg1 := argv[0]
5299 if arg1[0] == '-' {
5300 switch arg1 {
5301 default: fmt.Printf("--E-- Unknown option(%v)\n",arg1)
5302 return
5303 case "-b": builtin = true
5304 case "-d": debug = true
5305 case "-s": silent = true
5306 case "-v": info = true
5307 }
5308 }
5309 opt = arg1
5310 argv = argv[1:]

```

```

5310     }
5311 }
5312 }
5313 dicName := ""
5314 dicURL := ""
5315 if 0 < len(argv) {
5316     arg = argv[0]
5317     dicName = arg
5318     argv = argv[1:]
5319 }
5320 if 0 < len(argv) {
5321     dicURL = argv[0]
5322     argv = argv[1:]
5323 }
5324 if false {
5325     fprintf(stderr, "--Dd-- com(%v) opt(%v) arg(%v)\n", cmd, opt, arg)
5326 }
5327 if cmd == DIC_COM_LOAD {
5328     //dicType := ""
5329     dicBody := ""
5330     if !builtin && dicName != "" && dicURL == "" {
5331         f, err := os.Open(dicName)
5332         if err == nil {
5333             dicURL = dicName
5334         } else {
5335             f, err = os.Open(dicName+".html")
5336             if err == nil {
5337                 dicURL = dicName+".html"
5338             } else {
5339                 f, err = os.Open("gshdic-"+dicName+".html")
5340                 if err == nil {
5341                     dicURL = "gshdic-"+dicName+".html"
5342                 }
5343             }
5344         }
5345         if err == nil {
5346             var buf = make([]byte, 128*1024)
5347             count, err := f.Read(buf)
5348             f.Close()
5349             if info {
5350                 fprintf(stderr, "--Id-- ReadDic(%v,%v)\n", count, err)
5351             }
5352             dicBody = string(buf[0:count])
5353         }
5354     }
5355     if dicBody == "" {
5356         switch arg {
5357         default:
5358             dicName = "WorldDic"
5359             dicURL = "WorldDic"
5360             if info {
5361                 fprintf(stderr, "--Id-- default dictionary %v\n", dicName)
5362             }
5363         case "wnn":
5364             dicName = "WnnDic"
5365             dicURL = "WnnDic"
5366         case "sumomo":
5367             dicName = "Sumomodic"
5368             dicURL = "Sumomodic"
5369         case "sijimi":
5370             dicName = "Sijimidi"
5371             dicURL = "Sijimidi"
5372         case "jkl":
5373             dicName = "JKLJaDic"
5374             dicURL = "JA_JKLJaDic"
5375         }
5376         if debug {
5377             fprintf(stderr, "--Id-- %v URL=%v\n", dicName, dicURL)
5378         }
5379         dicv := strings.Split(dicURL, ",")
5380         if debug {
5381             fprintf(stderr, "--Id-- %v encoded data...\n", dicName)
5382             fprintf(stderr, "Type: %v\n", dicv[0])
5383             fprintf(stderr, "Body: %v\n", dicv[1])
5384             fprintf(stderr, "\n")
5385         }
5386         body, _ := base64.StdEncoding.DecodeString(dicv[1])
5387         dicBody = string(body)
5388     }
5389     if info {
5390         fmt.Printf("--Id-- %v %v\n", dicName, dicURL)
5391         fmt.Printf("%s\n", dicBody)
5392     }
5393     if debug {
5394         fprintf(stderr, "--Id-- dicName %v text...\n", dicName)
5395         fprintf(stderr, "%v\n", string(dicBody))
5396     }
5397     envv := strings.Split(dicBody, "\n");
5398     if info {
5399         fprintf(stderr, "--Id-- %v scan...\n", dicName);
5400     }
5401     var added int = 0
5402     var dup int = 0
5403     for i, v := range envv {
5404         var pat string
5405         var out string
5406         fmt.Sscanf(v, "%s %s", &pat, &out)
5407         if len(pat) <= 0 {
5408             } else {
5409                 if 0 <= isinDic(pat) {
5410                     dup += 1
5411                     continue
5412                 }
5413                 romkana{dicents} = RomKana{dicName, pat, out, 0}
5414                 dicents += 1
5415                 added += 1
5416                 Romkan = append(Romkan, RomKana{dicName, pat, out, 0})
5417                 if debug {
5418                     fmt.Printf("%3v: %2v %8-8v %2v %v\n",
5419                         i, len(pat), pat, len(out), out)
5420                 }
5421             }
5422         }
5423     }
5424     if !silent {
5425         url := dicURL
5426         if strBegins(url, "data:") {
5427             url = "builtin"
5428         }
5429         fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
5430             dicName, added, dup, len(Romkan), url);
5431     }
5432     // should sort by pattern length for complete match, for performance
5433     if debug {
5434         arg = "" // search pattern
5435         dump = true
5436     }
5437 }
5438 if cmd == DIC_COM_DUMP || dump {
5439     fprintf(stderr, "--Id-- %v dump... %v entries:\n", dicName, len(Romkan));
5440     var match = 0
5441     for i := 0; i < len(Romkan); i++ {
5442         dic := Romkan[i].dic
5443         pat := Romkan[i].pat
5444         out := Romkan[i].out
5445         if arg == "" || 0 <= strings.Index(pat, arg) || 0 <= strings.Index(out, arg) {
5446             fmt.Printf("\\\\ %v %v %2v %8-8v %2v %v\n",
5447                 i, dic, len(pat), pat, len(out), out)
5448             match += 1
5449         }
5450     }
5451     fprintf(stderr, "--Id-- %v matched %v / %v entries:\n", arg, match, len(Romkan));
5452 }
5453 }
5454 func loadDefaultDic(dic int) {
5455     if( 0 < len(Romkan) ) {
5456         return
5457     }
5458     //fprintf(stderr, "\r\n")
5459     xDic{[]string{"dic", DIC_COM_LOAD}};
5460 }
5461 var info = false
5462 if info {
5463     fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\r\n")
5464     fprintf(stderr, "--Id-- enter \"dic\" command for help.\r\n")
5465 }
5466 }
5467 func readDic()(int) {
5468     /*
5469     var rk *os.File
5470     var dic = "MyIME-dic.txt";
5471     //rk = fopen("romkana.txt", "r");
5472     //rk = fopen("JK-JA-morse-dic.txt", "r");
5473     rk = fopen(dic, "r");
5474     if( rk == NULL_FP ) {
5475         if( true ) {
5476             fprintf(stderr, "--s-- Could not load %s\n", MyIMEVER, dic);
5477         }
5478         return -1;
5479     }
5480     if( true ) {
5481         var di int;
5482         var line = make(StrBuff, 1024);
5483         var pat string
5484         var out string
5485         for di = 0; di < 1024; di++ {
5486             if( fgets(line, sizeof(line), rk) == NULLSP ) {

```

```

5487         break;
5488     }
5489     fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
5490     //sscanf(line, "%s %[\r\n]", &pat, &out);
5491     romkana[di].pat = pat;
5492     romkana[di].out = out;
5493     //fprintf(stderr, "--Dd- %s\n", pat, out)
5494 }
5495 dicents += di
5496 if (false) {
5497     fprintf(stderr, "--%s-- loaded romkana.txt [%d]\n", MyIMEVER, di);
5498     for di = 0; di < dicents; di++ {
5499         fprintf(stderr, "%s %s\n", romkana[di].pat, romkana[di].out);
5500     }
5501 }
5502 }
5503 }
5504 fclose(rk);
5505
5506 //romkana[dicents].pat = "//ddump"
5507 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
5508 /*
5509 return 0;
5510 }
5511 func matchlen(stri string, pati string)(int){
5512 if strBegins(stri, pati) {
5513     return len(pati)
5514 }else{
5515     return 0
5516 }
5517 }
5518 func convs(src string)(string){
5519     var si int;
5520     var sx = len(src);
5521     var di int;
5522     var mi int;
5523     var dstb []byte
5524
5525     for si = 0; si < sx; { // search max. match from the position
5526         if strBegins(src[si:], "%x/") {
5527             // %x/integer/ // s/a/b/
5528             ix := strings.Index(src[si+3:], "/")
5529             if 0 < ix {
5530                 var iv int = 0
5531                 //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
5532                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
5533                 sval := fmt.Sprintf("%x", iv)
5534                 bval := []byte(sval)
5535                 dstb = append(dstb, bval...)
5536                 si = si+3+ix+1
5537                 continue
5538             }
5539         }
5540         if strBegins(src[si:], "%d/") {
5541             // %d/integer/ // s/a/b/
5542             ix := strings.Index(src[si+3:], "/")
5543             if 0 < ix {
5544                 var iv int = 0
5545                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
5546                 sval := fmt.Sprintf("%d", iv)
5547                 bval := []byte(sval)
5548                 dstb = append(dstb, bval...)
5549                 si = si+3+ix+1
5550                 continue
5551             }
5552         }
5553         if strBegins(src[si:], "%t") {
5554             now := time.Now()
5555             if true {
5556                 date := now.Format(time.Stamp)
5557                 dstb = append(dstb, []byte(date)...)
5558                 si = si+3
5559             }
5560             continue
5561         }
5562         var maxlen int = 0;
5563         var len int;
5564         mi = -1;
5565         for di = 0; di < dicents; di++ {
5566             len = matchlen(src[si:], romkana[di].pat);
5567             if (maxlen < len) {
5568                 maxlen = len;
5569                 mi = di;
5570             }
5571         }
5572         if (0 < maxlen) {
5573             out := romkana[mi].out;
5574             dstb = append(dstb, []byte(out)...);
5575             si += maxlen;
5576         }else{
5577             dstb = append(dstb, src[si])
5578             si += 1;
5579         }
5580     }
5581     return string(dstb)
5582 }
5583 func trans(src string)(int){
5584     dst := convs(src);
5585     xiputs(dst, stderr);
5586     return 0;
5587 }
5588 //----- LINEEDIT
5589 // "?" at the top of the line means searching history
5590
5591 // should be compatilbe with Telnet
5592 const (
5593     EV_MODE = 255
5594     EV_IDLE = 254
5595     EV_TIMEOUT = 253
5596
5597     GO_UP = 252 // k
5598     GO_DOWN = 251 // j
5599     GO_RIGHT = 250 // l
5600     GO_LEFT = 249 // h
5601     DEL_RIGHT = 248 // x
5602     GO_TOP = 'A'-0x40 // o
5603     GO_ENDL = 'E'-0x40 // S
5604
5605     GO_TOPW = 239 // b
5606     GO_ENDW = 238 // e
5607     GO_NEXTW = 237 // w
5608
5609     GO_PORNC = 229 // f
5610     GO_PAIRCH = 228 // %
5611
5612     GO_DEL = 219 // d
5613
5614     HI_SRCH_FW = 209 // /
5615     HI_SRCH_BK = 208 // ?
5616     HI_SRCH_RFW = 207 // n
5617     HI_SRCH_MBK = 206 // N
5618 )
5619
5620 // should return number of octets ready to be read immediately
5621 //fprintf(stderr, "\n--Select(%v %v)\n", err, r.Bits[0])
5622
5623
5624
5625 var EventRecvFd = -1 // file descriptor
5626 var EventSendFd = -1
5627 const EventFdOffset = 1000000
5628 const NormalFdOffset = 100
5629
5630 /* 2020-1021 replaced poll() with channel/select
5631 func putKeyinEvent(event int, evarg int){
5632     if true {
5633         if EventRecvFd < 0 {
5634             var pv = [int(-1), -1]
5635             syscall.Pipe(pv)
5636             EventRecvFd = pv[0]
5637             EventSendFd = pv[1]
5638             //fmt.Printf("--De-- EventPipe created[%v,%v]\n", EventRecvFd, EventSendFd)
5639         }
5640     }else{
5641         if EventRecvFd < 0 {
5642             // the document differs from this spec
5643             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
5644             sv, err := syscall.Socketpair(syscall.AF_UNIX, syscall.SOCK_STREAM, 0)
5645             EventRecvFd = sv[0]
5646             EventSendFd = sv[1]
5647             if err != nil {
5648                 fmt.Printf("--De-- EventSock created[%v,%v]\n",
5649                     EventRecvFd, EventSendFd, err)
5650             }
5651         }
5652     }
5653     var buf = []byte{ byte(event)}
5654     n, err := syscall.Write(EventSendFd, buf)
5655     if err != nil {
5656         fmt.Printf("--De-- putEvent[%v] (%3v) (%v %v)\n", EventSendFd, event, n, err)
5657     }
5658 }
5659 */
5660 func ungets(str string){
5661     for _, ch := range str {
5662         putKeyinEvent(int(ch), 0)
5663     }
5664 }

```



```

5664 }
5665 func (gsh*GshContext)xReplay(argv[]string){
5666     hix := 0
5667     tempo := 1.0
5668     xtempo := 1.0
5669     repeat := 1
5670
5671     for _,a := range argv { // tempo
5672         if strBegins(a,"x"){
5673             fmt.Sscanf(a[1:],"%f",&xtempo)
5674             tempo = 1 / xtempo
5675             //fmt.Printf(stderr,"--Dr-- tempo=%v\n",a[2:],tempo);
5676         }else
5677         if strBegins(a,"r") { // repeat
5678             fmt.Sscanf(a[1:],"%v",&repeat)
5679         }else
5680         if strBegins(a,"l") {
5681             fmt.Sscanf(a[1:],"%d",&hix)
5682         }else{
5683             fmt.Sscanf(a,"%d",&hix)
5684         }
5685     }
5686     if hix == 0 || len(argv) <= 1 {
5687         hix = len(gsh.CommandHistory)-1
5688     }
5689     fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n",hix,xtempo,repeat)
5690     //dumpEvents(hix)
5691     //gsh.xScanReplay(hix,false,repeat,tempo,argv)
5692     go gsh.xScanReplay(hix,true,repeat,tempo,argv)
5693
5694     runtime.Gosched(); // wait xScanReplay is launched
5695     //fmt.Printf("--Ir-- Replay set\n");
5696 }
5697
5698 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
5699 // 2020-0827 GShell-0.2.3
5700 /*
5701 func FpollIn1(fp *os.File,usec int)(uintptr){
5702     nfd := 1
5703
5704     rdv := syscall.FdSet {}
5705     fd1 := fp.Fd()
5706     bank1 := fd1/32
5707     mask1 := int32(1 << fd1)
5708     rdv.Bits[bank1] = mask1
5709
5710     fd2 := -1
5711     bank2 := -1
5712     var mask2 int32 = 0
5713
5714     if 0 <= EventRecvFd {
5715         fd2 = EventRecvFd
5716         nfd = fd2 + 1
5717         bank2 = fd2/32
5718         mask2 = int32(1 << fd2)
5719         rdv.Bits[bank2] |= mask2
5720         //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
5721     }
5722
5723     tout := syscall.NsecToTimeval(int64(usec*1000))
5724     //n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
5725     err := syscall.Select(nfd,&rdv,nil,nil,&tout)
5726     if err != nil {
5727         //fmt.Printf("--De-- select() err(%v)\n",err)
5728     }
5729     if err == nil {
5730         if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
5731             if false {
5732                 fmt.Printf("--De-- got Event\n")
5733             }
5734             return uintptr(EventFdOffset + fd2)
5735         }else
5736         if (rdv.Bits[bank1] & mask1) != 0 {
5737             return uintptr(NormalFdOffset + fd1)
5738         }else{
5739             return 1
5740         }
5741     }else{
5742         return 0
5743     }
5744 }
5745 */
5746 /*
5747 func fgetcTimeout1(fp *os.File,usec int)(int){
5748     READ1:
5749     //readyFd := FpollIn1(fp,usec)
5750     readyFd := CPpollIn1(fp,usec)
5751     if readyFd < 100 {
5752         return EV_TIMEOUT
5753     }
5754
5755     var buf [1]byte
5756
5757     if EventFdOffset <= readyFd {
5758         fd := int(readyFd-EventFdOffset)
5759         _err := syscall.Read(fd,buf[0:1])
5760         if( err != nil ){
5761             return EOF;
5762         }else{
5763             if buf[0] == EV_MODE {
5764                 recvKeyEvent(fd)
5765                 goto READ1
5766             }
5767             return int(buf[0])
5768         }
5769     }
5770     _err := fp.Read(buf[0:1])
5771     if( err != nil ){
5772         return EOF;
5773     }else{
5774         return int(buf[0])
5775     }
5776 }
5777 */
5778
5779 func visibleChar(ch int)(string){
5780     switch {
5781     case '!' <= ch && ch <= '-':
5782         return string(ch)
5783     }
5784     switch ch {
5785     case '\t': return "\\s"
5786     case '\n': return "\\n"
5787     case '\r': return "\\r"
5788     case '\f': return "\\f"
5789     case '\t': return "\\t"
5790     }
5791     switch ch {
5792     case 0x00: return "NUL"
5793     case 0x07: return "BEL"
5794     case 0x08: return "BS"
5795     case 0x0E: return "SO"
5796     case 0x0F: return "SI"
5797     case 0x1B: return "ESC"
5798     case 0x7F: return "DEL"
5799     }
5800     switch ch {
5801     case EV_IDLE: return fmt.Sprintf("IDLE")
5802     case EV_MODE: return fmt.Sprintf("MODE")
5803     }
5804     return fmt.Sprintf("%X",ch)
5805 }
5806 /*
5807 func recvKeyEvent(fd int){
5808     var buf = make([]byte,1)
5809     _err := syscall.Read(fd,buf[0:1])
5810     if( buf[0] != 0 ){
5811         romkanmode = true
5812     }else{
5813         romkanmode = false
5814     }
5815 }
5816 */
5817 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
5818     var Start time.Time
5819     var events = []Event{}
5820     for _,e := range Events {
5821         if hix == 0 || e.CmdIndex == hix {
5822             events = append(events,e)
5823         }
5824     }
5825     elen := len(events)
5826     if 0 < elen {
5827         if events[elen-1].event == EV_IDLE {
5828             events = events[0:elen-1]
5829         }
5830     }
5831     for r := 0; r < repeat; r++ {
5832         for i,e := range events {
5833             nano := e.when.Nanosecond()
5834             micro := nano / 1000
5835             if Start.Second() == 0 {
5836                 Start = time.Now()
5837             }
5838             diff := time.Now().Sub(Start)
5839             if replay {
5840                 if e.event != EV_IDLE {

```

```

5841 //fmt.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
5842 putKeyEvent(e.event,0)
5843 if e.event == EV_MODE { // event with arg
5844     putKeyEvent(int(e.evarg),0)
5845 }
5846 }else{
5847     //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
5848 }
5849 }else{
5850     fmt.Printf("#7.3fms #8-3v 18-3v [%v.%06d] %3v %02X %4v %10.3fms\n",
5851         float64(diff)/1000000.0,
5852         i,
5853         e.CmdIndex,
5854         e.when.Format(time.Stamp),micro,
5855         e.event,e.event,visibleChar(e.event),
5856         float64(e.evarg)/1000000.0)
5857 }
5858 if e.event == EV_IDLE {
5859     //fmt.Printf("--replay %v / %v delay\n",i,len(events));
5860     d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
5861     //nsleep(time.Duration(e.evarg))
5862     nsleep(d)
5863 }
5864 }
5865 }
5866 }
5867 func dumpEvents(arg string){
5868     hix := 0
5869     if l < len(arg) {
5870         fmt.Sscanf(arg[1],"%d",&hix)
5871     }
5872     for i,e := range Events {
5873         nano := e.when.Nanosecond()
5874         micro := nano / 1000
5875         //if e.event != EV_TIMEOUT {
5876         if hix == 0 || e.CmdIndex == hix {
5877             fmt.Printf("#8-3v 18-3v [%v.%06d] %3v %02X %4v %10.3fms\n",i,
5878                 e.CmdIndex,
5879                 e.when.Format(time.Stamp),micro,
5880                 e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
5881         }
5882         //}
5883     }
5884 }
5885 /*
5886 func fgetcTimeout(fp *os.File,usec int)(int){
5887     ch := fgetcTimeout1(fp,usec)
5888     if ch != EV_TIMEOUT {
5889         now := time.Now()
5890         if 0 < len(Events) {
5891             last := Events[len(Events)-1]
5892             dura := int64(now.Sub(last.when))
5893             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5894         }
5895         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5896     }
5897     return ch
5898 }
5899 */
5900 // 2020-1021 replaced poll() with channel/select
5901 var Kbd = make(chan int);
5902 var Kbinit = false;
5903 var evQ = make(chan int);
5904 /*
5905 func keyInput(kbd chan int, fp *os.File){
5906     for {
5907         ch := C.getc(C.stdin);
5908         if( ch == C.EOF){
5909             break;
5910         }
5911         kbd <- int(ch);
5912     }
5913 }
5914 */
5915 // https://godoc.org/golang.org/x/crypto/ssh/terminal
5916 // https://stackoverflow.com/questions/14094190/function-similar-to-getchar
5917 func keyInput(kbd chan int, tty *os.File){
5918     tmode := C.setTermRaw();
5919     defer func(){ C.setTermMode(tmode); }();
5920     if( !OnWindows ){
5921         system("/bin/stty -echo -icanon");
5922         defer func(){ system("/bin/stty echo sane"); }();
5923     }
5924     for {
5925         var rbuf []byte = make([]byte,1);
5926         if( OnWindows ){
5927             C.setTermRaw();
5928         }
5929         ,rerr := tty.Read(rbuf);
5930         if( rerr != nil ){
5931             break;
5932         }
5933         //fmt.Printf("++KBD[%X]\n",rbuf[0]);
5934         kbd <- int(rbuf[0]);
5935     }
5936     if( !OnWindows ){
5937         system("/bin/stty echo sane");
5938     }
5939 }
5940 func fgetcTimeout(fp *os.File,usec int)(int){
5941     if( !Kbinit ){
5942         Kbinit = true;
5943         go keyInput(Kbd,fp);
5944     }
5945     for {
5946         select {
5947             case <- time.After(time.Duration(usec*1000)):
5948                 //fmt.Printf("--Timeout %v us\n",usec);
5949                 return EV_TIMEOUT;
5950             case ch := <- Kbd:
5951                 //fmt.Printf("--KBD[%X]\n",ch);
5952                 // record a KeyIn(ch) Event
5953                 {
5954                     now := time.Now()
5955                     if 0 < len(Events) {
5956                         last := Events[len(Events)-1]
5957                         dura := int64(now.Sub(last.when))
5958                         Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5959                     }
5960                     Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5961                 }
5962                 return ch;
5963             case ch := <- evQ:
5964                 if( ch == EV_MODE ){
5965                     recvKeyEvent()
5966                 }else{
5967                     return ch;
5968                 }
5969         }
5970     }
5971 }
5972 func putKeyEvent(event int, evarg int){
5973     evQ <- event;
5974 }
5975 func recvKeyEvent(){
5976     ch := <- evQ;
5977     if( ch != 0 ){
5978         romkanmode = true
5979     }else{
5980         romkanmode = false
5981     }
5982 }
5983 var AtConsoleLineTop = true
5984 var TtyMaxCol = 72 // to be obtained by ioctl?
5985 var EscTimeout = (100*1000)
5986 var {
5987     MODE_VicMode bool // vi compatible command mode
5988     MODE_ShowMode bool
5989     romkanmode bool // shown translation mode, the mode to be retained
5990     MODE_Recursive bool // recursive translation
5991     MODE_CapsLock bool // software CapsLock
5992     MODE_LowerLock bool // force lower-case character lock
5993     MODE_Vinsert int // visible insert mode, should be like "I" icon in X Window
5994     MODE_Vitrace bool // output newline before translation
5995 }
5996 type IInput struct {
5997     lno int
5998     lastlno int
5999     pch []int // input queue
6000     prompt string
6001     line string
6002     right string
6003     injmode bool
6004     pinjmode bool
6005     waitingMeta string // waiting meta character
6006     LastCmd string
6007 }
6008 func (iin*IInput)Getc(timeoutUs int)(int){
6009     ch1 := EOF
6010     ch2 := EOF
6011     ch3 := EOF
6012     if( 0 < len(iin.pch) ){ // deQ
6013         ch1 = iin.pch[0]
6014         iin.pch = iin.pch[1:]
6015     }else{
6016         ch1 = fgetcTimeout(stdin,timeoutUs);
6017     }

```

```

6018 if( ch1 == 033 ){ // escape sequence
6019   ch2 = fgetcTimeout(stdin,EscTimeout);
6020   if( ch2 == EV_TIMEOUT ){
6021     }else{
6022       ch3 = fgetcTimeout(stdin,EscTimeout);
6023       if( ch3 == EV_TIMEOUT ){
6024         iin.pch = append(iin.pch,ch2) // enQ
6025       }else{
6026         switch( ch2 ){
6027           default:
6028             iin.pch = append(iin.pch,ch2) // enQ
6029             iin.pch = append(iin.pch,ch3) // enQ
6030           case '|':
6031             switch( ch3 ){
6032               case 'A': ch1 = GO_UP; // ^
6033               case 'B': ch1 = GO_DOWN; // v
6034               case 'C': ch1 = GO_RIGHT; // >
6035               case 'D': ch1 = GO_LEFT; // <
6036               case '3':
6037                 ch4 := fgetcTimeout(stdin,EscTimeout);
6038                 //fprintf(stderr,"[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
6039                 //fprintf(stderr,"[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
6040                 ch1 = DEL_RIGHT
6041             }
6042           case '\\':
6043             //ch4 := fgetcTimeout(stdin,EscTimeout);
6044             //fprintf(stderr,"[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
6045             switch( ch3 ){
6046               case '-': ch1 = DEL_RIGHT
6047             }
6048           }
6049         }
6050       }
6051     }
6052   }
6053   return ch1
6054 }
6055 func (iin*Input)clearline(){
6056   var i int
6057   fprintf(stderr,"\r");
6058   // should be ANSI ESC sequence
6059   for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
6060     fputc(' ',os.Stderr);
6061   }
6062   fprintf(stderr,"\r");
6063 }
6064 func (iin*Input)Redraw(){
6065   redraw(iin,iin.lno,iin.line,iin.right)
6066 }
6067 func redraw(iin *Input, lno int, line string, right string){
6068   inMeta := false
6069   showMeta := "" // visible Meta mode on the cursor position
6070   showLino := fmt.Sprintf("%d!", lno)
6071   InsertMark := "" // in visible insert mode
6072   if MODE_VicMode {
6073     }else
6074   if 0 < len(iin.right) {
6075     InsertMark = " "
6076   }
6077   if( 0 < len(iin.waitingMeta) ){
6078     inMeta = true
6079     if iin.waitingMeta[0] != 033 {
6080       showMeta = iin.waitingMeta
6081     }
6082   }
6083   if( romkanmode ){
6084     //romkanmark = " ";
6085   }else{
6086     //romkanmark = " ";
6087   }
6088   if MODE_ShowMode {
6089     romkan := ""
6090     inmeta := ""
6091     inveri := ""
6092     if MODE_CapsLock {
6093       inmeta = "A"
6094     }
6095     if MODE_LowerLock {
6096       inmeta = "a"
6097     }
6098     if MODE_ViTrace {
6099       inveri = "v"
6100     }
6101     if MODE_VicMode {
6102       inveri = ":"
6103     }
6104     if romkanmode {
6105       romkan = "\343\201\202"
6106       if MODE_CapsLock {
6107         inmeta = "R"
6108       }else{
6109         inmeta = "r"
6110       }
6111     }
6112     if inMeta {
6113       inmeta = "\\ "
6114     }
6115     showMode = "[+romkan+inmeta+inveri+]";
6116   }
6117   Pre := "\r" + showMode + showLino
6118   Output := ""
6119   Left := ""
6120   Right := ""
6121   if romkanmode {
6122     Left = convs(line)
6123     Right = InsertMark+convs(right)
6124   }else{
6125     Left = line
6126     Right = InsertMark+right
6127   }
6128   Output = Pre+Left
6129   if MODE_ViTrace {
6130     Output += iin.LastCmd
6131   }
6132   Output += showMeta+Right
6133   for len(Output) < TtyMaxCol { // to the max. position that may be dirty
6134     Output += " "
6135     // should be ANSI ESC sequence
6136     // not necessary just after newline
6137   }
6138   Output += Pre+Left+showMeta // to set the cursor to the current input position
6139   fprintf(stderr,"%s",Output)
6140 }
6141 if MODE_ViTrace {
6142   if 0 < len(iin.LastCmd) {
6143     iin.LastCmd = ""
6144     fprintf(stderr,"\r\n")
6145   }
6146 }
6147 AtConsoleLineTop = false
6148 //fmt.Printf("(Redraw(%v)(%v))\n",len(line),len(right));
6149 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
6150 func delHeadChar(str string)(rline string,head string){
6151   _clen := utf8.DecodeRune([]byte(str))
6152   head = string(str[:clen])
6153   return str[clen:],head
6154 }
6155 func delTailChar(str string)(rline string, last string){
6156   var i = 0
6157   var clen = 0
6158   for {
6159     ,siz := utf8.DecodeRune([]byte(str)[i:])
6160     if siz <= 0 { break }
6161     clen = siz
6162     i += siz
6163   }
6164   last = str[len(str)-clen:]
6165   return str[0:len(str)-clen],last
6166 }
6167 // 3> for output and history
6168 // 4> for keylog?
6169 // <a name="get_line">Command Line Editor</a>
6170 func xgetline(lno int, prevline string, gsh*GshContext)(string){
6171   var iin Input
6172   iin.lno = lno
6173   iin.lno = lno
6174   CmdIndex = len(gsh.CommandHistory)
6175   if( isatty(0) == 0 ){
6176     if( sifgets(&iin.line,LINESIZE,stdin) == NULL ){
6177       iin.line = "exit\n";
6178     }else{
6179       }
6180     return iin.line
6181   }
6182   if( true ){
6183     //var pts string;
6184     //pts = ptsname(0);
6185     //pts = ttyname(0);
6186     //fprintf(stderr,"--pts[0] = %s\n,pts?pts:?"");
6187   }
6188   if( false ){
6189   }
6190 }

```

```

6195     fprintf(stderr, "! ");
6196     fflush(stderr);
6197     fgets(&iin.line, LINESIZE, stdin);
6198     return iin.line
6199 }
6200 if( !OnWindows ){ system("/bin/stty -echo -icanon"); }
6201 xiine := iin.xgetlinel(prevline, gsh);
6202 if( !OnWindows ){system("/bin/stty echo sane"); }
6203 return xiine
6204 }
6205 func (iin*IInput)Translate(cmdch int){
6206     romkanmode = !romkanmode;
6207     if MODE_VItrace {
6208         fprintf(stderr, "%v\n", string(cmdch));
6209     }else
6210     if( cmdch == 'J' ){
6211         fprintf(stderr, "\r\n");
6212         iin.inJmode = true
6213     }
6214     iin.Redraw();
6215     loadDefaultDc(cmdch);
6216     iin.Redraw();
6217 }
6218 func (iin*IInput)Replace(cmdch int){
6219     iin.LastCmd = fmt.Sprintf("%v", string(cmdch))
6220     iin.Redraw();
6221     loadDefaultDc(cmdch);
6222     det := convs(iin.line+iin.right);
6223     iin.line = det
6224     iin.right = ""
6225     if( cmdch == 'I' ){
6226         fprintf(stderr, "\r\n");
6227         iin.inJmode = true
6228     }
6229     iin.Redraw();
6230 }
6231 // aa 12 alal
6232 func isAlpha(ch rune)(bool){
6233     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
6234         return true
6235     }
6236     return false
6237 }
6238 func isAlnum(ch rune)(bool){
6239     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
6240         return true
6241     }
6242     if '0' <= ch && ch <= '9' {
6243         return true
6244     }
6245     return false
6246 }
6247 }
6248 // 0.2.8 2020-0901 created
6249 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
6250 func (iin*IInput)GotoTOPW(){
6251     str := iin.line
6252     i := len(str)
6253     if i <= 0 {
6254         return
6255     }
6256     //i0 := i
6257     i -= 1
6258     lastSize := 0
6259     var lastRune rune
6260     var found = -1
6261     for 0 < i { // skip preamble spaces
6262         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6263         if !isAlnum(lastRune) { // character, type, or string to be searched
6264             i -= lastSize
6265             continue
6266         }
6267         break
6268     }
6269     for 0 < i {
6270         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6271         if lastSize <= 0 { continue } // not the character top
6272         if !isAlnum(lastRune) { // character, type, or string to be searched
6273             found = i
6274             break
6275         }
6276         i -= lastSize
6277     }
6278     if found < 0 && i == 0 {
6279         found = 0
6280     }
6281     if 0 <= found {
6282         if !isAlnum(lastRune) { // or non-kana character
6283             //when positioning to the top o the word
6284             i += lastSize
6285         }
6286         iin.right = str[i:] + iin.right
6287         if 0 < i {
6288             iin.line = str[0:i]
6289         }else{
6290             iin.line = ""
6291         }
6292     }
6293     //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n", i0, i, found, iin.line, iin.right)
6294     //fmt.Printf("") // set debug messae at the end of line
6295 }
6296 // 0.2.8 2020-0901 created
6297 func (iin*IInput)GotoENDW(){
6298     str := iin.right
6299     if len(str) <= 0 {
6300         return
6301     }
6302     lastSize := 0
6303     var lastRune rune
6304     var lastW = 0
6305     i := 0
6306     inWord := false
6307 }
6308 lastRune, lastSize = utf8.DecodeRuneInString(str[0:])
6309 if !isAlnum(lastRune) {
6310     r, z := utf8.DecodeRuneInString(str[lastSize:])
6311     if 0 < z && !isAlnum(r) {
6312         inWord = true
6313     }
6314 }
6315 for i < len(str) {
6316     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6317     if lastSize <= 0 { break } // broken data?
6318     if !isAlnum(lastRune) { // character, type, or string to be searched
6319         break
6320     }
6321     lastW = i // the last alnum if in alnum word
6322     i += lastSize
6323 }
6324 if inWord {
6325     goto DISP
6326 }
6327 for i < len(str) {
6328     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6329     if lastSize <= 0 { break } // broken data?
6330     if !isAlnum(lastRune) { // character, type, or string to be searched
6331         break
6332     }
6333     i += lastSize
6334 }
6335 for i < len(str) {
6336     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6337     if lastSize <= 0 { break } // broken data?
6338     if !isAlnum(lastRune) { // character, type, or string to be searched
6339         break
6340     }
6341     lastW = i
6342     i += lastSize
6343 }
6344 DISP:
6345 if 0 < lastW {
6346     iin.line = iin.line + str[0:lastW]
6347     iin.right = str[lastW:]
6348 }
6349 //fmt.Printf("\n(%d)[%s][%s]\n", i, iin.line, iin.right)
6350 //fmt.Printf("") // set debug messae at the end of line
6351 }
6352 // 0.2.8 2020-0901 created
6353 func (iin*IInput)GotoNEXTW(){
6354     str := iin.right
6355     if len(str) <= 0 {
6356         return
6357     }
6358     lastSize := 0
6359     var lastRune rune
6360     var found = -1
6361     i := 1
6362     for i < len(str) {
6363         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
6364         if lastSize <= 0 { break } // broken data?
6365         if !isAlnum(lastRune) { // character, type, or string to be searched
6366             found = i
6367             break
6368         }
6369         i += lastSize
6370     }
6371     if 0 < found {

```

```

6372     if isAlnum(lastRune) { // or non-kana character
6373     }else { // when positioning to the top o the word
6374         found += lastSize
6375     }
6376     iin.line = iin.line + str[0:found]
6377     if 0 < found {
6378         iin.right = str[found:]
6379     }else{
6380         iin.right = ""
6381     }
6382 }
6383 //fmt.Printf("\n%d)[%s][%s]\n",i,iin.line,iin.right)
6384 //fmt.Printf("") // set debug messae at the end of line
6385 }
6386 // 0.2.8 2020-0902 created
6387 func (iin*Input)GotoPAIRCH(){
6388     str := iin.right
6389     if len(str) <= 0 {
6390         return
6391     }
6392     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
6393     if lastSize <= 0 {
6394         return
6395     }
6396     forw := false
6397     back := false
6398     pair := ""
6399     switch string(lastRune){
6400     case "(": pair = ")"; forw = true
6401     case ")": pair = "("; back = true
6402     case "[": pair = "]"; forw = true
6403     case "]": pair = "["; back = true
6404     case "{": pair = "}"; forw = true
6405     case "}": pair = "{"; back = true
6406     case "<": pair = ">"; forw = true
6407     case ">": pair = "<"; back = true
6408     case "\'": pair = "\'"; // context depednet, can be f" or back-double quote
6409     case "\"": pair = "\""; // context depednet, can be f" or back-quote
6410     // case Japanese Kakkos
6411     }
6412     if forw {
6413         iin.SearchForward(pair)
6414     }
6415     if back {
6416         iin.SearchBackward(pair)
6417     }
6418 }
6419 // 0.2.8 2020-0902 created
6420 func (iin*Input)SearchForward(pat string)(bool){
6421     right := iin.right
6422     found := -1
6423     i := 0
6424     if strBegins(right,pat) {
6425         ,z := utf8.DecodeRuneInString(right[i:])
6426         if 0 < z {
6427             i += z
6428         }
6429     }
6430     for i < len(right) {
6431         if strBegins(right[i:],pat) {
6432             found = i
6433             break
6434         }
6435         ,z := utf8.DecodeRuneInString(right[i:])
6436         if z <= 0 { break }
6437         i += z
6438     }
6439     if 0 <= found {
6440         iin.line = iin.line + right[0:found]
6441         iin.right = iin.right[found:]
6442         return true
6443     }else{
6444         return false
6445     }
6446 }
6447 // 0.2.8 2020-0902 created
6448 func (iin*Input)SearchBackward(pat string)(bool){
6449     line := iin.line
6450     found := -1
6451     i := len(line)-1
6452     for i := 1; 0 <= i; i-- {
6453         ,z := utf8.DecodeRuneInString(line[i:])
6454         if z <= 0 {
6455             continue
6456         }
6457         //fprintf(stderr,"-- %v\n",pat,line[i:])
6458         if strBegins(line[i:],pat) {
6459             found = i
6460             break
6461         }
6462     }
6463     //fprintf(stderr,"--%d\n",found)
6464     if 0 <= found {
6465         iin.right = line[found:] + iin.right
6466         iin.line = line[0:found]
6467         return true
6468     }else{
6469         return false
6470     }
6471 }
6472 // 0.2.8 2020-0902 created
6473 // search from top, end, or current position
6474 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
6475     if forw {
6476         for ,v := range gsh.CommandHistory {
6477             if 0 <= strings.Index(v.CmdLine,pat) {
6478                 //fprintf(stderr,"\n--De-- found %v [%v]\v\n",i,pat,v.CmdLine)
6479                 return true,v.CmdLine
6480             }
6481         }
6482     }else{
6483         hlen := len(gsh.CommandHistory)
6484         for i := hlen-1; 0 < i; i-- {
6485             v := gsh.CommandHistory[i]
6486             if 0 <= strings.Index(v.CmdLine,pat) {
6487                 //fprintf(stderr,"\n--De-- found %v [%v]\v\n",i,pat,v.CmdLine)
6488                 return true,v.CmdLine
6489             }
6490         }
6491     }
6492     //fprintf(stderr,"\n--De-- not-found(%v)\n",pat)
6493     return false,"(Not Found in History)"
6494 }
6495 // 0.2.8 2020-0902 created
6496 func (iin*Input)GotoFORWSTR(pat string,gsh*GshContext){
6497     found := false
6498     if 0 < len(iin.right) {
6499         found = iin.SearchForward(pat)
6500     }
6501     if !found {
6502         found,line := gsh.SearchHistory(pat,true)
6503         if found {
6504             iin.line = line
6505             iin.right = ""
6506         }
6507     }
6508 }
6509 func (iin*Input)GotoBACKSTR(pat string, gsh*GshContext){
6510     found := false
6511     if 0 < len(iin.line) {
6512         found = iin.SearchBackward(pat)
6513     }
6514     if !found {
6515         found,line := gsh.SearchHistory(pat,false)
6516         if found {
6517             iin.line = line
6518             iin.right = ""
6519         }
6520     }
6521 }
6522 func (iin*Input)getString(prompt string){ // should be editable
6523     iin.clearline();
6524     fprintf(stderr,"\r\v",prompt)
6525     str := ""
6526     for {
6527         ch := iin.Getc(10*1000*1000)
6528         if ch == '\n' || ch == '\r' {
6529             break
6530         }
6531         sch := string(ch)
6532         str += sch
6533         fprintf(stderr,"%s",sch)
6534     }
6535     return str
6536 }
6537 // search pattern must be an array and selectable with `N`P
6538 var SearchPat = ""
6539 var SearchForw = true
6540 var SearchForw = true
6541 func (iin*Input)xgetline(prevline string, gsh*GshContext)(string){
6542     var ch int;
6543     MODE_ShowMode = false
6544     MODE_VicMode = false
6545     iin.Redraw();
6546     first := true

```

```

6549
6550 for cix := 0; ; cix++ {
6551     iin.pinMode = iin.inMode
6552     iin.inMode = false
6553
6554     ch = iin.Getc(1000*1000)
6555
6556     if ch != EV_TIMEOUT && first {
6557         first = false
6558         mode := 0
6559         if romkanmode {
6560             mode = 1
6561         }
6562         now := time.Now()
6563         Events = append(Events, Event(now, EV_MODE, int64(mode), CmdIndex))
6564     }
6565     if ch == 033 {
6566         MODE_ShowMode = true
6567         MODE_VicMode = IMODE_VicMode
6568         iin.Redraw();
6569         continue
6570     }
6571     if MODE_VicMode {
6572         switch ch {
6573             case '0': ch = GO_TOPL
6574             case 'S': ch = GO_ENDL
6575             case 'b': ch = GO_TOPW
6576             case 'c': ch = GO_ENDW
6577             case 'w': ch = GO_NEXTW
6578             case '8': ch = GO_PAIRCH
6579
6580             case 'j': ch = GO_DOWN
6581             case 'k': ch = GO_UP
6582             case 'h': ch = GO_LEFT
6583             case 'l': ch = GO_RIGHT
6584             case 'x': ch = DEL_RIGHT
6585             case 'a': MODE_VicMode = IMODE_VicMode
6586             ch = GO_RIGHT
6587             case 'i': MODE_VicMode = IMODE_VicMode
6588                 iin.Redraw();
6589                 continue
6590             case '-':
6591                 right, head := delHeadChar(iin.right)
6592                 if len([]byte(head)) == 1 {
6593                     ch = int(head[0])
6594                     if 'a' <= ch && ch <= 'z' {
6595                         ch = ch + 'A'-'a'
6596                     } else
6597                     if 'A' <= ch && ch <= 'Z' {
6598                         ch = ch + 'a'-'A'
6599                     }
6600                     iin.right = string(ch) + right
6601                 }
6602                 iin.Redraw();
6603                 continue
6604             case 'f': // GO_FORWCH
6605                 iin.Redraw();
6606                 ch = iin.Getc(3*1000*1000)
6607                 if ch == EV_TIMEOUT {
6608                     iin.Redraw();
6609                     continue
6610                 }
6611                 SearchPat = string(ch)
6612                 SearchForw = true
6613                 iin.GotoFORWSTR(SearchPat, gsh)
6614                 iin.Redraw();
6615                 continue
6616             case '/':
6617                 SearchPat = iin.getstring("/") // should be editable
6618                 SearchForw = true
6619                 iin.GotoFORWSTR(SearchPat, gsh)
6620                 iin.Redraw();
6621                 continue
6622             case '?':
6623                 SearchPat = iin.getstring("?") // should be editable
6624                 SearchForw = false
6625                 iin.GotoBACKSTR(SearchPat, gsh)
6626                 iin.Redraw();
6627                 continue
6628             case 'n':
6629                 if SearchForw {
6630                     iin.GotoFORWSTR(SearchPat, gsh)
6631                 } else {
6632                     iin.GotoBACKSTR(SearchPat, gsh)
6633                 }
6634                 iin.Redraw();
6635                 continue
6636             case 'N':
6637                 if !SearchForw {
6638                     iin.GotoFORWSTR(SearchPat, gsh)
6639                 } else {
6640                     iin.GotoBACKSTR(SearchPat, gsh)
6641                 }
6642                 iin.Redraw();
6643                 continue
6644         }
6645     }
6646     switch ch {
6647         case GO_TOPW:
6648             iin.GotoTOPW()
6649             iin.Redraw();
6650             continue
6651         case GO_ENDW:
6652             iin.GotoENDW()
6653             iin.Redraw();
6654             continue
6655         case GO_NEXTW:
6656             // To next space then
6657             iin.GotoNEXTW()
6658             iin.Redraw();
6659             continue
6660         case GO_PAIRCH:
6661             iin.GotoPAIRCH()
6662             iin.Redraw();
6663             continue
6664     }
6665
6666     //fprintf(stderr, "A[%02X]\n", ch);
6667     if (ch == '\\') || ch == 033 {
6668         MODE_ShowMode = true
6669         metach := ch
6670         iin.waitingMeta = string(ch)
6671         iin.Redraw();
6672         // set cursor //fprintf(stderr, "???\b\b\b")
6673         ch = fgetcTimeout(stdin, 2000*1000)
6674         // reset cursor
6675         iin.waitingMeta = ""
6676
6677         cmdch := ch
6678         if (ch == EV_TIMEOUT) {
6679             if metach == 033 {
6680                 continue
6681             }
6682             ch = metach
6683         } else
6684         /*
6685         if (ch == 'm' || ch == 'M') {
6686             mch := fgetcTimeout(stdin, 1000*1000)
6687             if mch == '*' {
6688                 romkanmode = true
6689             } else {
6690                 romkanmode = false
6691             }
6692             continue
6693         } else
6694         /*
6695         if (ch == 'k' || ch == 'K') {
6696             MODE_Recursive = IMODE_Recursive
6697             iin.Translate(cmdch);
6698             continue
6699         } else
6700         if (ch == 'j' || ch == 'J') {
6701             iin.Translate(cmdch);
6702             continue
6703         } else
6704         if (ch == 'i' || ch == 'I') {
6705             iin.Replace(cmdch);
6706             continue
6707         } else
6708         if (ch == 'l' || ch == 'L') {
6709             MODE_LowerLock = IMODE_LowerLock
6710             MODE_CapsLock = false
6711             if MODE_ViTrace {
6712                 fprintf(stderr, "%v\r\n", string(cmdch));
6713             }
6714             iin.Redraw();
6715             continue
6716         } else
6717         if (ch == 'u' || ch == 'U') {
6718             MODE_CapsLock = IMODE_CapsLock
6719             MODE_LowerLock = false
6720             if MODE_ViTrace {
6721                 fprintf(stderr, "%v\r\n", string(cmdch));
6722             }
6723             iin.Redraw();
6724             continue
6725         } else

```

```

6726     if( ch == 'v' || ch == 'V' ){
6727         MODE_ViTrace = !MODE_ViTrace
6728         if MODE_ViTrace {
6729             fprintf(stderr, "%v\n", string(cmdch));
6730         }
6731         iin.Redraw();
6732         continue;
6733     }else
6734     if( ch == 'c' || ch == 'C' ){
6735         if 0 < len(iin.line) {
6736             xline,tail := delTailChar(iin.line)
6737             if len([]byte(tail)) == 1 {
6738                 ch = int(tail[0])
6739                 if 'a' <= ch && ch <= 'z' ){
6740                     ch = ch + 'A'-'a'
6741                 }else
6742                 if( 'A' <= ch && ch <= 'Z' ){
6743                     ch = ch + 'a'-'A'
6744                 }
6745                 iin.line = xline + string(ch)
6746             }
6747         }
6748         if MODE_ViTrace {
6749             fprintf(stderr, "%v\n", string(cmdch));
6750         }
6751         iin.Redraw();
6752         continue;
6753     }else{
6754         iin.pch = append(iin.pch,ch) // push
6755         ch = '\\'
6756     }
6757 }
6758 switch( ch ){
6759 case 'P'-0x40: ch = GO_UP
6760 case 'N'-0x40: ch = GO_DOWN
6761 case 'B'-0x40: ch = GO_LEFT
6762 case 'F'-0x40: ch = GO_RIGHT
6763 }
6764 //fprintf(stderr, "B[%02X]\n",ch);
6765 switch( ch ){
6766 case 0:
6767     continue;
6768
6769 case '\t':
6770     iin.Replace('j');
6771     continue;
6772 case 'X'-0x40:
6773     iin.Replace('j');
6774     continue;
6775
6776 case EV_TIMEOUT:
6777     iin.Redraw();
6778     if iin.pinmode {
6779         fprintf(stderr, "\\J\n")
6780         iin.inmode = true
6781     }
6782     continue;
6783 case GO_UP:
6784     if iin.lno == 1 {
6785         continue;
6786     }
6787     cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
6788     if ok {
6789         iin.line = cmd
6790         iin.right = ""
6791         iin.lno = iin.lno - 1
6792     }
6793     iin.Redraw();
6794     continue;
6795 case GO_DOWN:
6796     cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
6797     if ok {
6798         iin.line = cmd
6799         iin.right = ""
6800         iin.lno = iin.lno + 1
6801     }else{
6802         iin.line = ""
6803         iin.right = ""
6804         if iin.lno == iin.lastlno-1 {
6805             iin.lno = iin.lno + 1
6806         }
6807     }
6808     iin.Redraw();
6809     continue;
6810 case GO_LEFT:
6811     if 0 < len(iin.line) {
6812         xline,tail := delTailChar(iin.line)
6813         iin.line = xline
6814         iin.right = tail + iin.right
6815     }
6816     iin.Redraw();
6817     continue;
6818 case GO_RIGHT:
6819     if( 0 < len(iin.right) && iin.right[0] != 0 ){
6820         xright,head := delHeadChar(iin.right)
6821         iin.right = xright
6822         iin.line += head
6823     }
6824     iin.Redraw();
6825     continue;
6826 case EOF:
6827     goto EXIT;
6828 case 'R'-0x40: // replace
6829     dst := convs(iin.line+iin.right);
6830     iin.line = dst
6831     iin.right = ""
6832     iin.Redraw();
6833     continue;
6834 case 'T'-0x40: // just show the result
6835     readDi();
6836     romkanmode = !romkanmode;
6837     iin.Redraw();
6838     continue;
6839 case 'L'-0x40:
6840     iin.Redraw();
6841     continue;
6842 case 'K'-0x40:
6843     iin.right = ""
6844     iin.Redraw();
6845     continue;
6846 case 'E'-0x40:
6847     iin.line += iin.right
6848     iin.right = ""
6849     iin.Redraw();
6850     continue;
6851 case 'A'-0x40:
6852     iin.right = iin.line + iin.right
6853     iin.line = ""
6854     iin.Redraw();
6855     continue;
6856 case 'U'-0x40:
6857     iin.line = ""
6858     iin.right = ""
6859     iin.clearline();
6860     iin.Redraw();
6861     continue;
6862 case DEL_RIGHT:
6863     if( 0 < len(iin.right) ){
6864         iin.right_ = delHeadChar(iin.right)
6865         iin.Redraw();
6866     }
6867     continue;
6868 case 0x7F: // BS? not DEL
6869     if( 0 < len(iin.line) ){
6870         iin.line_ = delTailChar(iin.line)
6871         iin.Redraw();
6872     }
6873     /*
6874     else
6875     if( 0 < len(iin.right) ){
6876         iin.right_ = delHeadChar(iin.right)
6877         iin.Redraw();
6878     }
6879     */
6880     continue;
6881 case 'H'-0x40:
6882     if( 0 < len(iin.line) ){
6883         iin.line_ = delTailChar(iin.line)
6884         iin.Redraw();
6885     }
6886     continue;
6887 }
6888 if( OnWindows && ch == '\n' ){
6889     continue;
6890 }
6891 if( ch == '\n' || ch == '\r' ){
6892     iin.line += iin.right;
6893     iin.right = ""
6894     iin.Redraw();
6895     //putc(ch,stderr);
6896     fprintf(stderr, "\n"); // NL on Unix, CR on Windows
6897     AtConsoleLineTop = true
6898     break;
6899 }
6900 if MODE_CapsLock {
6901     if 'a' <= ch && ch <= 'z' {
6902         ch = ch+'A'-'a'

```

```

6903     }
6904     }
6905     if MODE_LowerLock {
6906         if 'A' <= ch && ch <= 'Z' {
6907             ch = ch+'a'-'A'
6908         }
6909     }
6910     iin.line += string(ch);
6911     iin.Redraw();
6912 }
6913 EXIT:
6914     return iin.line + iin.right;
6915 }
6916
6917 func getline_main(){
6918     line := xgetline(0,"",nil)
6919     fprintf(stderr,"%s\n",line);
6920 /*
6921     dp = strpbk(line,"\r\n");
6922     if( dp != NULL ){
6923         *dp = 0;
6924     }
6925
6926     if( 0 ){
6927         fprintf(stderr,"\n(%d)\n",int(strlen(line)));
6928     }
6929     if( lseek(3,0,0) == 0 ){
6930         if( romkammode ){
6931             var buf [8*1024]byte;
6932             convs(line,buf);
6933             strcpy(line,buf);
6934         }
6935         write(3,line,strlen(line));
6936         ftruncate(3,lseek(3,0,SEEK_CUR));
6937         //fprintf(stderr,"outsize=%d\n",int)lseek(3,0,SEEK_END);
6938         lseek(3,0,SEEK_SET);
6939         close(3);
6940     }else{
6941         fprintf(stderr,"\r\n gotline: ");
6942         trans(line);
6943         //printf("%s\n",line);
6944         printf("\n");
6945     }
6946 */
6947 }
6948 //== end ===== getline
6949
6950 //
6951 // $USERHOME/.gsh/
6952 // gsh-rc.txt, or gsh-configure.txt
6953 // gsh-history.txt
6954 // gsh-aliases.txt // should be conditional?
6955 //
6956 func (gshCtx *GshContext)gshSetupHomedir()(bool) {
6957     homedir,found := userHomeDir()
6958     if !found {
6959         fmt.Printf("--E-- You have no UserHomeDir\n")
6960         return true
6961     }
6962     gshhome := homedir + "/" + GSH_HOME
6963     err2 := os.Stat(gshhome)
6964     if err2 != nil {
6965         err3 := os.Mkdir(gshhome,0700)
6966         if err3 != nil {
6967             fmt.Printf("--E-- Could not Create %s (%s)\n",
6968                 gshhome,err3)
6969             return true
6970         }
6971         fmt.Printf("--I-- Created %s\n",gshhome)
6972     }
6973     gshCtx.GshHomeDir = gshhome
6974     return false
6975 }
6976
6977 func setupGshContext()(GshContext,bool){
6978     //gshPA := syscall.ProcAttr {
6979     gshPA := os.ProcAttr {
6980         // the starting directory
6981         os.Environ(), // environ[]
6982         //[]uintptr(os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()),
6983         []os.File{os.Stdin,os.Stdout,os.Stderr},
6984         nil, // OS specific
6985     }
6986     cwd, _ := os.Getwd()
6987     gshCtx := GshContext {
6988         cwd, // StartDir
6989         // GetLine
6990         []GchdirHistory { {cwd,time.Now(),0} }, // ChdirHistory
6991         gshPA,
6992         []GCommandHistory{}, //something for invokation?
6993         GCommandHistory{}, // CmdCurrent
6994         false,
6995         []os.ProcessState{}, //[]int{},
6996         rUsage{},
6997         "", // GshHomeDir
6998         Ttyid(),
6999         false,
7000         false,
7001         []PluginInfo{},
7002         []string{},
7003         "v",
7004         ValueStack{},
7005         GServer{"",""}, // LastServer
7006         "", // RSEKV
7007         cwd, // RWD
7008         CheckSum{},
7009     }
7010     err := gshCtx.gshSetupHomedir()
7011     return gshCtx, err
7012 }
7013
7014 func (gsh *GshContext)gshellh(gline string)(bool){
7015     ghist := gsh.CmdCurrent
7016     ghist.WorKDir,_ = os.Getwd()
7017     ghist.WorKDir = len(ghist.ChdirHistory)-1
7018     //fmt.Printf("--D--ChdirHistory(%d)\n",len(ghist.ChdirHistory))
7019     ghist.StartAt = time.Now()
7020     rusagev1 := Getrusagev()
7021     gsh.CmdCurrent.FoundFile = []string{}
7022     fin := gsh.tgshellh(gline)
7023     rusagev2 := Getrusagev()
7024     ghist.Rusagev = RusageSubv(rusagev2,rusagev1)
7025     ghist.EndAt = time.Now()
7026     ghist.CmdLine = gline
7027     ghist.FoundFile = gsh.CmdCurrent.FoundFile
7028     /* record it but not show in list by default
7029     if len(gline) == 0 {
7030         continue
7031     }
7032     if gline == "hi" || gline == "history" { // don't record it
7033         continue
7034     }
7035     */
7036     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
7037     return fin
7038 }
7039 // <a name="main">Main loop</a>
7040 func script(gshCtxGiven *GshContext) (_ GshContext) {
7041     gshCtxBuf,err0 := setupGshContext()
7042     if err0 {
7043         return gshCtxBuf;
7044     }
7045     gshCtx := &gshCtxBuf
7046     //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
7047     //resmap()
7048     //
7049     /*
7050     if false {
7051         gsh_getlinev, with_exgetline :=
7052             which("PATH",[]string{"which","gsh-getline","-s"})
7053         if with_exgetline {
7054             gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
7055             gshCtx.GetLine = toFullpath(gsh_getlinev[0])
7056         }else{
7057             fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
7058         }
7059     }
7060     */
7061
7062     ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
7063     gshCtx.CommandHistory = append(gshCtx.CommandHistory,ghist0)
7064
7065     prevline := ""
7066     skipping := false
7067     for hix := len(gshCtx.CommandHistory); {
7068         gline := gshCtx.getline(hix,skipping,prevline)
7069         if skipping {
7070             if strings.Index(gline,"fi") == 0 {
7071                 fmt.Printf("fi\n");
7072                 skipping = false;
7073             }else{
7074                 //fmt.Printf("%s\n",gline);
7075             }
7076             continue
7077         }
7078         if strings.Index(gline,"if") == 0 {

```





```

7257 "77yX77yScm5hbmFgdXVuaVgJ77yX77yScuS4g+WngeS6jHgJnzIKa29idW5uCeWai+WlHgp0" +
7258 "aWthcmFxcE0BoeOB1+OC1Op0aWthcmEJ5YqbCmNoaWthcmEJ5YqbcjwvdGV4dGfyZWE+cg=" +
7259 //</span>
7260
7261 var JA_JKLDic = //<span id="gsh-ja-jkl-dic">
7262 "data:text/dic;base64," +
7263 "LyY2XkxkI1S1F1amRyZ2p3JzZ2p3K0wPKS0womjAYMGoW0DE5KSheLV4pL1NhdG94SVRT"+
7264 "CmtqamRbGtqa2tsa2s1OS4lueVjAqamtqamwJ44GCCmtqbnjgY0Ka2tqbAnjgYKamtq"+
7265 "amwJ44G1Cmtqa2trbAnjgY0Ka2pra2wJ44GLCmpra2rbAnjgY0Ka2tramuJ44GFCmpra2s"+
7266 "Ce0B0pampqpbAnjgZKamtqa2psCe0B1Opamtqa2wJ44GKCmpqamqbnjgJkKa2pamts"+
7267 "Ce0BmwpqamrbanjgZ0KamtsCe0Bnwpr2prbanjgaEka2pqa2wJ44GKmtqa2pqbAnjgaYK"+
7268 "a2tqa2tsCe0BqApramtsCe0Bqppqa2prbanjgask2tra2wJ44G6Cmpqa2psCe0B0pra2pqa"+
7269 "bnjgaKamtra2wJ44GVCmpqa2tqbAnjgblKampra2wJ44G1CmtaCe0B0pqa2tsCe0Buwppq"+
7270 "a2tqbAnjgKa2tra2pqbAnjgKaKamtra2psCe0C0pamtqa2pqa2wJ44K0CmtqamwJ44"+
7271 "44KECmpra2pqbAnjgoKampsCe0C1Apra2tsCe0C1OpqamtsCe0C1pqa2pqa2wJ44KLCmpq"+
7272 "amwJ44KMCmtqa2psCe0Cj0pqa2psCe0Cjwpramtra2wJ44K0CmtqamrbanjgEka2pqa2wJ44"+
7273 "44KSCmtqa2prbanjgMka2pqa2psCe0Bvpra2wJ44KbCmtrampbrbanjgWka2pramtqbAnj"+
7274 "qIEK";
7275 //</span>
7276
7277 //</span>
7278 /*
7279 <style id="gsh-references-style">
7280 #references details { font-family:Georgia; }
7281 #references a { font-family:Georgia; }
7282 .wrap { white-space:normal; }
7283 </style>
7284 <details id="references"><summary>References</summary><div class="gsh-src">
7285 Web technology
7286 <a href="https://html.spec.whatwg.org">HTML: The Living Standard</a> (September 2020)
7287 <a href="https://html.spec.whatwg.org/dev">Developer Version</a>
7288
7289 <a href="https://drafts.csswg.org">CSS Working Group Editor Drafts</a> (September 2020)
7290
7291 <a href="https://developer.mozilla.org/ja/docs/Web">MDN web docs</a>
7292 <a href="https://developer.mozilla.org/ja/docs/Web/HTML/Element">HTML</a>
7293 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS">CSS</a> : <span class="wrap">
7294 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS>Selectors">selectors</a>
7295 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/background-repeat">repeat</a></span>
7296 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
7297 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
7298 <a href="https://mdn-web-dns.a3-us-west-2.amazonaws.com/MDN-Browser-Compatibility-Report-2020.pdf">MDN Browser Compatibility Report (2020)</a>
7299
7300 Go language (August 2020 / Go 1.15)
7301 <a href="https://golang.org">The Go Programming Language</a>
7302 <a href="https://golang.org/pkg/>Packages</a>
7303 <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>
7304
7305 <a href="https://stackoverflow.com/>Stackoverflow</a>
7306 </--
7307 <iframe src="https://golang.org" width="100%" height="300"></iframe>
7308 -->
7309 </div></details>
7310 */
7311 /*
7312 <details id="html-src" onclick="frame_open();"><summary>Raw Sources</summary><div>
7313
7314 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
7315 <details id="gsh-whole-view"><summary>Whole file</summary>
7316 <a name="whole-src-view"></a>
7317 <span id="src-frame"></span><!-- a window to show source code -->
7318 </details>
7319
7320 <details id="gsh-style-frame" onclick="fill_CSSView();"><summary>CSS part</summary>
7321 <a name="style-src-view"></a>
7322 <span id="gsh-style-view"></span>
7323 </details>
7324
7325 <details id="gsh-script-frame" onclick="fill_JavaScriptView();"><summary>JavaScript part</summary>
7326 <a name="script-src-view"></a>
7327 <span id="gsh-script-view"></span>
7328 </details>
7329
7330 <details id="gsh-data-frame" onclick="fill_DataView();"><summary>Builtin data part</summary>
7331 <a name="gsh-data-frame"></a>
7332 <span id="gsh-data-view"></span>
7333 </details>
7334
7335 </div></details>
7336 */
7337
7338 /*
7339 <div id="GshFooter0"></div>
7340 <!-- 2020-09-17 SatoxTTS, visible script { -->
7341 <details id="gjs"><summary>GScript</summary>
7342 <style>gjscript { font-family:Georgia; }</style>
7343 <pre id="gjscript_1" class="gjscript"> function gjtestl(){ alert('Hello GJScript!'); }
7344 gjtestl()
7345 </pre>
7346 </script>
7347 gjs = document.getElementById('gjscript_1');
7348 //eval(gjs.inlineHTML);
7349 //gjs.outerHTML = " ";
7350 </script>
7351 </details><!-- END-OF-VISIBLE-PART ----- } -->
7352
7353 <!--
7354 // 2020-0906 added,
7355 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
7356 https://developer.mozilla.org/en-US/docs/Web/CSS/position
7357 -->
7358 <span id="GshGrid">(^_^)</small>(Hit j k l h)</small></span>
7359
7360 <span id="GStat"><br>
7361 </span>
7362 <span id="CMenu" onclick="GShellMenu(this)" draggable="true"></span>
7363 <span id="GTop"></span>
7364 <div id="GShellPlane" onclick="showGShellPlane();"></div>
7365 <div id="RawTextViewer"></div>
7366 <div id="RawTextViewerClose" onclick="hideRawTextViewer();"> CLOSE </div>
7367
7368 <style id="GshStylebef">
7369 #LineNumbered table, tr, td {
7370   margin: 0;
7371   padding: 4px;
7372   spacing: 0;
7373   border: 12px;
7374 }
7375 textarea.LineNumber {
7376   font-size: 12px;
7377   font-family: monospace, Courier New;
7378   color: #282;
7379   padding: 4px;
7380   text-align: right;
7381 }
7382 textarea.LineNumbered {
7383   font-size: 12px;
7384   font-family: monospace, Courier New;
7385   padding: 4px;
7386   wrap: off;
7387 }
7388 #RawTextViewer {
7389   z-index: 0;
7390   position: fixed; top: 0px; left: 0px;
7391   width: 100%; xxxheight: 50px; xheight: 0px;
7392   overflow: auto;
7393   color: #fff; background-color: rgba(128, 128, 256, 0.2);
7394   font-size: 12px;
7395   spellcheck: false;
7396 }
7397 #RawTextViewerClose {
7398   z-index: 0;
7399   position: fixed; top: -100px; left: -100px;
7400   color: #fff; background-color: rgba(128, 128, 256, 0.2);
7401   font-size: 20px; font-family: Georgia;
7402   white-space: pre;
7403 }
7404 #xxxGShellPlane {
7405   z-index: 0;
7406   position: fixed; top: 0px; left: 0px;
7407   width: 100%; height: 50px;
7408   overflow: auto;
7409   color: #fff; background-color: rgba(128, 128, 256, 0.3);
7410   font-size: 12px;
7411 }
7412 #xxxGTop {
7413   z-index: 9;
7414   opacity: 1.0;
7415   position: fixed; top: 0px; left: 0px;
7416   width: 320px; height: 20px;
7417   color: #fff; background-color: rgba(32, 32, 160, 0.15);
7418   color: #fff; font-size: 12px;
7419 }
7420 #xxxGPos {
7421   z-index: 12;
7422   position: fixed; top: 0px; left: 0px;
7423   opacity: 1.0;
7424   width: 640px; height: 30px;
7425   color: #fff; background-color: rgba(0, 0, 0, 0.2);
7426   color: #fff; font-size: 12px;
7427 }
7428 #GMenu {
7429   z-index: 100000000;
7430   position: fixed; top: 250px; left: 0px;
7431   opacity: 1.0;
7432   width: 100px; height: 100px;
7433   color: #fff;

```

```

7434 color:#fff; background-color:rgba(0,0,0,0.0);
7435 color:#fff; font-size:16px; font-family:Georgia;
7436 background-repeat:no-repeat;
7437
7438 #xxxGStat{
7439 z-index:8;
7440 xopac:0.0;
7441 position:fixed; top:20px; left:0px;
7442 xwidth:640px;
7443 width:640px; height:90px;
7444 color:#fff; background-color:rgba(0,0,128,0.04);
7445 color:#fff; font-family:Georgia;
7446 }
7447 #GLog{
7448 z-index:10;
7449 position:fixed; top:50px; left:0px;
7450 opacity:1.0;
7451 width:640px; height:60px;
7452 color:#fff; background-color:rgba(0,0,128,0.10);
7453 font-size:12px;
7454 }
7455 #GshGrid {
7456 z-index:11;
7457 xopac:0.0;
7458 position:fixed; top:0px; left:0px;
7459 width:320px; height:30px;
7460 color:#9f9; font-size:16px;
7461 }
7462 xbody {display:none;
7463 #gsh-link{color:green;
7464 #gsh {border-width:1px;margin:0;padding:0;
7465 #gsh {font-family:monospace; cursor:New; color:#44d; font-size:8px;
7466 #gsh header{height:100px;
7467 #gsh header{height:100px;background-image:url(GShell-Logo00.png);
7468 #GshMenu{font-size:14pt;color:#44d;
7469 #GshMenu{font-size:14pt;color:#44d;
7470 #GshMenu{font-size:14pt;color:#2a2;padding:4px; text-align:right;
7471 }
7472 #GshMenu:hovert{font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
7473 }
7474 #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;
7475 #gsh note{color:#000;font-size:10pt;
7476 #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;
7477 #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;
7478 #gsh details{color:#888;background-color:#fff;font-family:monospace;
7479 #gsh summary{font-size:16pt;color:#fff;background-color:#8af;xxxheight:30px;
7480 #gsh summary{font-size:16pt;color:#fff;
7481 padding:4pt;
7482 line-height:1.0;
7483 vertical-align:middle;
7484 xxx-background-color:#faf;
7485 background-color:#6881AD;xxx-Blue;
7486 xxxheight:30px;
7487 }
7488 #gsh pre{font-size:11pt;color:#223;background-color:#ffff;
7489 #gsh a{color:#24a;
7490 #gsh a{name}{color:#24a;font-size:16pt;
7491 #gsh #gsh {font-size:11pt;font-family:monospace, Courier New;font-size:11pt;
7492 #gsh #gsh-src{background-color:#ffff;color:#223;
7493 #gsh-src-src{spellcheck:false
7494 #srcTextarea{white-space:pre;font-family:Courier New;font-size:10pt;
7495 #srcTextarea{background-color:#fff;color:#223;
7496 #gsh-code {white-space:pre;font-family:Courier New !important;
7497 #gsh-code {color:#024;font-size:11pt; background-color:#fafaf;
7498 #gsh-golang-da {display:none;
7499 #gsh-winid {color:#000;font-size:14pt;
7500 }
7501 #gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;
7502 #gsh-document {color:#000;background-color:#fff !important;
7503 #gsh-document > h2{color:#000;background-color:#fff !important;
7504 #gsh-document details{color:#000;background-color:#fff;font-family:Georgia;
7505 #gsh-document p{max-width:550pt;color:#000;background-color:#fff;font-family:Georgia;
7506 #gsh-document address{width:500pt;color:#000;background-color:#fff;font-family:Georgia;
7507 }
7508 #media print {
7509 #gsh pre{font-size:11pt !important;
7510 }
7511 }
7512 }
7513 }
7514 }
7515 }
7516 }
7517 }
7518 }
7519 }
7520 }
7521 }
7522 }
7523 }
7524 }
7525 }
7526 }
7527 }
7528 }
7529 }
7530 }
7531 }
7532 }
7533 }
7534 }
7535 }
7536 }
7537 }
7538 }
7539 }
7540 }
7541 }
7542 }
7543 }
7544 }
7545 }
7546 }
7547 }
7548 }
7549 }
7550 }
7551 }
7552 }
7553 }
7554 }
7555 }
7556 }
7557 }
7558 }
7559 }
7560 }
7561 }
7562 }
7563 }
7564 }
7565 }
7566 }
7567 }
7568 }
7569 }
7570 }
7571 }
7572 }
7573 }
7574 }
7575 }
7576 }
7577 }
7578 }
7579 }
7580 }
7581 }
7582 }
7583 }
7584 }
7585 }
7586 }
7587 }
7588 }
7589 }
7590 }
7591 }
7592 }
7593 }
7594 }
7595 }
7596 }
7597 }
7598 }
7599 }
7600 }
7601 }
7602 }
7603 }
7604 }
7605 }
7606 }
7607 }
7608 }
7609 }
7610 }

```





```

7965 z-index:10000;
7966 display:inline;
7967 position:relative;
7968 flex-wrap: wrap;
7969 top:0; left:0px;
7970 width:285px !important; height:205px !important;
7971 border:1px solid #eee; border-radius:2px;
7972 margin:0px; padding:0px;
7973 font-size:8pt;
7974 line-height:0.9;
7975 color:#fff; background-color:rgba(0,0,64,0.1) !important;
7976 }
7977 .GJTab{
7978 display:inline;
7979 position:relative;
7980 top:0px; left:0px;
7981 margin:0px; padding:2px;
7982 border:0px solid #000; border-radius:2px;
7983 width:90px; height:20px;
7984 font-family:Georgia;
7985 font-size:9pt;
7986 line-height:1.0;
7987 white-space:nowrap;
7988 color:#fff; background-color:rgba(0,0,64,0.7);
7989 text-align:center;
7990 vertical-align:middle;
7991 }
7992 .GJStat:focus{
7993 color:#df8 !important;
7994 background-color:rgba(32,32,32,1.0) !important;
7995 line-height:1.0;
7996 }
7997 .GJStat{
7998 display:inline;
7999 position:relative;
8000 top:0px; left:0px;
8001 margin:0px; padding:2px;
8002 border:0px solid #00f; border-radius:2px;
8003 width:160px; height:20px;
8004 font-family:monospace;
8005 font-size:9pt;
8006 line-height:1.0;
8007 color:#fff; background-color:rgba(0,0,64,0.2);
8008 text-align:center;
8009 vertical-align:middle;
8010 }
8011 .GJIcon{
8012 display:inline;
8013 position:relative;
8014 top:0px; left:1px;
8015 border:2px solid #44a;
8016 margin:0px; padding:1px;
8017 width:13.2; height:13.2px;
8018 border-radius:2px;
8019 font-family:Georgia;
8020 font-size:13.2px;
8021 line-height:1.0;
8022 white-space:nowrap;
8023 color:#fff; background-color:rgba(32,32,160,0.8);
8024 text-align:center;
8025 vertical-align:middle;
8026 text-shadow:0px 0px;
8027 }
8028 .GJText:focus{
8029 color:#fff !important;
8030 background-color:rgba(32,32,160,0.8) !important;
8031 line-height:1.0;
8032 }
8033 .GJText{
8034 display:inline;
8035 position:relative;
8036 top:0px; left:0px;
8037 border:0px solid #000; margin:0px; padding:0px;
8038 width:280px; height:160px;
8039 border:0px;
8040 font-family:Courier New,monospace !important;
8041 font-size:8pt;
8042 line-height:1.0;
8043 white-space:pre;
8044 color:#fff; xbackground-color:rgba(0,0,64,0.5);
8045 background-color:rgba(32,32,128,0.8) !important;
8046 }
8047 .GJMode{
8048 display:inline;
8049 position:relative;
8050 top:0px; left:0px;
8051 border:0px solid #000; border-radius:0px;
8052 margin:0px; padding:0px;
8053 width:280px; height:20px;
8054 font-size:9pt;
8055 line-height:1.0;
8056 white-space:nowrap;
8057 color:#fff; background-color:rgba(0,0,64,0.7);
8058 text-align:left;
8059 vertical-align:middle;
8060 }
8061 </style>
8062
8063 <script id="gsh-script">
8064 // 2020-0909 added, permanet local storage
8065 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
8066 var MyHistory =
8067 Permanent = localStorage;
8068 MyHistory = Permanent.getItem('MyHistory')
8069 if( MyHistory == null ){ MyHistory = "" }
8070 d = new Date()
8071 MyHistory = d.getTime()/1000+ " "+document.URL+"\n" + MyHistory
8072 Permanent.setItem('MyHistory',MyHistory)
8073 //Permanent.setItem('MyWindow',window)
8074
8075 var GJLog_Win = null
8076 var GJLog_Tab = null
8077 var GJLog_Stat = null
8078 var GJLog_Text = null
8079 var GJWin_Mode = null
8080 var FProductInterval = 0
8081
8082 var GJ_FactoryID = -1
8083 var GJFactory = null
8084 if( e = document.getElementById('GJFactory_0') ){
8085 GJFactory_1.height = 0
8086 GJFactory = e
8087 e.setAttribute('class','GJFactory')
8088 var GJ_FactoryID = 0
8089 }else{
8090 GJFactory = GJFactory_1
8091 var GJ_FactoryID = 1
8092 }
8093
8094 function GJFactory_Destroy(){
8095 gjf = GJFactory
8096 //gjf = document.getElementById('GJFactory')
8097 //alert('gjf='+gjf)
8098 if( gjf != null ){
8099 if( gjf.childNodes != null ){
8100 for( i = 0; i < gjf.childNodes.length; i++ ){
8101 gjf.removeChild(gjf.childNodes[i])
8102 }
8103 }
8104 gjf.innerHTML = ''
8105 gjf.style.width = 0
8106 gjf.style.height = 0
8107 gjf.removeAttribute('style')
8108 GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
8109 window.clearInterval(FProductInterval)
8110 return '-- Destroy; work product destroyed'
8111 }else{
8112 return '-- Destroy; work product not exist'
8113 }
8114 }
8115
8116 var TransMode = false
8117 var OnKeyControl = false
8118 var OnKeyShift = false
8119 var OnKeyAlt = false
8120 var OnKeyJ = false
8121 var OnKeyK = false
8122 var OnKeyL = false
8123
8124 function GJWin_OnKeyUp(ev){
8125 keycode = ev.code;
8126 if( keycode == 'ShiftLeft' ){
8127 OnKeyShift = false
8128 }else
8129 if( keycode == 'ControlLeft' ){
8130 onKeyControl = false
8131 }else
8132 if( keycode == 'AltLeft' ){
8133 OnKeyAlt = false
8134 }else
8135 if( keycode == 'KeyJ' ){ OnKeyJ = false }else
8136 if( keycode == 'KeyK' ){ OnKeyK = false }else
8137 if( keycode == 'KeyL' ){ OnKeyL = false }else
8138 {
8139 }
8140 ev.preventDefault()
8141 }

```

```

8142 function and(a,b){ if(a){ if(b){ return true; } return false; } }
8143 function GJWin_OnKeyDown(ev){
8144     keycode = ev.code;
8145     mode = '';
8146     key = '';
8147     if( keycode == 'ControlLeft' ){
8148         onKeyControl = true
8149         ev.preventDefault()
8150         return;
8151     }else
8152     if( keycode == 'ShiftLeft' ){
8153         OnKeyShift = true
8154         ev.preventDefault()
8155         return;
8156     }else
8157     if( keycode == 'AltLeft' ){
8158         ev.preventDefault()
8159         OnKeyAlt = true
8160         return;
8161     }else
8162     if( keycode == 'Backquote' ){
8163         TransMode = !TransMode
8164         ev.preventDefault()
8165     }else
8166     if( and(keycode == 'Space', OnKeyShift) ){
8167         TransMode = !TransMode
8168         ev.preventDefault()
8169     }else
8170     if( keycode == 'ShiftRight' ){
8171         TransMode = !TransMode
8172     }else
8173     if( keycode == 'Escape' ){
8174         TransMode = true
8175         ev.preventDefault()
8176     }else
8177     if( keycode == 'Enter' ){
8178         TransMode = false
8179         //ev.preventDefault()
8180     }
8181     if( keycode == 'KeyJ' ){ OnKeyJ = true }else
8182     if( keycode == 'KeyK' ){ OnKeyK = true }else
8183     if( keycode == 'KeyL' ){ OnKeyL = true }else
8184     {
8185     }
8186
8187     if( ev.altKey ){ key += 'Alt+' }
8188     if( onKeyControl ){ key += 'Ctrl+' }
8189     if( OnKeyShift ){ key += 'Shift+' }
8190     if( and(keycode == 'KeyJ', OnKeyJ) ){ key += 'J+' }
8191     if( and(keycode == 'KeyK', OnKeyK) ){ key += 'K+' }
8192     if( and(keycode == 'KeyL', OnKeyL) ){ key += 'L+' }
8193     key += keycode
8194
8195     if( TransMode ){
8196         //mode = "[343\201\202r]"
8197         JaUtf8 = new Uint8Array([0343,0201,0202]);
8198         utf8enc = new TextEncoder();
8199         JaA = utf8dec.decode(JaUtf8);
8200         mode = "[" + JaA + "r]";
8201     }
8202     }else{
8203         mode = "[---]"
8204     }
8205     ///// /gmode.innerHTML = "[---]"
8206     GJWin_Mode.innerHTML = mode + ' ' + key
8207     //alert('Key: '+keycode)
8208     ev.stopPropagation()
8209     //ev.preventDefault()
8210 }
8211 function GJWin_OnScroll(ev){
8212     x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
8213     y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
8214     GJLog_append('OnScroll: x='+x+',y='+y)
8215 }
8216 document.addEventListener('scroll',GJWin_OnScroll)
8217 function GJWin_OnResize(ev){
8218     w = window.innerWidth
8219     h = window.innerHeight
8220     GJLog_append('OnResize: w='+w+',h='+h)
8221 }
8222 window.addEventListener('resize',GJWin_OnResize)
8223
8224 var DragStartX = 0
8225 var DragStartY = 0
8226 function GJWin_DragStart(ev){
8227     // maybe this is the grabbing position
8228     this.style.position = 'fixed'
8229     x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
8230     y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
8231     GJLog_Stat.value = 'DragStart: x='+x+',y='+y
8232 }
8233 function GJWin_Drag(ev){
8234     x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
8235     this.style.left = x - DragStartX
8236     this.style.top = y - DragStartY
8237     this.style.zIndex = '30000'
8238     this.style.position = 'fixed'
8239     x = this.getBoundingClientRect().left.toFixed(0)
8240     y = this.getBoundingClientRect().top.toFixed(0)
8241     GJLog_Stat.value = "x="+x+",y="+y
8242     ev.preventDefault()
8243     ev.stopPropagation()
8244 }
8245 function GJWin_DragEnd(ev){
8246     x = ev.clientX; y = ev.clientY
8247     //x = ev.pageX; y = ev.pageY
8248     this.style.left = x - DragStartX
8249     this.style.top = y - DragStartY
8250     this.style.zIndex = '30000'
8251     this.style.position = 'fixed'
8252     if( true ){
8253         console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
8254             +' parent='+this.parentNode.id)
8255     }
8256     x = this.getBoundingClientRect().left.toFixed(0)
8257     y = this.getBoundingClientRect().top.toFixed(0)
8258     GJLog_Stat.value = "x="+x+",y="+y
8259     ev.preventDefault()
8260     ev.stopPropagation()
8261 }
8262 function GJWin_DragIgnore(ev){
8263     ev.preventDefault()
8264     ev.stopPropagation()
8265 }
8266 // 2020-09-15 let every object have console view!
8267 var GJ_ConsoleID = 0
8268 var PrevReport = new Date()
8269 function GJLog_StatUpdate(){
8270     txa = GJLog_Stat;
8271     if( txa == null ){
8272         return;
8273     }
8274     tmlap0 = new Date();
8275     p = txa.parentNode;
8276     pw = txa.getBoundingClientRect().width;
8277     ph = txa.getBoundingClientRect().height;
8278     //txa.value += '#'+p.id+' pw='+pw+' ph='+ph+'\n';
8279     txl = '#'+p.id+' pw='+pw+' ph='+ph+'\n';
8280
8281     w = txa.getBoundingClientRect().width;
8282     h = txa.getBoundingClientRect().height;
8283     //txa.value += 'w='+w+', h='+h+'\n';
8284     txl += 'w='+w+', h='+h+'\n';
8285
8286     //txa.value += '\n';
8287     //txa.value += DateShort() + '\n';
8288     txl += '\n';
8289     txl += DateShort() + '\n';
8290     tmlap1 = new Date();
8291
8292     txa.value += txl;
8293     tmlap2 = new Date();
8294
8295     // vertical centering of the last line
8296     sHeight = txa.scrollHeight - 30; // depends on the font-size
8297     tmlap3 = new Date();
8298
8299     txa.scrollTop = sHeight; // depends on the font-size
8300     tmlap4 = new Date();
8301
8302     now = tmlap0.getTime();
8303     if( PrevReport == 0 || 10000 <= now-PrevReport ){
8304         PrevReport = now;
8305         console.log('StatBarUpdate: '
8306             + 'len=' + txa.value.length + ' byte, '
8307             + 'time=' + (tmlap4 - tmlap0) + ' ms {'
8308             + 'tadd=' + (tmlap2 - tmlap1) + ', '
8309             + 'hcal=' + (tmlap3 - tmlap2) + ', '
8310             + 'scl=' + (tmlap4 - tmlap3) + '}'
8311             );
8312     }
8313 }
8314 GJWin_StatUpdate = GJLog_StatUpdate;
8315 function GJ_showTime(wid){
8316     //e = document.getElementById(wid);
8317     //console.log(wid.id+'.value.length='+wid.value.length)
8318     if( e != null ){

```

```

8319 //e.value = DateShort();
8320 }else{
8321 // should remove the Listener
8322 }
8323 }
8324 function GJWin_OnResizeTextarea(ev){
8325 this.value += 'resized: ' + '\n'
8326 }
8327 function GJ_NewConsole(wname){
8328 wid = wname + ' ' + GJ_ConsoleID
8329 GJ_ConsoleID += 1
8330
8331 GJFactory.style.setProperty('width',360+'px'); //GJFsize
8332 GJFactory.style.setProperty('height',320+'px')
8333 e = GJFactory;
8334 console.log('GJFa #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
8335
8336 if( GJFactory.innerHTML == "" ){
8337 GJFactory.innerHTML = '<+H3>GJ_Factory_' + GJ_FactoryID + '<+H3><+hr>\n'
8338 }else{
8339 GJFactory.innerHTML += '<+hr>\n'
8340 }
8341
8342 gjwin = GJLog_Win = document.createElement('span')
8343 gjwin.id = wid
8344 gjwin.setAttribute('class', 'GJWin')
8345 gjwin.setAttribute('draggable', 'true')
8346 gjwin.addEventListener('dragstart', GJWin_DragStart)
8347 gjwin.addEventListener('drag', GJWin_Drag)
8348 gjwin.addEventListener('dragend', GJWin_Drag)
8349 gjwin.addEventListener('dragover', GJWin_DragIgnore)
8350 gjwin.addEventListener('dragenter', GJWin_DragIgnore)
8351 gjwin.addEventListener('dragleave', GJWin_DragIgnore)
8352 gjwin.addEventListener('dragexit', GJWin_DragIgnore)
8353 gjwin.addEventListener('drop', GJWin_DragIgnore)
8354 gjwin.addEventListener('keydown', GJWin_OnKeyDown)
8355
8356 gjtab = GJLog_Tab = document.createElement('textarea')
8357 gjtab.addEventListener('keydown', GJWin_OnKeyDown)
8358 gjtab.style.readonly = true
8359 gjtab.contentEditable = false
8360 gjtab.value = wid
8361 gjtab.id = wid + ' Tab'
8362 gjtab.setAttribute('class', 'GJTab')
8363 gjtab.setAttribute('spellcheck', 'false')
8364 gjwin.appendChild(gjtab)
8365
8366 gjstat = GJLog_Stat = document.createElement('textarea')
8367 gjstat.addEventListener('keydown', GJWin_OnKeyDown)
8368 gjstat.id = wid + ' Stat'
8369 gjstat.value = DateShort()
8370 gjstat.setAttribute('class', 'GJStat')
8371 gjstat.setAttribute('spellcheck', 'false')
8372 gjwin.appendChild(gjstat)
8373
8374 gjicon = document.createElement('span')
8375 gjicon.addEventListener('keydown', GJWin_OnKeyDown)
8376 gjicon.id = wid + ' Icon'
8377 gjicon.innerHTML = '<font color=#f44>J</font>'
8378 gjicon.setAttribute('class', 'GJIcon')
8379 gjicon.setAttribute('spellcheck', 'false')
8380 gjwin.appendChild(gjicon)
8381
8382 gjtext = GJLog_Text = document.createElement('textarea')
8383 gjtext.addEventListener('keydown', GJWin_OnKeyDown)
8384 gjtext.addEventListener('keyup', GJWin_OnKeyUp)
8385 gjtext.addEventListener('resize', GJWin_OnResizeTextarea)
8386 gjtext.id = wid + ' Text'
8387 gjtext.setAttribute('class', 'GJText')
8388 gjtext.setAttribute('spellcheck', 'false')
8389 gjwin.appendChild(gjtext)
8390
8391 // user's mode as of IME
8392 gjmode = GJWin_Mode = document.createElement('textarea')
8393 gjmode.addEventListener('keydown', GJWin_OnKeyDown)
8394 gjmode.addEventListener('keydown', GJWin_OnKeyDown)
8395 gjmode.id = wid + ' Mode'
8396 gjmode.setAttribute('class', 'GJMode')
8397 gjmode.setAttribute('spellcheck', 'false')
8398 gjmode.innerHTML = '[---]'
8399 gjwin.appendChild(gjmode)
8400
8401 gjwin.zIndex = 30000
8402 GJFactory.appendChild(gjwin)
8403
8404 gjtab.scrollTop = 0
8405 gjstat.scrollTop = 0
8406
8407 //x = gjwin.getBoundingClientRect().left.toFixed(0)
8408 //y = gjwin.getBoundingClientRect().top.toFixed(0)
8409 //gjwin.style.position = 'static'
8410 //gjwin.style.left = 0
8411 //gjwin.style.top = 0
8412
8413 //update = '{'+wid+' value=DateShort()}',
8414 update = '{GJ_showTime('+wid+'));',
8415 // 2020-09-19 this causes memory leaks
8416 //FProductInterval = window.setInterval(update,200)
8417 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
8418 //FProductInterval = window.setInterval(GJ_showTime,200,wid);
8419 FProductInterval = window.setInterval(GJ_showTime,200,gjstat);
8420 return update
8421 }
8422 }
8423 function xxxGJF_StripeClass(){
8424 GJLog_Win.style.removeProperty('width')
8425 GJLog_Tab.style.removeProperty('width')
8426 GJLog_Stat.style.removeProperty('width')
8427 GJLog_Text.style.removeProperty('width')
8428 return "Stripped classes"
8429 }
8430 function isElem(id){
8431 return document.getElementById(id) != null
8432 }
8433 function GJLog_append(...args){
8434 txt = GJLog_Text;
8435 if( txt == null ){
8436 return; // maybe GJLog element is removed
8437 }
8438 logs = args.join(' ');
8439 txt.value += logs + '\n'
8440 txt.scrollTop = txt.scrollHeight
8441 //GJLog_Stat.value = DateShort()
8442 }
8443 //window.addEventListener('time',GJLog_StatUpdate)
8444 function test_GJ_Console(){
8445 window.setInterval(GJLog_StatUpdate,1000);
8446 GJ_NewConsole('GJ_Console')
8447 e = GJFactory;
8448 console.log('GJF0 #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
8449 e.style.width = 360; //GJFsize
8450 e.style.height = 320;
8451 console.log('GJF0 #' + e.id + ' to w=' + e.style.width + ', h=' + e.style.height)
8452 }
8453 /// test_GJ_Console();
8454
8455 var StopConsoleLog = true
8456 // 2020-09-15 added,
8457 // log should be saved to permanent memory
8458 // const px = new Proxy(console.log, { alert() })
8459
8460 console_log = console.log
8461 console_info = console.info
8462 console_warn = console.warn
8463 console_error = console.error
8464 console_exception = console.exception
8465 // should pop callstack info.
8466 console.exception = function(...args){
8467 console_exception(...args)
8468 alert('-- got console.exception('+args+')')
8469 }
8470 console.error = function(...args){
8471 console_error(...args)
8472 alert('-- got console.error('+args+')')
8473 }
8474 console.warn = function(...args){
8475 console_warn(...args)
8476 alert('-- got console.warn('+args+')')
8477 }
8478 console.info = function(...args){
8479 alert('-- got console.info('+args+')')
8480 console_info(...args)
8481 }
8482 console.xxxlog = function(...args){ // rewrite xxxlog to log to enable it
8483 console_log(...args)
8484 if( StopConsoleLog ){
8485 return;
8486 }
8487 if( 0 <= args[0].indexOf('!') ){
8488 //alert('-- got console.log('+args+')')
8489 }
8490 GJLog_append(...args)
8491 }
8492 //document.getElementById('GshFaviconURL').href = GShellFavicon
8493 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
8494 //document.getElementById('GshFaviconURL').href = ITSMoreQR
8495 //document.getElementById('GshFaviconURL').href = GShellLogo

```



```

8496
8497 // id of GShell HTML elemets
8498 var E_BANNER = "GshBanner" // banner element in HTML
8499 var E_FOOTER = "GshFooter" // footer element in HTML
8500 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
8501 var E_GOCODE = "gsh-gocode" // Golang code of GShell
8502 var E_TODO = "gsh-todo" // TODO of GShell
8503 var E_DICT = "gsh-dict" // dictionary of GShell
8504
8505 function bannerElem(){ return document.getElementById(E_BANNER); }
8506 function bannerStyleFunc(){ return bannerElem().style; }
8507 var bannerStyle = bannerStyleFunc()
8508 function GshSetImages(){
8509     document.getElementById('GshFaviconURL').href = GShellInsideIcon
8510     bannerStyle.backgroundImage = "url("+GShellLogo+")";
8511     //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
8512     //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
8513     //GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
8514     //showFooter();
8515 }
8516 function GshInsideIconSetup(){
8517     GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
8518     GMenu.style.zIndex = 1000000;
8519     //GMenu.style.left = window.innerWidth - 100
8520     GMenu.style.left = 0;
8521     GMenu.style.top = window.innerHeight - 90; // - 200
8522     window.addEventListener('resize',GshInsideIconSetup);
8523 }
8524
8525 function footerElem(){ return document.getElementById(E_FOOTER); }
8526 function footerStyle(){ return footerElem().style; }
8527 //footerElem().style.backgroundImages="url("+ITSmoreQR+")";
8528 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
8529
8530 function html_fold(e){
8531     if ( e.innerHTML == "Fold" ){
8532         e.innerHTML = "Unfold"
8533         document.getElementById('gsh-menu-exit').innerHTML=""
8534         document.getElementById('GshStatement').open=false
8535         GshFeatures.open = false
8536         document.getElementById('html-src').open=false
8537         document.getElementById(E_GINDEX).open=false
8538         document.getElementById(E_GOCODE).open=false
8539         document.getElementById(E_TODO).open=false
8540         document.getElementById('References').open=false
8541     }else{
8542         e.innerHTML = "Fold"
8543         document.getElementById('GshStatement').open=true
8544         GshFeatures.open = true
8545         document.getElementById(E_GINDEX).open=true
8546         document.getElementById(E_GOCODE).open=true
8547         document.getElementById(E_TODO).open=true
8548         document.getElementById('References').open=true
8549     }
8550 }
8551 function html_pure(e){
8552     if ( e.innerHTML == "Pure" ){
8553         document.getElementById('gsh').style.display=true
8554         //document.style.display = false
8555         e.innerHTML = "Unpure"
8556     }else{
8557         document.getElementById('gsh').style.display=false
8558         //document.style.display = true
8559         e.innerHTML = "Pure"
8560     }
8561 }
8562
8563 var bannerIsStopping = false
8564 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
8565 function shiftBG(){
8566     bannerIsStopping = !bannerIsStopping
8567     bannerStyle.backgroundPosition = "0 0";
8568 }
8569 // status should be inherited on Window Fork(), so use the status in DOM
8570 function html_stop(e,toggle){
8571     if( toggle ){
8572         if ( e.innerHTML == "Stop" ){
8573             bannerIsStopping = true
8574             e.innerHTML = "Start"
8575         }else{
8576             bannerIsStopping = false
8577             e.innerHTML = "Stop"
8578         }
8579     }else{
8580         // update JavaScript variable from DOM status
8581         if ( e.innerHTML == "Stop" ){ // shown if it's running
8582             bannerIsStopping = false
8583         }else{
8584             bannerIsStopping = true
8585         }
8586     }
8587 }
8588 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
8589 //html_stop(bannerElem(),false) // onInit.
8590
8591 //https://www.w3schools.com/jsref/met_win_setinterval.asp
8592 var banNshift = 0;
8593 function consoleg(str){
8594     //console.log(str);
8595 }
8596 function shiftBanner(){
8597     var now = new Date().getTime();
8598     bpos = ((now/10)*1000).toFixed(0)+'px' + " 0px";
8599     if ( !bannerIsStopping ){
8600         bannerStyle.backgroundPosition = bpos;
8601         //GshBanner.style.setProperty('background-position',bpos,'important');
8602         banNshift += 1;
8603         consoleg('shiftBanner <'+GshBanner.nodeName+'> '+banNshift
8604             + ' now'+(now/10)
8605             + ' stop'+bannerIsStopping
8606             + ' pos'+bpos
8607             + ' -> '+bannerStyle.backgroundPosition);
8608     }
8609 }
8610 function Banner_init(){
8611     console.log('-- Banner Shift init. ');
8612     window.setInterval(shiftBanner,10); // onInit.
8613 }
8614
8615 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open(</a>
8616 // from embeded html to standalone page
8617 var MyChildren = 0
8618 function html_fork(){
8619     ResetPerfMon();
8620     ResetIdView();
8621     Reset_ShadingCanvas();
8622     GFactory_Destroy()
8623     MyChildren += 1
8624     WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
8625     newwin = window.open("",WinId,"");
8626     src = document.getElementById('gsh');
8627     srchtml = src.outerHTML
8628     newwin.document.write("/*<"+"html>\n");
8629     newwin.document.write(srchtml);
8630     newwin.document.write("<"+"html>\n");
8631     newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
8632     newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
8633     newwin.document.close();
8634     newwin.focus();
8635 }
8636 function html_close(){
8637     window.close()
8638 }
8639 function win_jump(win){
8640     //win = window.top;
8641     win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
8642     if( win == null ){
8643         console.log("jump to window.opener("+win+") (Error)\n")
8644     }else{
8645         console.log("jump to window.opener("+win+")\n")
8646         win.focus();
8647     }
8648 }
8649
8650 // 0.2.9 2020-0902 created checksum of HTML
8651 CRC32UNIX = 0x04C1DB7 // Unix cksun
8652 function byteCRC32Add(bigcrc,octstr,octlen){
8653     var crc = new Int32Array(1)
8654     crc[0] = bigcrc
8655
8656     let oi = 0
8657     for( ; oi < octlen; oi++ ){
8658         var oct = new Int8Array(1)
8659         oct[0] = octstr[oi]
8660         for( bi = 0; bi < 8; bi++ ){
8661             //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
8662             ovf1 = crc[0] < 0 ? 1 : 0
8663             ovf2 = oct[0] < 0 ? 1 : 0
8664             ovf = ovf1 ^ ovf2
8665             oct[0] <<= 1
8666             crc[0] <<= 1
8667             if( ovf ){ crc[0] ^= CRC32UNIX }
8668         }
8669     }
8670     //console.log("--CRC32 byteAdd return crc="+crc[0]+" "+oi+"/"+octlen+"\n")
8671     return crc[0];
8672 }

```

```

8673 function strCRC32add(bigcrc,stri,strlen){
8674     var crc = new Uint32Array(1)
8675     crc[0] = bigcrc
8676     var code = new Uint8Array(strlen);
8677     for( i = 0; i < strlen; i++){
8678         code[i] = stri.charCodeAt(i) // not charAt(i) !!!!
8679         //console.log("code["+i+"]="+code[i].toString(16)+" <== "+stri[i]+"\\n")
8680     }
8681     crc[0] = byteCRC32add(crc,code,strlen)
8682     //console.log("---CRC32 strAdd return crc="+crc[0]+"\\n")
8683     return crc[0]
8684 }
8685 function byteCRC32end(bigcrc,len){
8686     var crc = new Uint32Array(1)
8687     crc[0] = bigcrc
8688     var slen = new Uint8Array(4)
8689     let li = 0
8690     for( ; li < 4; ){
8691         slen[li] = len
8692         li += 1
8693         len >>= 8
8694         if( len == 0 ){
8695             break
8696         }
8697     }
8698     crc[0] = byteCRC32add(crc[0],slen,li)
8699     crc[0] ^= 0xFFFFFFFF
8700     return crc[0]
8701 }
8702 function strCRC32(stri,len){
8703     var crc = new Uint32Array(1)
8704     crc[0] = 0
8705     crc[0] = strCRC32add(0,stri,len)
8706     crc[0] = byteCRC32end(crc[0],len)
8707     //console.log("---CRC32 "+crc[0]+" "+len+"\\n")
8708     return crc[0]
8709 }
8710
8711 DestroyGJLink = null; // to be replaced
8712 DestroyFooter = null; // to be defined
8713 DestroyEventSharingCodeView = function dummy(){}
8714 DestroyVirtualDesktop = function(){}
8715 DestroyNaviButtons = function(){}
8716
8717 function getSourceText(){
8718     if( DestroyFooter != null ) DestroyFooter();
8719     version = document.getElementById('GshVersion').innerHTML
8720     sfavico = document.getElementById('GshFaviconURL').href;
8721     sbanner = document.getElementById('GshBanner').style.backgroundColor;
8722     spositr = document.getElementById('GshBanner').style.backgroundPosition;
8723
8724     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
8725     if( DestroyGJLink != null ) DestroyGJLink();
8726     DestroyEventSharingCodeView();
8727     DestroyVirtualDesktop();
8728     GshTopbar.innerHTML = "";
8729     DestroyIndexBar();
8730     DestroyNaviButtons();
8731     ResetPerfMon();
8732     ResetActiveView();
8733     ResetShadingCanvas();
8734
8735     // these should be removed by CSS selector or class, after seaved to non-printed attribute
8736     GshBanner.removeAttribute("style");
8737     document.getElementById('GshMenuSign').removeAttribute("style");
8738     styleGMenu = GMenu.getAttribute("style");
8739     GMenu.removeAttribute("style");
8740     styleGStat = GStat.getAttribute("style");
8741     GStat.removeAttribute("style");
8742     styleGTop = GTop.getAttribute("style");
8743     GTop.removeAttribute("style");
8744     styleGshGrid = GshGrid.getAttribute("style");
8745     GshGrid.removeAttribute("style");
8746     //styleGPos = GPos.getAttribute("style");
8747     //GPos.removeAttribute("style");
8748     //GPos.innerHTML = "";
8749     //styleGLog = GLog.getAttribute("style");
8750     //GLog.removeAttribute("style");
8751     //GLog.innerHTML = "";
8752     styleGShellPlane = GShellPlane.getAttribute("style");
8753     GShellPlane.removeAttribute("style");
8754     styleRawTextViewer = RawTextViewer.getAttribute("style")
8755     RawTextViewer.removeAttribute("style")
8756     styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
8757     RawTextViewerClose.removeAttribute("style")
8758
8759     GshFaviconURL.href = "";
8760     if( iselem('ConfigIcon') ) ConfigIcon.src = "";
8761
8762     //it seems that interHTML and outerHTML generate style="" for these (??)
8763     //GshBanner.removeAttribute("style");
8764     //GshFooter.removeAttribute("style");
8765     //GshMenuSign.removeAttribute("style");
8766     GshBanner.style=""
8767     GshMenuSign.style=""
8768
8769     textarea = document.createElement("textarea")
8770     srchtml = document.getElementById("gsh").outerHTML;
8771     //textarea = document.createElement("textarea")
8772     // 2020-0910 ? ? this causes inserting style="" to Banner and Footer,
8773     // with Chromium?/ after reloading from file:///
8774     textarea.innerHTML = srchtml
8775     // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
8776     var rawtext = textarea.value
8777     //textarea.destroy()
8778     //rawtext = gsh.textContent // this removes #include <FILENAME> too
8779     var orgtext
8780     + "<"+rawtext>\\n" // lost preamble text
8781     + rawtext
8782     + "<"+rawtext>\\n" // lost trail text
8783     ;
8784
8785     tlen = orgtext.length
8786     //console.log("getSourceText: length="+tlen+"\\n")
8787     document.getElementById('GshFaviconURL').href = sfavico;
8788
8789     document.getElementById('GshBanner').style.backgroundColor = sbanner;
8790     document.getElementById('GshBanner').style.backgroundPosition = spositr;
8791
8792     GStat.setAttribute("style",styleGStat)
8793     GMenu.setAttribute("style",styleGMenu)
8794     GTop.setAttribute("style",styleGTop)
8795     //GLog.setAttribute("style",styleGLog)
8796     //GPos.setAttribute("style",styleGPos)
8797     GshGrid.setAttribute("style",styleGshGrid)
8798     GShellPlane.setAttribute("style",styleGShellPlane)
8799     RawTextViewer.setAttribute("style",styleRawTextViewer)
8800     RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
8801     canonext = orgtext.replace(' style=""',',')
8802     // open="" too
8803     return canonext
8804 }
8805
8806 function getDigest(){
8807     var text = ""
8808     text = getSourceText()
8809     var digest = ""
8810     tlen = text.length
8811     digest = strCRC32(text,tlen) + " " + tlen
8812     return { text, digest }
8813 }
8814
8815 function html_digest(){
8816     version = document.getElementById('GshVersion').innerHTML
8817     let {text, digest} = getDigest()
8818     alert("cksum: " + digest + " " + version)
8819 }
8820
8821 function charsin(stri,char){
8822     ln = 0;
8823     for( i = 0; i < stri.length; i++ ){
8824         if( stri.charCodeAt(i) == char.charCodeAt(0) )
8825             ln++;
8826     }
8827     return ln;
8828 }
8829
8830 //class digestElement extends HTMLElement {
8831 //< script>customElements.define('digest',digestElement)< /script>
8832 function showDigest(e){
8833     result = 'version=' + GshVersion.innerHTML + '\\n'
8834     result += 'lines=' + e.dataset.lines + '\\n'
8835     + 'length=' + e.dataset.length + '\\n'
8836     + 'crc32u=' + e.dataset.crc32u + '\\n'
8837     + 'time=' + e.dataset.time + '\\n';
8838     alert(result)
8839 }
8840
8841 function html_sign(e){
8842     if( RawTextViewer.style.zIndex == 1000 ){
8843         hideRawTextViewer()
8844         return
8845     }
8846     GshTopbar.innerHTML = "";
8847     ResetPerfMon();
8848     ResetActiveView();
8849     ResetShadingCanvas();
8850     DestroyIndexBar();
8851     DestroyNaviButtons();
8852 }

```

```

8850 DestroyEventSharingCodeview());
8851 Destroy_VirtualDesktop();
8852 GJFactory_Destroy();
8853 if( DestroyGJLink != null ) DestroyGJLink();
8854 //gsh_digest_innerHTML = "";
8855 text = getSourceText() // the original text
8856 tlen = text.length
8857 digest = strCRC32(text,tlen)
8858 //gsh_digest_innerHTML = digest + " " + tlen
8859 //text = getSourceText() // the text with its digest
8860 Lines = charsIn(text,'\n')
8861
8862 name = "gsh"
8863 sid = name + "-digest"
8864 d = new Date()
8865 signedAt = d.getTime()
8866
8867 sign = '/'+'<'+'span'\n'
8868 + ' id="' + sid + '"\n'
8869 + ' class="digest"\n'
8870 + ' data-target-id="name+"\n'
8871 + ' data-crc32u="' + digest + '"\n'
8872 + ' data-length="' + tlen + '"\n'
8873 + ' data-lines="' + Lines + '"\n'
8874 + ' data-time="' + signedAt + '"\n'
8875 + '>' + '/span>\n'+'/\n'
8876
8877 text = sign + text
8878
8879 txhtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td'
8880 + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
8881 for( i = 1; i <= Lines; i++ ){
8882   txhtml += i.toString() + '\n'
8883 }
8884 txhtml += ""
8885 + '<' + '/textarea>'
8886 + '<' + '/td><' + 'td'
8887 + '<' + 'td><' + 'td'
8888 + ' class="LineNumbered">'
8889 + text + '<' + '/textarea>'
8890 + '<' + '/td><' + 'tr><' + '/table>'
8891
8892 for( i = 1; i <= 30; i++ ){
8893   txhtml += '<br>\n'
8894 }
8895 RawTextViewer.innerHTML = txhtml
8896 RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
8897
8898 btn = e
8899 e.style.color = "rgba(128,128,255,0.9)";
8900 y = e.getBoundingClientRect().top.toFixed(0)
8901 //h = e.getBoundingClientRect().height.toFixed(0)
8902 RawTextViewer.style.top = Number(y) + 30
8903 RawTextViewer.style.left = 100;
8904 RawTextViewer.style.height = window.innerHeight - 20;
8905 //RawTextViewer.style.opacity = 1.0;
8906 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
8907 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
8908 RawTextViewer.style.zIndex = 1000;
8909 RawTextViewer.style.display = true;
8910
8911 if( RawTextViewerClose.style == null ){
8912   RawTextViewerClose.style = "";
8913 }
8914 RawTextViewerClose.style.top = Number(y) + 10
8915 RawTextViewerClose.style.left = 100;
8916 RawTextViewerClose.style.zIndex = 1001;
8917 ScrollToElement(CurElement,RawTextViewerClose)
8918 }
8919 function hideRawTextViewer(){
8920   RawTextViewer.style.left = 10000;
8921   RawTextViewer.style.zIndex = -100;
8922   RawTextViewer.style.opacity = 0.0;
8923   RawTextViewer.style = null
8924   RawTextViewer.innerHTML = "";
8925 }
8926
8927 GshMenuSign.style.color = "rgba(255,128,128,1.0)";
8928 RawTextViewerClose.style.top = 0;
8929 RawTextViewerClose.style = null
8930 }
8931
8932 // source code view
8933 function frame_close(){
8934   srcframe = document.getElementById("src-frame");
8935   srcframe.innHTML = "";
8936   //srcframe.style.cols = 1;
8937   //srcframe.style.rows = 1;
8938   //srcframe.style.height = 0;
8939   //srcframe.style.display = false;
8940   src = document.getElementById("SrcTextarea");
8941   src.innerHTML = ""
8942   //src.cols = 0
8943   //src.rows = 0
8944   //src.display = false
8945   //alert("--closed--")
8946 }
8947 //<!-- | <span onclick="html_view();">Source</span> -->
8948 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
8949 //<!-- | <span>Down</span> -->
8950 function frame_open(){
8951   GshTopbar.innerHTML = "";
8952   ResetParMon();
8953   ResetAffView();
8954   Reset_ShadingCanvas();
8955   DestroyIndexBar();
8956   DestroyVibButtons();
8957   if( DestroyFooter != null ) DestroyFooter();
8958   document.getElementById("GshFaviconURL").href = "";
8959   oldsrc = document.getElementById("GENSRC");
8960   if( oldsrc != null ){
8961     //alert("--I--(erasing old text)")
8962     oldsrc.innHTML = "";
8963     return
8964   }else{
8965     //alert("--I--(no old text)")
8966   }
8967   styleBanner = GshBanner.getAttribute("style")
8968   GshBanner.removeAttribute("style")
8969   if( document.getElementById("GJC_1") ){ GJC_1.remove() }
8970
8971   GshFaviconURL.href = "";
8972   if( iselem('ConfigIcon') ) ConfigIcon.src = "";
8973   GStat.removeAttribute('style')
8974   GshGrid.removeAttribute('style')
8975   GshMenuSign.removeAttribute('style')
8976   //Gpos.removeAttribute('style')
8977   //Glog.innerHTML = "";
8978   //Glog.removeAttribute('style')
8979   //Glog.innerHTML = "";
8980   GMenu.removeAttribute('style')
8981   GTop.removeAttribute('style')
8982   GShellPlane.removeAttribute('style')
8983   RawTextViewer.removeAttribute('style')
8984   RawTextViewerClose.removeAttribute('style')
8985
8986   if( DestroyGJLink != null ) DestroyGJLink();
8987   GJFactory_Destroy()
8988   Destroy_VirtualDesktop()
8989   DestroyEventSharingCodeview();
8990
8991   src = document.getElementById("gsh");
8992   srhtml = src.outerHTML
8993   srcframe = document.getElementById("src-frame");
8994   srcframe.innerHTML =
8995   + "<"+"cite id=\"GENSRC">\n"
8996   + "<"+"style>\n"
8997   + "#GENSRC textarea(tab-size:4;)\n"
8998   + "#GENSRC textarea(-moz-tab-size:4;)\n"
8999   + "#GENSRC textarea(-moz-tab-size:4;)\n"
9000   + "#GENSRC textarea(spellcheck:false;)\n"
9001   + "<\/+style>\n"
9002   + "textarea id='SrcTextarea' cols=100 rows=20 class='gsh-code' spellcheck='false'>"
9003   + "<\/+html>\n" // lost preamble text
9004   + srhtml
9005   + "<\/html>\n" // lost trail text
9006   + "<\/+textarea>\n"
9007   + "<\/+cite><!-- GENSRC -->\n";
9008
9009   //srcframe.style.cols = 80;
9010   //srcframe.style.rows = 80;
9011
9012   GshBanner.setAttribute('style',styleBanner)
9013 }
9014 function fill_CSSView(){
9015   part = document.getElementById("GshStyleDef")
9016   view = document.getElementById("gsh-style-view")
9017   view.innerHTML = ""
9018   + "<"+"textarea cols=100 rows=20 class='gsh-code'>"
9019   + part.innerHTML
9020   + "<\/+textarea>"
9021 }
9022 function fill_JavaScriptView(){
9023   jspart = document.getElementById("gsh-script")
9024   view = document.getElementById("gsh-script-view")
9025   view.innerHTML = ""
9026   + "<"+"textarea cols=100 rows=20 class='gsh-code'>"

```

```

9027     + jspart.innerHTML
9028     + "<+\"/textarea>"
9029 }
9030 function fill_DataView(){
9031     part = document.getElementById('gsh-data')
9032     view = document.getElementById('gsh-data-view')
9033     view.innerHTML =
9034     + "<+\"/textarea cols=100 rows=20 class='gsh-code'>"
9035     + part.innerHTML
9036     + "<+\"/textarea-"
9037 }
9038 function jumpto_StyleView(){
9039     jsview = document.getElementById('html-src')
9040     jsview.open = true
9041     jsview = document.getElementById('gsh-style-frame')
9042     jsview.open = true
9043     fill_CSSView()
9044 }
9045 function jumpto_JavaScriptView(){
9046     jsview = document.getElementById('html-src')
9047     jsview.open = true
9048     jsview = document.getElementById('gsh-script-frame')
9049     jsview.open = true
9050     fill_JavaScriptView()
9051 }
9052 function jumpto_DataView(){
9053     jsview = document.getElementById('html-src')
9054     jsview.open = true
9055     jsview = document.getElementById('gsh-data-frame')
9056     jsview.open = true
9057     fill_DataView()
9058 }
9059 function jumpto_WholeView(){
9060     jsview = document.getElementById('html-src')
9061     jsview.open = true
9062     jsview = document.getElementById('gsh-whole-view')
9063     jsview.open = true
9064     frame_open()
9065 }
9066 function html_view(){
9067     html_stop();
9068 }
9069 banner = document.getElementById('GshBanner').style.backgroundImage;
9070 footer = document.getElementById('GshFooter').style.backgroundImage;
9071 document.getElementById('GshBanner').style.backgroundImage = "";
9072 document.getElementById('GshBanner').style.backgroundColor = "";
9073 document.getElementById('GshFooter').style.backgroundImage = "";
9074
9075 //srcwin = window.open("", "CodeView2", "");
9076 srcwin = window.open("", "t", "");
9077 srcwin.document.write("<span id='gsh'>\n");
9078
9079 src = document.getElementById("gsh");
9080 srcwin.document.write("<+\"style>\n");
9081 srcwin.document.write("textarea{tab-size:4;}\n");
9082 srcwin.document.write("textarea{-o-tab-size:4;}\n");
9083 srcwin.document.write("textarea{-moz-tab-size:4;}\n");
9084 srcwin.document.write("</style>\n");
9085 srcwin.document.write("<h2>\n");
9086 srcwin.document.write("<+\"span onclick='\"window.close();\">Close</span> | \n");
9087 //srcwin.document.write("<+\"span onclick='\"html_stop();\">Run</span>\n");
9088 srcwin.document.write("</h2>\n");
9089 srcwin.document.write("<+\"textarea id='gsh-src-src' cols=100 rows=60>");
9090 srcwin.document.write("<+\"html>\n");
9091 srcwin.document.write("<+\"span id='gsh'>");
9092 srcwin.document.write(src.innerHTML);
9093 srcwin.document.write("<+\"/span><+\"/html>\n");
9094 srcwin.document.write("<+\"/textarea>\n");
9095
9096 document.getElementById('GshBanner').style.backgroundImage = banner;
9097 document.getElementById('GshFooter').style.backgroundImage = footer
9098
9099 sty = document.getElementById("GshStyleDef");
9100 srcwin.document.write("<+\"style>\n");
9101 srcwin.document.write(sty.innerHTML);
9102 srcwin.document.write("<+\"/style>\n");
9103
9104 run = document.getElementById("gsh-script");
9105 srcwin.document.write("<+\"script>\n");
9106 srcwin.document.write(run.innerHTML);
9107 srcwin.document.write("<+\"/script>\n");
9108
9109 srcwin.document.write("<+\"/span><+\"/html>\n"); // gsh span
9110 srcwin.document.close();
9111 srcwin.focus();
9112 }
9113 GSH = document.getElementById("gsh")
9114
9115 //GSH.onclick = "alert('Ouch!')"
9116 //GSH.css = "{background-color:#eef;}";
9117 //GSH.style = "background-color:#eef;";
9118 //GSH.style.display = false;
9119 //alert('Ouch0!');
9120 //GSH.style.display = true;
9121
9122 // 2020-0904 created, tentative
9123 //document.addEventListener('keydown', jgshCommand);
9124 //CurElement = GshStatement
9125 CurElement = GshMenu
9126 MomElement = GshMenu
9127
9128 function nextSib(e){
9129     n = e.nextSibling;
9130     for( i = 0; i < 100; i++ ){
9131         if( n == null ){
9132             break;
9133         }
9134         if( n.nodeName == "DETAILS" ){
9135             return n;
9136         }
9137         n = n.nextSibling;
9138     }
9139     return null;
9140 }
9141 function prevSib(e){
9142     n = e.previousSibling;
9143     for( i = 0; i < 100; i++ ){
9144         if( n == null ){
9145             break;
9146         }
9147         if( n.nodeName == "DETAILS" ){
9148             return n;
9149         }
9150         n = n.previousSibling;
9151     }
9152     return null;
9153 }
9154 function setColor(e,eName,eColor){
9155     if( e.hasChildNodes() ){
9156         s = e.childNodes;
9157         if( s != null ){
9158             for( ci = 0; ci < s.length; ci++ ){
9159                 if( s[ci].nodeName == eName ){
9160                     s[ci].style.color = eColor;
9161                     //s[ci].style.backgroundColor = eColor;
9162                     break;
9163                 }
9164             }
9165         }
9166     }
9167 }
9168
9169 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
9170 function showCurElementPosition(ev){
9171     // if( document.getElementById("GPos") == null ){
9172     //     return;
9173     // }
9174     // if( GPos == null ){
9175     //     return;
9176     // }
9177     e = CurElement
9178     y = e.getBoundingClientRect().top.toFixed(0)
9179     x = e.getBoundingClientRect().left.toFixed(0)
9180
9181     h = ev + " "
9182     h += "y=" + y + " " + "x=" + x + " -- "
9183     h += "w=" + window.innerWidth + " , h=" + window.innerHeight + " -- "
9184     //GPos.test = h
9185     //GPos.innerHTML = h
9186     // GPos.innerHTML = h
9187 }
9188
9189 function zero2(n){
9190     if( n < 10 ){
9191         return '0' + n;
9192     }else{
9193         return n;
9194     }
9195 }
9196 function DateHourMin(){
9197     d = new Date();
9198     //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
9199     return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
9200 }
9201 function DateShort0(d){
9202     return d.getFullYear()
9203         + '/' + zero2(d.getMonth())

```

```

9204     + '/' + zero2(d.getDate())
9205     + ' ' + zero2(d.getHours())
9206     + ':' + zero2(d.getMinutes())
9207     + ':' + zero2(d.getSeconds())
9208 }
9209 function DateShort(){
9210     return DateShort0(new Date());
9211 }
9212 function DateLong0(ms){
9213     d = new Date();
9214     d.setTime(ms);
9215     return DateShort0(d)
9216     + ' ' + d.getMilliseconds()
9217     + ' ' + d.getTimezoneOffset()/60
9218     + ' ' + d.getTime() + ' ' + d.getMilliseconds()
9219 }
9220 function DateLong(){
9221     return DateLong0(new Date());
9222 }
9223 function GShellMenu(e){
9224     //GLog.innerHTML += "Hello, World! (" + DateLong() + ") "
9225     //showGShellPlane()
9226     ConfigClick();
9227 }
9228 // placements of planes
9229 function GShellResizeX(ev){
9230     //if( document.getElementById("GMenu") != null ){
9231         //GShellIconSetup();
9232         //GMenu.style.left = window.innerWidth - 100
9233         //GMenu.style.top = window.innerHeight - 90 - 200
9234         //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
9235     }
9236     //
9237     GStat.style.width = window.innerWidth
9238     //if( document.getElementById("GPos") != null ){
9239         //GPos.style.width = window.innerWidth
9240         //GPos.style.top = window.innerHeight - 30; //GPos.style.height
9241     }
9242     //
9243     //if( document.getElementById("GLog") != null ){
9244         // GLog.style.width = window.innerWidth
9245         //GLog.innerHTML = ""
9246     }
9247     //if( document.getElementById("GLog") != null ){
9248         //GLog.innerHTML = "Resize: w=" + window.innerWidth +
9249         //", h=" + window.innerHeight
9250     }
9251     showCurElementPosition(ev)
9252 }
9253 function GShellResize(){
9254     GShellResizeX("RESIZE")
9255 }
9256 window.onresize = GShellResize
9257 var prevNode = null
9258 var LogMouseMoveOverElement = false;
9259 function GJSH_OnMouseMove(ev){
9260     if( LogMouseMoveOverElement == false ){
9261         return;
9262     }
9263     x = ev.clientX
9264     y = ev.clientY
9265     d = new Date()
9266     t = d.getTime() / 1000
9267     if( document.elementFromPoint(x,y)
9268         e = document.elementFromPoint(x,y)
9269         if( e != null ){
9270             if( e == prevNode ){
9271                 console.log('Mo-'+t+'('+x+', '+y+') '
9272                     +e.nodeType+' '+e.tagName+'#'+e.id)
9273                 prevNode = e
9274             }
9275             }else{
9276                 console.log(t+'('+x+', '+y+') no element')
9277             }
9278         }else{
9279             console.log(t+'('+x+', '+y+') no elementFromPoint')
9280         }
9281     }
9282 window.addEventListener('mousemove',GJSH_OnMouseMove);
9283 }
9284 function GJSH_OnMouseMoveScreen(ev){
9285     x = ev.screenX
9286     y = ev.screenY
9287     d = new Date()
9288     t = d.getTime() / 1000
9289     console.log(t+'('+x+', '+y+') no elementFromPoint')
9290 }
9291 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
9292 }
9293 function ScrollToElement(oe,ne){
9294     ne.scrollIntoView()
9295     ny = ne.getBoundingClientRect().top.toFixed(0)
9296     dx = ne.getBoundingClientRect().left.toFixed(0)
9297     //GLog.innerHTML += "["+ny+", "+nx+"]"
9298     //window.scrollTo(0,0)
9299 }
9300 GTop.style.backgroundColor = "rgba(0,0,0,0.4)"
9301 GshGrid.style.left = "250px";
9302 GshGrid.style.zindex = 0
9303 if( false ){
9304     oy = oe.getBoundingClientRect().top.toFixed(0)
9305     ox = oe.getBoundingClientRect().left.toFixed(0)
9306     y = e.getBoundingClientRect().top.toFixed(0)
9307     x = e.getBoundingClientRect().left.toFixed(0)
9308     window.scrollTo(x,y)
9309     ny = e.getBoundingClientRect().top.toFixed(0)
9310     nx = e.getBoundingClientRect().left.toFixed(0)
9311     //GLog.innerHTML += "["+oy+", "+ox+"]->["+y+", "+x+"]->["+ny+", "+nx+"]"
9312 }
9313 }
9314 function showGShellPlane(){
9315     if( GShellPlane.style.zindex == 0 ){
9316         GShellPlane.style.zindex = 1000;
9317         GShellPlane.style.left = 30;
9318         GShellPlane.style.height = 320;
9319         GShellPlane.innerHTML = DateLong() + "<br>" +
9320         "-- History --<br>" + MyHistory;
9321     }else{
9322         GShellPlane.style.zindex = 0;
9323         GShellPlane.style.left = 0;
9324         GShellPlane.style.height = 50;
9325         GShellPlane.innerHTML = "";
9326     }
9327 }
9328 var SuppressGJShell = false
9329 function jgshCommand(kevent){
9330     if( SuppressGJShell ){
9331         return
9332     }
9333     key = kevent
9334     keycode = key.code
9335     //GStat.style.width = window.innerWidth
9336     GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
9337 }
9338 console.log("JSGsh-Key:"+keycode+"(^~)"/)
9339 if( keycode == "Slash" ){
9340     console.log('('+x+', '+y+') '
9341         e = document.elementFromPoint(x,y)
9342     console.log('('+x+', '+y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
9343 }else
9344 if( keycode == "Digit0" ){ // fold side-bar
9345     // "hero page"
9346     showGShellPlane();
9347 }else
9348 if( keycode == "Digit1" ){ // fold side-bar
9349     primary.style.width = "94%"
9350     secondary.style.width = "0%"
9351     secondary.style.opacity = 0
9352     GStat.innerHTML = "[Single Column View]"
9353 }else
9354 if( keycode == "Digit2" ){ // unfold side-bar
9355     primary.style.width = "58%"
9356     secondary.style.width = "36%"
9357     secondary.style.opacity = 1
9358     GStat.innerHTML = "[Double Column View]"
9359 }else
9360 if( keycode == "KeyU" ){ // fold/unfold all
9361     html_fold(GshMenuFold);
9362     location.href = "#"+CurElement.id;
9363 }else
9364 if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
9365     CurElement.open = !CurElement.open;
9366 }else
9367 if( keycode == "ArrowRight" ){ // unfold the element
9368     CurElement.open = true
9369 }else
9370 if( keycode == "ArrowLeft" ){ // unfold the element
9371     CurElement.open = false
9372 }else
9373 if( keycode == "KeyI" ){ // inspect the element
9374     e = CurElement
9375     //GLog.innerHTML =
9376     GJLog_append("Current Element: " + e + "<br>"
9377         + "name="+e.nodeName + ", "
9378         + "id="+e.id + ", "
9379         + "children="+e.childNodes.length + ", "
9380         + "parent="+e.parentNode.id + "<br>"

```

```

9381         + "text="+e.textContent)
9382         GStat.style.backgroundColor = "rgba(0,0,0.8)"
9383         return
9384     }else
9385     if( keycode == "KeyM" ){ // memory the position
9386         MemElement = CurElement
9387     }else
9388     if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
9389         e = nextSib(CurElement)
9390         if( e != null ){
9391             setColor(CurElement,"SUMMARY","#fff")
9392             setColor(e,"SUMMARY","#8f8") // should be complement ?
9393             oe = CurElement
9394             CurElement = e
9395             //Location.href = "#"+e.id;
9396             ScrollToElement(oe,e)
9397         }
9398     }else
9399     if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
9400         oe = CurElement
9401         e = prevSib(CurElement)
9402         if( e != null ){
9403             setColor(CurElement,"SUMMARY","#fff")
9404             setColor(e,"SUMMARY","#8f8") // should be complement ?
9405             CurElement = e
9406             //Location.href = "#"+e.id;
9407             ScrollToElement(oe,e)
9408         }else{
9409             e = document.getElementById("GshBanner")
9410             if( e != null ){
9411                 setColor(CurElement,"SUMMARY","#fff")
9412                 CurElement = e
9413                 ScrollToElement(oe,e)
9414             }else{
9415                 e = document.getElementById("primary")
9416                 if( e != null ){
9417                     setColor(CurElement,"SUMMARY","#fff")
9418                     CurElement = e
9419                     ScrollToElement(oe,e)
9420                 }
9421             }
9422         }
9423     }else
9424     if( keycode == "KeyR" ){
9425         location.reload()
9426     }else
9427     if( keycode == "KeyJ" ){
9428         GshGrid.style.top = '120px';
9429         GshGrid.innerHTML = '>_{Down}';
9430     }else
9431     if( keycode == "KeyK" ){
9432         GshGrid.style.top = '0px';
9433         GshGrid.innerHTML = '(-_{Up}';
9434     }else
9435     if( keycode == "KeyH" ){
9436         GshGrid.style.left = '0px';
9437         GshGrid.innerHTML = '(-_{Left}';
9438     }else
9439     if( keycode == "KeyL" ){
9440         //Glog.innerHTML +=
9441         GLog.append(
9442             'screen'+screen.width+'px'+<br>'+
9443             'window'+window.innerWidth+'px'+<br>'+
9444             )
9445         GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
9446         GshGrid.innerHTML = '(@_{Right}';
9447     }else
9448     if( keycode == "KeyS" ){
9449         html_stop(GshMenuStop,true)
9450     }else
9451     if( keycode == "KeyF" ){
9452         html_fork()
9453     }else
9454     if( keycode == "KeyC" ){
9455         window.close()
9456     }else
9457     if( keycode == "KeyD" ){
9458         html_digest()
9459     }else
9460     if( keycode == "KeyV" ){
9461         e = document.getElementById('gsh-digest')
9462         if( e != null ){
9463             showDigest(e)
9464         }
9465     }
9466 }
9467 showCurElementPosition("[+key.code+] --");
9468 //if( document.getElementById("GPos") != null ){
9469 //GPos.innerHTML += "[+key.code+] --"
9470 //}
9471 //GShellResizeX("[+key.code+] --");
9472 }
9473 var initGSKC = false;
9474 function GShell_initKeyCommands(){
9475     if( initGSKC ){ return; } initGSKC = true;
9476 }
9477 GShellResizeX("[INIT]");
9478 DisplaySize = "-- Display: "
9479 + "screen="+screen.width+'px, ' + 'window'+window.innerWidth+'px';
9480 }
9481 let {text, digest} = getDigest()
9482 //Glog.innerHTML +=
9483 GLog.append(
9484     "-- GShell: ' + GshVersion.innerHTML + '\n' +
9485     "-- Digest: ' + digest + '\n' +
9486     DisplaySize
9487     //"/>_<br>"+ "-- LastVisit:<br>" + MyHistory
9488 )
9489 GShellResizeX(null);
9490 }
9491 //GShell_initKeyCommands();
9492 }
9493 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
9494 //Convert a string into an ArrayBuffer
9495 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
9496 function str2ab(str) {
9497     const buf = new ArrayBuffer(str.length);
9498     const bufView = new Uint8Array(buf);
9499     for (let i = 0, strLen = str.length; i < strLen; i++) {
9500         bufView[i] = str.charCodeAt(i);
9501     }
9502     return buf;
9503 }
9504 function importPrivateKey(pem) {
9505     const binaryDerString = window.atob(pemContents);
9506     const binaryDer = str2ab(binaryDerString);
9507     return window.crypto.subtle.importKey(
9508         "pkcs8",
9509         binaryDer,
9510         {
9511             name: "RSA-PSS",
9512             modulusLength: 2048,
9513             publicExponent: new Uint8Array([1, 0, 1]),
9514             hash: "SHA-256",
9515         },
9516         true,
9517         ["sign"]
9518     );
9519 }
9520 //importPrivateKey(ppem)
9521 }
9522 //key = {}
9523 //buf = "abc"
9524 //enc = "xyzxxxxx"; //crypto.publicEncrypt(key,buf)
9525 //b64 = btoa(enc)
9526 //dec = atob(b64)
9527 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
9528 </script>
9529 */
9530
9531 //!-- ===== Work { ===== -->
9532 //<span id="PackmonGo_WorkCodeSpan">
9533 /*
9534 <details id="PackmonGo_Section"><summary id="PackmonGo_Summary">PackmonGo</summary>
9535 <!-- ----- PackmonGo // 2020-1025 SatoxITS { -->
9536 <h2>PackmonGo</h2>
9537 <div id="PackmonGo_1" class="PackmonGo">
9538 <canvas id="PackmonGo_1_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
9539 </div>
9540 <div id="PackmonGo_2" class="PackmonGo">
9541 <canvas id="PackmonGo_2_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
9542 </div>
9543 <div id="PackmonGo_3" class="PackmonGo">
9544 <canvas id="PackmonGo_3_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
9545 </div>
9546 <div id="PackmonGo_4" class="PackmonGo">
9547 <canvas id="PackmonGo_4_Canvas" class="PackmonGo_Canvas" width="400px" height="400px" draggable="true"></canvas>
9548 </div>
9549 <style>
9550 .PackmonGo {
9551     position:fixed !important;
9552     background-color:rgba(0,0,0,0.0);
9553 }
9554 .PackmonGo_Canvas {
9555     background-color:rgba(0,0,0,0.0);
9556 }
9557 </style>

```

```

9558 <script>
9559 var stopPackmonFlag = false;
9560 var PackmonInit = false;
9561 function stopPackmon(){
9562   stopPackmonFlag = !stopPackmonFlag;
9563 }
9564 function spawnPackmonGo(){
9565   if( PackmonInit == false ){
9566     pgw = 100;
9567     pgh = 100;
9568     pgo = 0.5;
9569
9570     ctx = PackmonGo_1_Canvas.getContext('2d');
9571     ctx.fillStyle = 'rgba(255,255,0,'+pgo+')';
9572     ctx.fillRect(0,0,pgw,pgh);
9573     base = PackmonGo_1;
9574     gsh.appendChild(base);
9575     base.style.zIndex = 1000;
9576     base.style.position = "fixed";
9577     base.style.left = 0 + 'px';
9578     base.style.top = 0 + 'px';
9579     base.xinc = 4;
9580     base.yinc = 4;
9581     base.scrx = 0;
9582     base.scry = 0;
9583     base.pgw = 100;
9584     base.pgh = 100;
9585     movePWindow(PackmonGo_1);
9586
9587     ctx = PackmonGo_2_Canvas.getContext('2d');
9588     ctx.fillStyle = 'rgba(127,255,127,'+pgo+')';
9589     ctx.fillRect(0,0,pgw,pgh);
9590     base = PackmonGo_2;
9591     gsh.appendChild(base);
9592     base.style.zIndex = 1001;
9593     base.style.position = "fixed";
9594     base.style.left = 200 + 'px';
9595     base.style.top = 0 + 'px';
9596     base.xinc = 4;
9597     base.yinc = 4;
9598     base.scrx = 0;
9599     base.scry = 0;
9600     base.pgw = 100;
9601     base.pgh = 100;
9602     movePWindow(PackmonGo_2);
9603
9604     ctx = PackmonGo_3_Canvas.getContext('2d');
9605     pgo = 0.3;
9606     ctx.fillStyle = 'rgba(64,64,255,'+pgo+')';
9607     ctx.fillRect(0,0,pgw,pgh);
9608     base = PackmonGo_3;
9609     gsh.appendChild(base);
9610     base.style.zIndex = 1002;
9611     base.style.position = "fixed";
9612     base.style.left = 200 + 'px';
9613     base.style.top = 0 + 'px';
9614     base.xinc = 4;
9615     base.yinc = 4;
9616     base.scrx = 0;
9617     base.scry = 0;
9618     base.soff = 0;
9619     base.pgw = 100;
9620     base.pgh = 100;
9621     movePWindow(PackmonGo_3);
9622
9623     ctx = PackmonGo_4_Canvas.getContext('2d');
9624     pgh = 400;
9625     ctx.beginPath();
9626     ctx.strokeStyle = 'rgba(0,0,0,0)';
9627     ctx.arc(200,200,200,0,Math.PI*2,true);
9628     ctx.stroke();
9629     pgo = 0.4;
9630     ctx.fillStyle = 'rgba(255,31,32,'+pgo+')';
9631     ctx.fill();
9632     base = PackmonGo_4;
9633     gsh.appendChild(base);
9634     base.style.zIndex = 1002;
9635     base.style.position = "fixed";
9636     base.style.left = 200 + 'px';
9637     base.style.top = 0 + 'px';
9638     base.xinc = 4;
9639     base.yinc = 4;
9640     base.scrx = 0;
9641     base.scry = 0;
9642     base.soff = 5000;
9643     base.pgw = pgw;
9644     base.pgh = pgh;
9645     movePWindow(PackmonGo_4);
9646   }
9647   function movePWindow(base){
9648     if( stopPackmonFlag ){
9649       return;
9650     }
9651     x = parseInt(base.style.left);
9652     y = parseInt(base.style.top);
9653     w = window.innerWidth;
9654     h = window.innerHeight;
9655     if( x < 0 || w-base.pgw < x ){
9656       base.xinc = -base.xinc;
9657     }
9658     x += base.xinc;
9659     if( y < 0 || h-base.pgh < y ){
9660       //yinc = -yinc;
9661       base.yinc = -base.yinc;
9662     }
9663     y += base.yinc;
9664     base.style.left = x + 'px';
9665     base.style.top = y + 'px';
9666     //console.log('PG x='+x+',y='+y);
9667     //window.setTimeout(movePG,10);
9668   }
9669   function movePScreen(base){
9670     if( stopPackmonFlag ){
9671       return;
9672     }
9673     sw = screen.width;
9674     sh = screen.height;
9675     d = new Date();
9676     s = d.getTime(); // milli-seconds
9677     sx = ((s+base.soff) % 10000) * (screen.width / 10000);
9678     sy = ((s+base.soff) % 5000) * (screen.height / 5000);
9679     x = sx - window.screenX;
9680     y = sy - window.screenY;
9681     base.style.left = x + 'px';
9682     base.style.top = y + 'px';
9683   }
9684   function movePScreenCircle(base){
9685     if( stopPackmonFlag ){
9686       return;
9687     }
9688     sw = screen.width;
9689     sh = screen.height;
9690     pgw = base.pgw;
9691     pgh = base.pgh;
9692     d = new Date();
9693     s = d.getTime(); // milli-seconds
9694     ds = s + base.soff; // delay
9695     vsw = pgw + sw + pgw;
9696     vsh = pgh + sh + pgh;
9697     sx = -pgw + vsw * ((ds % 10000)/10000);
9698     sy = -pgh + vsh * ((ds % 10000)/10000);
9699     x = sx - window.screenX;
9700     y = sy - window.screenY;
9701     base.style.left = x + 'px';
9702     base.style.top = y + 'px';
9703   }
9704   if( PackmonInit == false ){
9705     //window.setTimeout(movePG,mm300);
9706     window.setInterval(movePWindow,10,PackmonGo_1);
9707     window.setInterval(movePWindow,10,PackmonGo_2);
9708     window.setInterval(movePScreen,10,PackmonGo_3);
9709     window.setInterval(movePScreen,10,PackmonGo_4);
9710     window.setInterval(movePScreenCircle,10,PackmonGo_4);
9711     window.addEventListener('click',stopPackmon);
9712     PackmonInit = true;
9713     stopPackmonFlag = false;
9714   }
9715 }
9716 function PackmonGo_Setup(e){
9717   stopPackmon();
9718   spawnPackmonGo();
9719   if( e != null ){
9720     e.stopPropagation();
9721   }
9722 }
9723 PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
9724 if( PackmonGo_Section.open == true ){
9725   PackmonGo_Setup();
9726 }
9727 </script>
9728
9729 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
9730 <input id="PackmonGo_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
9731 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
9732 <span id="PackmonGo_WorkCodeView"></span>
9733 <script id="PackmonGo_WorkScript">
9734 function PackmonGo_openWorkCodeView(){

```

```

9735 function PackmonGo_showWorkCode(){
9736     showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
9737 }
9738 PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
9739 }
9740 PackmonGo_openWorkCodeView(); // should be invoked by an event
9741 </script>
9742 </details>
9743 <!-- Template_WorkCodeSpan -->
9744 *! //</span>
9745 //<!-- ===== Work } ===== -->
9746
9747
9748
9749 //<!-- ===== Work { ===== -->
9750 <span id="SightGlass_WorkCodeSpan">
9751 /*
9752 <details id="SightGlass"><summary>SightGlass</summary>
9753 <!-- ----- SightGlass // 2020-1023 SatoXITS -->
9754 <h2>SightGlass</h2>
9755 <details><summary>enter</summary>
9756 <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9757 <div id="SightGlass_l_BGScroffset" class="SightGlass_TextData">(0000, 0000)</div>
9758 <div id="SightGlass_l_CPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9759 <div id="SightGlass_l_BGPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9760 <div id="SightGlass_l_Wheel" class="SightGlass_TextData">Wheel</div>
9761 </details>
9762 <p>
9763 <div id="SightGlass_l_Window" class="SightGlass_Window" draggable="true"></div>
9764 </p>
9765 <style>
9766 .SightGlass_TextData {
9767     color:#000;
9768     font-size:10pt;
9769 }
9770 .SightGlass_Window {
9771     xzoom:2;
9772     resize:both;
9773     width:400px;
9774     height:320px;
9775     background-color:rgba(200,200,200,0.5);
9776     background-size:200%;
9777     xbackground-size:1280px 720px;
9778     background-image:url(WD-WallPaper03.png);
9779 }
9780 xbody {
9781     scroll-behavior:smooth;
9782 }
9783 </style>
9784 <script>
9785 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
9786 var SGScrInitX = 0;
9787 var SGScrInitY = 0;
9788 var SG_initX = 0;
9789 var SG_initY = 0;
9790 function SG_adjust1(){
9791     xy = window.screenX + ' ' + window.screenY;
9792     wh = window.innerWidth + ' x ' + window.innerHeight;
9793     xywh = "xy(' + xy + ') wh(' + wh + ')";
9794     if( SightGlass.open) SightGlass_l_BPosition.innerHTML = 'Browser: ' + xywh;
9795
9796     sg = SightGlass_l_Window;
9797     sgr = sg.getBoundingClientRect();
9798     xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
9799     wh = sgr.width + ' x ' + sgr.height;
9800     xywh = "xy(' + xy + ') wh(' + wh + ')";
9801     if( SightGlass.open) SightGlass_l_CPosition.innerHTML = 'SiGlass: ' + xywh;
9802
9803     //SightGlass_l_Window.style.backgroundColor = sgr.left+'px ' + sgr.top+'px';
9804     if( SG_initX == 0 ){
9805         SGScrInitX = window.screenX;
9806         SGScrInitY = window.screenY;
9807         //SightGlass_l_Window.style.backgroundColor = '200%';
9808         //SightGlass_l_Window.style.backgroundColor = 'url("WD-WallPaper03.png")';
9809         SG_initX = sgr.left;
9810         SG_initY = sgr.top;
9811     }
9812     dx = SG_initX - sgr.left;
9813     dy = SG_initY - sgr.top;
9814
9815     dsx = SGScrInitX - window.screenX;
9816     dsy = SGScrInitY - window.screenY;
9817     scroff = 'Screen: ' + 'sx'+dsx + ',sy'+dsy;
9818     if( SightGlass.open) SightGlass_l_BGScroffset.innerHTML = scroff;
9819     dx += dsx;
9820     dy += dsy;
9821
9822     bgpos = dx+'px ' + dy+'px';
9823     SightGlass_l_Window.style.backgroundColor = bgpos;
9824     if( SightGlass.open) SightGlass_l_BGPosition.innerHTML = 'BGround: '
9825         + SightGlass_l_Window.style.backgroundColor;
9826 }
9827
9828 var wheels = 0;
9829 function SG_wheel(e){
9830     wheels += 1;
9831     SightGlass_l_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
9832     if( e.target.id == 'SightGlass_l_Window' ){
9833         e.preventDefault();
9834     }
9835 }
9836
9837 function SG_adjust1(){
9838     SG_adjust();
9839     //sampling = 0;
9840     sampling = 20;
9841     //sampling = 200;
9842     window.setTimeout(SG_adjust1,sampling);
9843 }
9844
9845 function SightGlass_Setup(){
9846     SG_adjust();
9847     window.addEventListener('resize',SG_adjust);
9848     window.addEventListener('mousemove',SG_adjust);
9849     window.addEventListener('scroll',SG_adjust);
9850     window.addEventListener('wheel',SG_wheel,{passive:false});
9851     //window.moveTo(0,0);
9852
9853     // if focused
9854     //window.setInterval(SG_adjust,1);
9855     window.setTimeout(SG_adjust,1);
9856 }
9857 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
9858 function Electron_Setup(){
9859     const { BrowserWindow } = require('electron');
9860     const currentWindow = BrowserWindow.getFocusedWindow();
9861     //currentWindow.on('move',function(){ SG_adjust(); });
9862     currentWindow.addEventListener('move',SG_adjust);
9863 }
9864 </script>
9865
9866 <input id="SightGlass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
9867 <input id="SightGlass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
9868 <input id="SightGlass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
9869 <span id="SightGlass_WorkCodeView"></span>
9870 <script id="SightGlass_WorkScript">
9871 function SightGlass_openWorkCodeView(){
9872     function SightGlass_showWorkCode(){
9873         showHtmlCode(SightGlass_WorkCodeView,SightGlass_WorkCodeSpan);
9874     }
9875     SightGlass_WorkCodeViewOpen.addEventListener('click',SightGlass_showWorkCode);
9876 }
9877 SightGlass_openWorkCodeView(); // should be invoked by an event
9878 </script>
9879 </details>
9880 <!-- SightGlass_WorkCodeSpan -->
9881 *! //</span>
9882 //<!-- ===== Work } ===== -->
9883
9884
9885
9886 /*
9887 <!-- ----- GJConsole BEGIN { ----- -->
9888 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
9889 <details><summary>GJ Console</summary>
9890 <p>
9891 <span id="GJE_RootNode0"></span>
9892 <span id="GJC_Container"></span>
9893 </p>
9894 <style id="GJConsoleStyle">
9895 .GJConsole {
9896     z-index:1000;
9897     width:400; height:200px;
9898     margin:2px;
9899     color:#fff; background-color:#66a;
9900     font-size:12px; font-family:monospace,Courier New;
9901 }
9902 </style>
9903 <script id="GJConsoleScript" class="GJConsole">
9904 var P$1 = " "
9905 function GJC_Keydown(keyevent){
9906     key = keyevent.code
9907     if( key == "Enter" ){
9908         GJC_Command(this)
9909         this.value += "\n" + P$1 // prompt
9910     }else
9911     if( key == "Escape"){
9912         SuppressGJShell = false

```



```

9912     GshMenu.focus() // should be previous focus
9913 }
9914 }
9915 var GJC_SessionId
9916 function GJC_SetSessionId(){
9917     var xd = new Date()
9918     GJC_SessionId = xd.getTime() / 1000
9919 }
9920 GJC_SetSessionId()
9921 function GJC_Memory(mem,args,text){
9922     argv = args.split(' ')
9923     cmd = argv[0]
9924     argv.shift()
9925     args = argv.join(' ')
9926     ret = ""
9927
9928     if( cmd == 'clear' ){
9929         Permanent.setItem(mem,'')
9930     }else
9931     if( cmd == 'read' ){
9932         ret = Permanent.getItem(mem)
9933     }else
9934     if( cmd == 'save' ){
9935         val = Permanent.getItem(mem)
9936         if( val == null ){ val = "" }
9937         d = new Date()
9938         val += d.getTime()/1000+ "GJC_SessionId+" +document.URL+ " +args+"\n"
9939         val = text+val
9940         Permanent.setItem(mem,val)
9941     }else
9942     if( cmd == 'write' ){
9943         val = Permanent.getItem(mem)
9944         if( val == null ){ val = "" }
9945         d = new Date()
9946         val += d.getTime()/1000+ "GJC_SessionId+" +document.URL+ " +args+"\n"
9947         Permanent.setItem(mem,val)
9948     }else{
9949         ret = "Commands: write | read | save | clear"
9950     }
9951     return ret
9952 }
9953 // -- 2020-09-14 added TableEditor
9954 var GJE_CurElement = null; //GJE_RootNode
9955 GJE_NodeSaved = null
9956 GJE_TableNo = 1
9957 function GJE_StyleKeyCommand(kev){
9958     keycode = Kev.code
9959     console.log('GJE-Key: '+keycode)
9960     if( keycode == "Escape" ){
9961         GJE_SetStyle(this);
9962     }
9963     kev.stopPropagation()
9964     // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9965 }
9966 var GJE_CommandMode = false
9967 function GJE_TableKeyCommand(kev,tab){
9968     wasCmdMode = GJE_CommandMode
9969     key = kev.code
9970     if( key == "Escape" ){
9971         console.log("no command mode: "+tab.nodeName+"#"+tab.id)
9972         //tab.setAttribute('contenteditable','false')
9973         tab.style.caretColor = "blue"
9974         GJE_CommandMode = true
9975     }else
9976     if( key == "KeyA" ){
9977         tab.style.caretColor = "red"
9978         GJE_CommandMode = false
9979     }else
9980     if( key == "KeyI" ){
9981         tab.style.caretColor = "red"
9982         GJE_CommandMode = false
9983     }else
9984     if( key == "KeyO" ){
9985         tab.style.caretColor = "red"
9986         GJE_CommandMode = false
9987     }else
9988     if( key == "KeyJ" ){
9989         console.log("ROW-DOWN")
9990     }else
9991     if( key == "KeyK" ){
9992         console.log("ROW-UP")
9993     }else
9994     if( key == "KeyW" ){
9995         console.log("COL-FORW")
9996     }else
9997     if( key == "KeyB" ){
9998         console.log("COL-BACK")
9999     }
10000 }
10001 kev.stopPropagation()
10002 if( wasCmdMode ){
10003     kev.preventDefault()
10004 }
10005 }
10006 function GJE_DragEvent(ev,elem){
10007     x = ev.clientX
10008     y = ev.clientY
10009     console.log("Dragged: "+this.nodeName+"#"+this.id+" x="+x+" y="+y)
10010 }
10011 // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
10012 // https://www.w3.org/TR/uievents/#events-mouseevents
10013 function GJE_DropEvent(ev,elem){
10014     x = ev.clientX
10015     y = ev.clientY
10016     this.style.x = x
10017     this.style.y = y
10018     this.style.position = 'absolute' // 'fixed'
10019     this.parentNode = gsh // just for test
10020     console.log("Dropped: "+this.nodeName+"#"+this.id+" x="+x+" y="+y)
10021     + " parent="+this.parentNode.id)
10022 }
10023 function GJE_SetTableStyle(ev){
10024     this.innerHTML = this.value; // sync. for external representation?
10025     if(false){
10026         stid = this.parentNode.id+this.id
10027         // and remove "span" at the end
10028         e = document.getElementById(stid)
10029         //alert('SetTableStyle #' +e.id+'\n'+this.value)
10030         if( e != null ){
10031             e.innerHTML = this.value
10032         }else{
10033             console.log('Style Not found: '+stid)
10034         }
10035         //alert('event StopPropagation: '+ev)
10036     }
10037 }
10038 function setCSSofClass(cclass,cstyle){
10039     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
10040     rlen = ss.cssRules.length;
10041     let tabrule = null;
10042     rulen = -1
10043
10044     // should skip white space at the top of cstyle
10045     sel = cstyle.charAt(0);
10046     selector = sel+cclass;
10047     console.log("-- search style rule for '+selector')
10048
10049     for(let i = 0; i < rlen; i++){
10050         cr = ss.cssRules[i];
10051         console.log('CSS rule ['++i++/'+rlen+] '+cr.selectorText);
10052         if( cr.selectorText == selector ){ // css class selector
10053             tabrule = ss.cssRules[i];
10054             console.log('CSS rule found for:['++i++/'+rlen+] '+selector);
10055             ss.deleteRule(i);
10056             //rlen = ss.cssRules.length;
10057             rulen = i
10058             // should search and replace the property here
10059         }
10060     }
10061     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
10062     if( tabrule == null ){
10063         console.log('CSS rule NOT found for:['+rlen+] '+selector);
10064         ss.insertRule(cstyle,rlen);
10065         ss.insertRule(cstyle,0); // override by 0?
10066         console.log('CSS rule inserted:['+(rlen+1)+']\n'+cstyle);
10067     }else{
10068         ss.insertRule(cstyle,rlen);
10069         ss.insertRule(cstyle,0);
10070         console.log('CSS rule replaced:['+(rlen+1)+']\n'+cstyle);
10071     }
10072 }
10073 function GJE_SetStyle(te){
10074     console.log('Apply the style to:'+te.id+'\n');
10075     console.log('Apply the style to:'+te.parentNode.id+'\n');
10076     console.log('Apply the style to:'+te.parentNode.class+'\n');
10077     cclass = te.parentNode.class;
10078     setCSSofClass(cclass,te.value); // should get selector part from
10079     // selector { rules }
10080
10081     if(false){
10082         //console.log('Apply the style:')
10083         //stid = this.parentNode.id+this.id+"
10084         //stid = this.id+".style"
10085         css = te.value
10086         stid = te.parentNode.id+".style"
10087         e = document.getElementById(stid)
10088         if( e != null ){

```

```

10089 //console.log('Apply the style:'+e.id+'\n'+e.value);
10090 console.log('Apply the style:'+e.id+'\n'+css);
10091 //e.innerHTML = css; //te.value;
10092 //ncss = e.sheet;
10093 //ncss.insertRule(te.value,ncss.cssRules.length);
10094 }else{
10095 console.log('No element to Apply the style: '+stid)
10096 }
10097 tblad = te.parentNode.id+"-table";
10098 e = document.getElementById(tblad);
10099 if( e != null ){
10100 //e.setAttribute('style',css);
10101 e.setProperty('style',css,'!important');
10102 }
10103 }
10104 }
10105 function makeTable(argv){
10106 //tid = ""
10107 //cwe = GJE_CurElement
10108 cwe = GJCI_Container;
10109 //cwd = GJFactory;
10110 tid = 'table' + GJE_TableNo
10111
10112 nt = new Text('\n')
10113 cwe.appendChild(nt)
10114
10115 ne = document.createElement('span'); // the container
10116 cwe.appendChild(ne)
10117 ne.id = tid + '-span'
10118 ne.setAttribute('contenteditable',true)
10119
10120 htspan = document.createElement('span'); // html part
10121 //htspan.id = tid + '-html'
10122 //ne.innerHTML = '\n'
10123 nt = new Text('\n')
10124 ne.appendChild(nt)
10125 ne.appendChild(htspan)
10126
10127 htspan.id = tid
10128 htspan.setAttribute('class',tid)
10129
10130 ne.setAttribute('draggable','true')
10131 ne.addEventListener('drag',GJE_DragEvent);
10132 ne.addEventListener('dragend',GJE_DropEvent);
10133
10134 var col = 3
10135 var row = 2
10136 if( argv[0] != null ){
10137 col = argv[0]
10138 argv.shift()
10139 }
10140 if( argv[1] != null ){
10141 row = argv[1]
10142 argv.shift()
10143 }
10144
10145 //ne.setAttribute('class',tid)
10146 ht = "\n"
10147 //ht += '<+table ' + 'id="'+tid+'"' + ' class="'+tid+'"'
10148 ht += '<+table ' + 'id="'+tid+'"' + ' class="'+tid+'"'
10149 // + ' onkeydown="GJE_TableKeyCommand(event,this)"'
10150 // + ' ondrag="GJE_DragEvent(event,this)"\n'
10151 // + ' ondragend="GJE_DropEvent(event,this)"\n'
10152 // + ' draggable="true"\n'
10153 // + ' contenteditable="true"'
10154 // + >\n'
10155 ht += '<+tbody>\n';
10156 for( r = 0; r < row; r++ ){
10157 ht += "<+tr>\n"
10158 for( c = 0; c < col; c++ ){
10159 ht += "<+td"
10160 ht += " ABCDEFGHIJKLMNOPQRSTUVWXYZ. charAt(c) + r
10161 ht += "<+>\n"
10162 }
10163 ht += "<+tr>\n"
10164 }
10165 ht += '<+tbody>\n';
10166 ht += '<+table>\n';
10167 htspan.innerHTML = ht;
10168 nt = new Text('\n')
10169 ne.appendChild(nt)
10170
10171 st = '#'+tid+' *{\n' // # for instanse specific
10172 + ' border:1px solid #aaa;\n'
10173 + ' background-color:#efe;\n'
10174 + ' color:#222;\n'
10175 + ' font-size:#14pt !important;\n'
10176 + ' font-family:monospace,Courier New !important;\n'
10177 + ' } /* hit ESC to apply */\n'
10178
10179 // wish script to be included
10180 //nj = document.createElement('script')
10181 //ne.appendChild(nj)
10182 //ne.innerHTML = 'function SetStyle(e){}'
10183
10184 // selector seems lost in dynamic style appending
10185 if(false){
10186 ns = document.createElement('style')
10187 ne.appendChild(ns)
10188 ns.id = tid + '-style'
10189 ns.innerHTML = '\n'+st
10190 nt = new Text('\n')
10191 ne.appendChild(nt)
10192 }
10193 setCSSofClass(tid,st); // should be in JavaScript script?
10194
10195 nx = document.createElement('textarea')
10196 ne.appendChild(nx)
10197 nx.id = tid + '-style_def'
10198 nx.setAttribute('class','GJ_StyleEditor')
10199 nx.spellcheck = false
10200 nx.cols = 60
10201 nx.rows = 10
10202 nx.innerHTML = '\n'+st
10203 nx.addEventListener('change',GJE_SetTableStyle);
10204 nx.addEventListener('keydown',GJE_StyleKeyCommand);
10205 //nx.addEventListener('click',GJE_SetTableStyle);
10206
10207 nt = new Text('\n')
10208 cwe.appendChild(nt)
10209
10210 GJE_TableNo += 1
10211 return 'created TABLE id="'+tid+'"'
10212 }
10213 function GJE_NodeEdit(argv){
10214 cwe = GJE_CurElement
10215 cmd = argv[0]
10216 argv.shift()
10217 args = argv.join(' ')
10218 ret = ""
10219
10220 if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
10221 if( GJE_NodeSaved != null ){
10222 xn = GJE_RootNode
10223 GJE_RootNode = GJE_NodeSaved
10224 GJE_NodeSaved = xn
10225 ret = '-- did undo'
10226 }else{
10227 ret = '-- could not undo'
10228 }
10229 return ret
10230 }
10231 GJE_NodeSaved = GJE_RootNode.cloneNode()
10232 if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
10233 if( argv[0] == null ){
10234 ne = GJE_RootNode
10235 }else{
10236 if( argv[0] == '..' ){
10237 ne = cwe.parentNode
10238 }else{
10239 ne = document.getElementById(argv[0])
10240 }
10241 if( ne != null ){
10242 GJE_CurElement = ne
10243 ret = "-- current node: " + ne.id
10244 }else{
10245 ret = "-- not found: " + argv[0]
10246 }
10247 }else{
10248 if( cmd == '.mkt' || cmd == '.mktable' ){
10249 makeTable(argv)
10250 }else{
10251 if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
10252 ne = document.createElement(argv[0])
10253 //ne.id = argv[0]
10254 ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
10255 cwe.appendChild(ne)
10256 if( cmd == '.m' || cmd == '.mk' ){
10257 GJE_CurElement = ne
10258 }
10259 }else{
10260 if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
10261 cwe.id = argv[0]
10262 }else{
10263 if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
10264 }else{
10265 if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){

```

```

10266     s = argv.join(' ')
10267     cwe.innerHTML = s
10268 }else
10269 if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
10270     cwe.setAttribute(argv[0],argv[1])
10271 }else
10272 if( cmd == '.l' ){
10273 }else
10274 if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
10275     ret = cwe.innerHTML
10276 }else
10277 if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
10278     ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
10279     for( we = cwe.parentNode; we != null; ){
10280         ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
10281         we = we.parentNode
10282     }
10283 }else
10284 {
10285     ret = "Command: mk | rm \n"
10286     ret += " pw -- print current node\n"
10287     ret += " mk type -- make node with name and type\n"
10288     ret += " nm name -- set the id #name of current node\n"
10289     ret += " rm name -- remove named node\n"
10290     ret += " cd name -- change current node\n"
10291 }
10292 //alert(ret)
10293 return ret
10294 }
10295 function GJC_Command(text){
10296     lines = text.value.split('\n')
10297     line = lines[lines.length-1]
10298     argv = line.split(' ')
10299     text.value += '\n'
10300     if( argv[0] == '$' ){ argv.shift() }
10301     args0 = argv.join(' ')
10302     cmd = argv[0]
10303     argv.shift()
10304     args = argv.join(' ')
10305 }
10306 if( cmd == 'nolog' ){
10307     StopConsoleLog = true
10308 }else
10309 if( cmd == 'new' ){
10310     if( argv[0] == 'table' ){
10311         argv.shift()
10312         console.log('argv='+argv)
10313         text.value += makeTable(argv)
10314     }else
10315     if( argv[0] == 'console' ){
10316         text.value += GJ_NewConsole('GJ_Console')
10317     }else{
10318         text.value += "-- new { console | table }'
10319     }
10320 }else
10321 if( cmd == 'strip' ){
10322     //text.value += GJF_StripClass()
10323 }else
10324 if( cmd == 'css' ){
10325     sel = '#table 1'
10326     if( argv[0] == '0' ){
10327         rule1 = sel+'(color:#000 !important; background-color:#fff !important;);'
10328     }else
10329     rule1 = sel+'(color:#f00 !important; background-color:#eef !important;);'
10330     document.styleSheets[3].deleteRule(0);
10331     document.styleSheets[3].insertRule(rule1,0);
10332     text.value += "CSS rule added: "+rule1
10333 }else
10334 if( cmd == 'print' ){
10335     e = null;
10336     if( e == null ){
10337         e = document.getElementById('GJFactory_0')
10338     }
10339     if( e == null ){
10340         e = document.getElementById('GJFactory_1')
10341     }
10342     if( argv[0] != null ){
10343         id = argv[0]
10344         if( id == '$' ){
10345             //e = document.getElementById('GJE_RootNode');
10346         }else{
10347             e = document.getElementById(id)
10348         }
10349         if( e != null ){
10350             text.value += e.outerHTML
10351         }else{
10352             text.value += "Not found: " + id
10353         }
10354     }else{
10355         text.value += GJE_RootNode.outerHTML
10356         //text.value += e.innerHTML
10357     }
10358 }else
10359 if( cmd == 'destroy' ){
10360     text.value += GJFactory_Destroy()
10361 }else
10362 if( cmd == 'save' ){
10363     e = document.getElementById('GJFactory')
10364     Permanent.setItem('GJFactory-1',e.innerHTML)
10365     text.value += "-- Saved GJFactory"
10366 }else
10367 if( cmd == 'load' ){
10368     gjf = Permanent.getItem('GJFactory-1')
10369     e = document.getElementById('GJFactory')
10370     e.innerHTML = gjf
10371     // must restore EventListener
10372     text.value += "-- EventListener was not restored"
10373 }else
10374 if( cmd.charAt(0) == '.' ){
10375     argv0 = argv0.split(' ')
10376     text.value += GJE_NodeEdit(argv0)
10377 }else
10378 if( cmd == 'cont' ){
10379     bannerIsStopping = false
10380     GshMenuStop.innerHTML = "Stop"
10381 }else
10382 if( cmd == 'date' ){
10383     text.value += DateLong()
10384 }else
10385 if( cmd == 'echo' ){
10386     text.value += args
10387 }else
10388 if( cmd == 'fork' ){
10389     html_fork()
10390 }else
10391 if( cmd == 'last' ){
10392     text.value += MyHistory
10393     //h = document.createElement("span")
10394     //h.innerHTML = MyHistory
10395     //text.value += h.innerHTML
10396     //tx = MyHistory.replace("\n","")
10397     //text.value += tx.replace("<"+<br>","<n") + "xxxxx"+<br>yyyyy"
10398 }else
10399 if( cmd == 'ne' ){
10400     text.value += GJE_NodeEdit(argv)
10401 }else
10402 if( cmd == 'reload' ){
10403     location.reload()
10404 }else
10405 if( cmd == 'mem' ){
10406     text.value += GJC_Memory('GJC_Storage',args,text)
10407 }else
10408 if( cmd == 'stop' ){
10409     bannerIsStopping = true
10410     GshMenuStop.innerHTML = "Start"
10411 }else
10412 if( cmd == 'who' ){
10413     text.value += "SessionId="+GJC_SessionId+" "+document.URL
10414 }else
10415 if( cmd == 'wall' ){
10416     text.value += GJC_Memory('GJC_Wall','write',text)
10417 }else
10418 {
10419     text.value += "Commands: help | echo | date | last \n"
10420     + "      new | save | load | mem \n"
10421     + "      who | wall | fork | nife"
10422 }
10423 }
10424 }
10425 function GJC_Input(){
10426     if( this.value.endsWith("\n") ){ // remove NL added by textarea
10427         this.value = this.value.slice(0,this.value.length-1)
10428     }
10429 }
10430 }
10431 var GCJ_Id = null
10432 function GJC_Resize(){
10433     GCJ_Id.style.zIndex = 20000
10434     //GCJ_Id.style.width = window.innerWidth - 16
10435     GCJ_Id.style.width = '100%';
10436     GCJ_Id.style.height = 200;
10437     GCJ_Id.style.backgroundColor = "rgba(0,64,16,1.0) // blackboard color
10438     GCJ_Id.style.color = "rgba(255,255,255,1.0)"
10439 }
10440 function GJC_FocusIn(){
10441     this.spellCheck = false
10442     SuppressGJShell = true

```

```

10443 this.onkeydown = GJC_Keydown
10444 GJC_Resize()
10445 }
10446 function GJC_FocusOut(){
10447   SuppressGJShell = false
10448   this.removeEventListener('keydown',GJC_Keydown);
10449 }
10450 window.addEventListener('resize',GJC_Resize);
10451
10452 function GJC_OnStorage(e){
10453   //alert('Got Message')
10454   //GJC.value += "\n((ReceivedMessage))\n"
10455 }
10456 window.addEventListener('storage',GJC_OnStorage);
10457 //window.addEventListener('storage',()=>{alert('GotMessage')})
10458
10459 function GJC_Setup(gjcId){
10460   //gjcId.style.width = gsh.getBoundingClientRect().width
10461   gjcId.style.width = '100%';
10462   gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
10463   //gjcId.value += "Date: " + DateLong() + "\n"
10464   gjcId.value += Ps1
10465   gjcId.onfocus = GJC_FocusIn
10466   gjcId.addEventListener('input',GJC_Input);
10467   gjcId.addEventListener('focusout',GJC_FocusOut);
10468   GJC_Id = gjcId
10469 }
10470 function GJC_Clear(id){
10471 }
10472 function GJConsole_initConsole(){
10473   if( document.getElementById("GJC_0") != null ){
10474     GJC_Setup(GJC_0)
10475   }else{
10476     GJCl_Container.innerHTML = '<'
10477     " " + "textArea id='GJC_1' class='GJConsole'><'+'/textArea>";
10478     GJC_Setup(GJC_1)
10479     factory = document.createElement('span');
10480     gsh.appendChild(factory)
10481     GJE_RootNode = factory;
10482     GJE_CurElement = GJE_RootNode;
10483   }
10484 }
10485 var initGJCF = false;
10486 function GJConsole_initFactory(){
10487   if( !initGJCF ){ return; } initGJCF = true;
10488   GShell_initKeyCommands();
10489   GJConsole_initConsole();
10490 }
10491 //GJConsole_initFactory();
10492 // TODO: focus handling
10493</script>
10494<style>
10495 .GJ_StyleEditor {
10496   font-size:9pt !important;
10497   font-family:Courier New, monospace !important;
10498 }
10499</style>
10500
10501</details>
10502</span>
10503<!-- GJConsole END ) ----- -->
10504*!
10505
10506<span id="BlinderText">
10507<style id="BlinderTextStyle">
10508 #GJLinkView {
10509   xposition:absolute; z-index:5000;
10510   position:relative;
10511   display:block;
10512   left:8px;
10513   color:#fff;
10514   width:800px; height:300px; resize:both;
10515   margin:0px; padding:4px;
10516   background-color:rgba(200,200,200,0.5) !important;
10517 }
10518 .MsggText {
10519   width:578px !important;
10520   resize:both !important;
10521   color:#000 !important;
10522 }
10523 }
10524 .GjNote {
10525   font-family:Georgia !important;
10526   font-size:13pt !important;
10527   color:#22a !important;
10528 }
10529 .textField {
10530   display:inline;
10531   border:0.5px solid #444;
10532   border-radius:3px;
10533   color:#000; background-color:#fff;
10534   width:106pt; height:18pt;
10535   margin:2px;
10536   padding:2px;
10537   resize:none;
10538   vertical-align:middle;
10539   font-size:10pt; font-family:Courier New;
10540 }
10541 .textLabel {
10542   border:0px solid #000 !important;
10543   background-color:rgba(0,0,0,0);
10544 }
10545 .textURL {
10546   width:300pt !important;
10547   border:0px solid #000 !important;
10548   background-color:rgba(0,0,0,0);
10549 }
10550 .VisibleText {
10551 }
10552 .BlinderText {
10553   color:#000; background-color:#eee;
10554 }
10555 .JoinButton {
10556   font-family:Georgia !important;
10557   font-size:11pt;
10558   line-height:1.1;
10559   height:18pt;
10560   width:50pt;
10561   padding:3px !important;
10562   text-align:center !important;
10563   border-color:#aaa !important;
10564   border-radius:5px;
10565   color:#fff; background-color:#4a4 !important;
10566   vertical-align:middle !important;
10567 }
10568 .SendButton {
10569   vertical-align:top;
10570 }
10571 .ws0_log {
10572   font-size:10pt;
10573   color:#000 !important;
10574   line-height:1.0;
10575   background-color:rgba(255,255,255,0.7) !important;
10576   font-family:Courier New,monospace !important;
10577   width:99.38;
10578   white-space:pre;
10579 }
10580</style>
10581
10582<!-- Form autofill test
10583Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
10584<form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;>
10585<dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
10586-->
10587<details><summary>Form Auto. Filling</summary>
10588<style>
10589 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
10590   display:inline !important; font-size:10pt !important; padding:1px !important;
10591 }
10592</style>
10593<span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
10594 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;>
10595 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
10596 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
10597 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
10598 SessionId: <input id="xxsid" class="xxinput" name="SESSION_ID" type="text" size="80">
10599 <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
10600</span>
10601<script>
10602 function XXSetFormAction(){
10603   xxform.setAttribute('action',xxserv.value);
10604 }
10605 xxform.setAttribute('action',xxserv.value);
10606 xxserv.addEventListener('change',XXSetFormAction);
10607 //xxserv.value = location.href;
10608</script>
10609</details>
10610*!
10611
10612<details id="BlinderTextClass"><summary>BlinderText</summary>
10613<span class="gsh-src">
10614<span id="BlinderTextScript">
10615// https://w3c.github.io/uievents/#event-type-keydown
10616//
10617// 2020-09-21 class BlinderText - textArea element not to be readable
10618//

```

```

10620// BlinderText attributes
10621// bl_plainText - null
10622// bl_hideChecksum - [false]
10623// bl_showLength - [false]
10624// bl_visible - [false]
10625// data-bl_config - []
10626// - min. length
10627// - max. length
10628// - acceptable charset in generate text
10629//
10630function BlinderChecksum(text){
10631  plain = text.bl_plainText;
10632  return strCRC32(plain,plain.length).toFixed(0);
10633}
10634function BlinderKeydown(ev){
10635  pass = ev.target;
10636  if( ev.code == 'Enter' ){
10637    ev.preventDefault();
10638  }
10639  ev.stopPropagation();
10640}
10641function BlinderKeyUp1(ev){
10642  blind = ev.target;
10643  if( ev.code == 'Backspace' ){
10644    blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
10645  }else
10646  if( and(ev.code == 'KeyV', ev.ctrlKey) ){
10647    blind.bl_visible = !blind.bl_visible;
10648  }else
10649  if( and(ev.code == 'KeyL', ev.ctrlKey) ){
10650    blind.bl_showLength = !blind.bl_showLength;
10651  }else
10652  if( and(ev.code == 'KeyU', ev.ctrlKey) ){
10653    blind.bl_plainText = "";
10654  }else
10655  if( and(ev.code == 'KeyR', ev.ctrlKey) ){
10656    checksum = BlinderChecksum(blind);
10657    blind.bl_plainText = checksum; // toString(32);
10658  }else
10659  if( ev.code == 'Enter' ){
10660    ev.stopPropagation();
10661    ev.preventDefault();
10662    return;
10663  }else
10664  if( ev.key.length != 1 ){
10665    console.log('KeyUp: '+ev.code+' '+ev.key);
10666    return;
10667  }else{
10668    blind.bl_plainText += ev.key;
10669  }
10670
10671  leng = blind.bl_plainText.length;
10672  //console.log('KeyUp: '+ev.code+' '+blind.bl_plainText);
10673  checksum = BlinderChecksum(blind) & 10; // show last one digit only
10674
10675  visual = '';
10676  if( !blind.bl_hideChecksum || blind.bl_showLength ){
10677    visual += '|';
10678  }
10679  if( !blind.bl_hideChecksum ){
10680    visual += '#' + checksum.toString(10);
10681  }
10682  if( blind.bl_showLength ){
10683    visual += '/' + leng;
10684  }
10685  if( !blind.bl_hideChecksum || blind.bl_showLength ){
10686    visual += '|';
10687  }
10688  if( blind.bl_visible ){
10689    visual += blind.bl_plainText;
10690  }else{
10691    visual += '*'.repeat(leng);
10692  }
10693  blind.value = visual;
10694}
10695function BlinderKeyUp(ev){
10696  BlinderKeyUp1(ev);
10697  ev.stopPropagation();
10698}
10699// https://w3c.github.io/uievents/#keyboardevent
10700// https://w3c.github.io/uievents/#uievent
10701// https://dom.spec.whatwg.org/#event
10702function BlinderTextEvent(){
10703  ev = event;
10704  blind = ev.target;
10705  console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
10706  if( ev.type == 'keyup' ){
10707    BlinderKeyUp(ev);
10708  }else
10709  if( ev.type == 'keydown' ){
10710    BlinderKeydown(ev);
10711  }else{
10712    console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
10713  }
10714}
10715//< textarea hidden id="BlinderTextClassDef" class="textField"
10716// onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
10717// spellcheck="false"></textarea>
10718//< textarea hidden id="sj_pass1"
10719// class="textField BlinderText"
10720// placeholder="PassWord"
10721// onkeydown="BlinderTextEvent()"
10722// onkeyup="BlinderTextEvent()"
10723// spellcheck="false"></textarea>
10724function SetupBlinderText(parent,txa,phold){
10725  if( txa == null ){
10726    txa = document.createElement('textarea');
10727    //txa.id = id;
10728  }
10729  txa.setAttribute('class','textField BlinderText');
10730  txa.setAttribute('placeholder',phold);
10731  txa.setAttribute('onkeydown','BlinderTextEvent()');
10732  txa.setAttribute('onkeyup','BlinderTextEvent()');
10733  txa.setAttribute('spellcheck','false');
10734  //txa.setAttribute('bl_plainText','false');
10735  txa.bl_plainText = '';
10736  //parent.appendChild(txa);
10737}
10738function DestroyBlinderText(txa){
10739  txa.removeAttribute('class');
10740  txa.removeAttribute('placeholder');
10741  txa.removeAttribute('onkeydown');
10742  txa.removeAttribute('onkeyup');
10743  txa.removeAttribute('spellcheck');
10744  txa.bl_plainText = '';
10745}
10746//
10747// visible textarea like Username
10748//
10749function VisibleTextEvent(){
10750  if( event.code == 'Enter' ){
10751    if( event.target.NoEnter ){
10752      event.preventDefault();
10753    }
10754  }
10755  event.stopPropagation();
10756}
10757function SetupVisibleText(parent,txa,phold){
10758  if( false ){
10759    txa.setAttribute('class','textField VisibleText');
10760  }else{
10761    newclass = txa.getAttribute('class');
10762    if( and(newclass != null, newclass != '') ){
10763      newclass += ' ';
10764    }
10765    newclass += 'VisibleText';
10766    txa.setAttribute('class',newclass);
10767  }
10768  //console.log('SetupVisibleText class='+txa.class);
10769  txa.setAttribute('placeholder',phold);
10770  txa.setAttribute('onkeydown','VisibleTextEvent()');
10771  txa.setAttribute('onkeyup','VisibleTextEvent()');
10772  txa.setAttribute('spellcheck','false');
10773  cols = txa.getAttribute('cols');
10774  if( cols != null ){
10775    txa.style.width = '580px';
10776    //console.log('VisualText'+txa.id+' cols='+cols)
10777  }else{
10778    //console.log('VisualText'+txa.id+' NO cols')
10779  }
10780  rows = txa.getAttribute('rows');
10781  if( rows != null ){
10782    txa.style.height = '30px';
10783    txa.style.resize = 'both';
10784    txa.NoEnter = false;
10785  }else{
10786    txa.NoEnter = true;
10787  }
10788}
10789function DestroyVisibleText(txa){
10790  txa.removeAttribute('class');
10791  txa.removeAttribute('placeholder');
10792  txa.removeAttribute('onkeydown');
10793  txa.removeAttribute('onkeyup');
10794  txa.removeAttribute('spellcheck');
10795  cols = txa.removeAttribute('cols');
10796}

```

```

1079 </span>
1079 <script>
1079 js = document.getElementById('BlinderTextScript');
1080 eval(js.innerHTML);
1080 //js.outerHTML = ""
1080 </script>
1080
1080 </span><!-- end of class="gsh-src" -->
1080 </details>
1080 </span>
1080 */
1080 /*
1081 <script id="GJLinkScript">
1081 function gjkey_hash(text){
1081 return strCRC32(text,text.length) % 0x10000;
1081 }
1081 function gj_addlog(e,msg){
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 tstamp = ['+now+'];
1081 e.value += tstamp + msg;
1081 e.scrollTop = e.scrollHeight;
1081 }
1081 function gj_addlog_cl(msg){
1081 ws0_log.value += (console.log) ' + msg + '\n';
1081 }
1081 var GJ_Channel = null;
1081 var GJ_Log = null;
1081 var gjx; // the global variable
1081 function GJ_Join(){
1081 target = gj_join;
1081 if( target.value == 'Leave' ){
1081 GJ_Channel.close();
1081 GJ_Channel = null;
1081 target.value = 'Join';
1081 return;
1081 }
1081 var ws0;
1081 var ws0_log;
1081 sav_console_log = console.error
1081 console.error = gj_addlog_cl
1081 ws0 = new WebSocket(gj_serv.innerHTML);
1081 console.error = sav_console_log
1081
1081 GJ_Channel = ws0;
1081 ws0_log = document.getElementById('ws0_log');
1081 GJ_Log = ws0_log;
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
1081 cst = wsstats[ws0.readyState];
1081 gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
1081
1081 ws0.addEventListener('error', function(event){
1081 gj_addlog(ws0_log,'stat error : transport error?\n');
1081 });
1081 ws0.addEventListener('open', function(event){
1081 //console.log('#'+GJLinkView.id+'.zIndex='+GJLinkView.style.zIndex);
1081 datel = new Date().getTime();
1081 date2 = (date1 / 1000).toFixed(3);
1081 seed = date1.toString(16);
1081 // user name and key
1081 user = document.getElementById('gj_user').value;
1081 if( user.length == 0 ){
1081 gj_user.value = 'nemo';
1081 user = 'nemo';
1081 }
1081 key1 = document.getElementById('gj_ukey').bl_plainText;
1081 ukey = gjkey_hash(seed+user+key1).toString(16);
1081 // session name and key
1081 chan = document.getElementById('gj_chan').value;
1081 if( chan.length == 0 ){
1081 gj_chan.value = 'main';
1081 chan = 'main';
1081 }
1081 key2 = document.getElementById('gj_ckey').bl_plainText;
1081 ckey = gjkey_hash(seed+chan+key2).toString(16);
1081 msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + ' ' + ckey;
1081 gj_addlog(ws0_log,'send '+msg+'\n');
1081 ws0.send(msg);
1081
1081 target.value = 'Leave';
1081 //console.log(['+date2+'] + '#'+target.id+' '+target.value+'\n');
1081 //gj_addlog(ws0_log,'label '+target.value+'\n');
1081 });
1081 ws0.addEventListener('message', function(event){
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 msg = event.data;
1081 if( false ){
1081 gj_addlog(ws0_log,'recv '+msg+'\n');
1081 }
1081 argv = msg.split(' ');
1081 tstamp = argv[0];
1081 argv.shift();
1081 if( argv[0] == 'reload' ){
1081 location.reload()
1081 }
1081 gjcmd = argv[0];
1081 otstamp = '';
1081 if( gjcmd == 'CAST' ){ // from reflector
1081 otstamp = argv[0];
1081 argv.shift(); // original time stamp
1081 ofrom = argv[0];
1081 argv.shift(); // original from
1081 }
1081 argv.shift(); // command
1081 from = argv[0];
1081 argv.shift(); // from/to
1081 cmd1 = argv[0];
1081 argv.shift(); // xxxx command
1081
1081 if( false ){
1081 gj_addlog(ws0_log,'--'
1081 + ' tstamp='+tstamp
1081 + ' gjcmd='+gjcmd
1081 + ' ,from='+from
1081 + ' ,cmd=[ '+cmd1+' ]'
1081 + ' '+argv+'\n');
1081 }
1081 if( cmd1 == 'auth' ){
1081 // doing authorization required
1081 }
1081 if( cmd1 == 'echo' ){
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 msg = now + ' +RESP '+argv.join(' ');
1081 gj_addlog(ws0_log,'send '+msg+'\n');
1081 ws0.send(msg);
1081 }
1081 if( cmd1 == 'eval' ){
1081 argv.shift();
1081 js = argv.join(' ');
1081 ret = eval(js); //----- eval()
1081 gj_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 msg = now + ' ' + 'RESP ' + ret;
1081 ws0.send(msg);
1081 gj_addlog(ws0_log,'send '+msg+'\n')
1081 }
1081 if( cmd1 == 'DRAW' ){
1081 if( false ){
1081 gj_addlog(ws0_log,'DRAW '+argv[0]+'\n')
1081 }
1081 Pointillism_RemoteDraw(argv[0]);
1081 }
1081 });
1081 ws0.addEventListener('close', function(event){
1081 if( GJ_Channel == null ){
1081 gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
1081 return;
1081 }
1081 GJ_Channel.close();
1081 GJ_Channel = null;
1081 target.value = 'Join';
1081 gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
1081 });
1081
1081 function GJ_BcastMessageUserPass(user,chan,msgbody){
1081 now = (new Date().getTime() / 1000).toFixed(3);
1081 msg = now + ' BCAST '+user+'|'+chan+' '+msgbody;
1081 if( false ){
1081 gj_addlog(GJ_Log,'send '+msg+'\n');
1081 }
1081 GJ_Channel.send(msg);
1081 }
1081 function GJ_BcastMessage(msgbody){
1081 if( GJ_Channel == null ){
1081 gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
1081 return;
1081 }
1081 //target = event.target;
1081 user = document.getElementById('gj_user').value;

```

```

10974 chan = document.getElementById('gj_chan').value;
10975 GJ_BcastMessageUserPass(user,chan,msgbody);
10976 }
10977 function GJ_SendMessageUserPass(user,chan,msgbody){
10978 now = (new Date()).getTime() / 1000;.toFixed(3);
10979 msg = now + " ISAI "+user+" |"+chan+" "+ msgbody;
10980 gj_addlog(GJ_Log, send +msg+"\n");
10981 GJ_Channel.send(msg);
10982 }
10983 function GJ_SendMessage(msgbody){
10984 if( GJ_Channel == null ){
10985 gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10986 return;
10987 }
10988 //target = event.target;
10989 user = document.getElementById('gj_user').value;
10990 chan = document.getElementById('gj_chan').value;
10991 GJ_SendMessageUserPass(user,chan,msgbody);
10992 }
10993 function GJ_Send(){
10994 msgbody = gj_sendText.value;
10995 GJ_SendMessage(msgbody);
10996 }
10997 </script>
10998 <!-- ----- GJLINK ----- -->
11000 <!-- - User can subscribe to a channel
11001 - A channel will be broadcasted
11002 - A channel can be a pattern (regular expression)
11003 - User is like From:(me) and channel is like To: or Recipient:
11004 - like VARSUB
11005 - watch message with SENDME, WATCH, CATCH, HEAR, or so
11006 - routing with path expression or name pattern (with routing with DNS like system)
11007 -->
11008 */
11009
11010
11011 <span id="GJLinkGolang">
11012 // <details id="gshWebsocket"><summary>Golang / JavaScript Link</summary>
11013 // <span class="gsh-src"><!-- { -->
11014 // 2020-0920 created
11015 // <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
11016 // <a href="https://golang.org/x/net/websocket">WS</a>
11017 // INSTALL: go get golang.org/x/net/websocket
11018 // INSTALL: sudo (apt,yum) install git (if git is not instilled yet)
11019 // import "golang.org/net/websocket"
11020 const gshws_origin = "http://localhost:9999"
11021 const gshws_server = "localhost:9999"
11022 const gshws_port = 9999
11023 const gshws_path = "gjlink1"
11024 const gshws_url = "ws://" + gshws_server + "/" + gshws_path
11025 const GSHWS_MSGSIZE = (8*1024)
11026 func fmtstring(fmts string, params ...interface{})(string){
11027 return fmt.Sprintf(fmts,params...)
11028 }
11029 func GSHWS_MARK(what string)(string){
11030 now := time.Now()
11031 us := fmtstring("%06d",now.Nanosecond() / 1000)
11032 mark := "-"
11033 if( !AtConsoleLineTop ){
11034 mark += "\n"
11035 AtConsoleLineTop = true
11036 }
11037 mark += "[" + now.Format(time.Stamp) + "." + us + "]" -GJ- + what + " : "
11038 return mark
11039 }
11040 func gchk(what string,err error){
11041 if( err != nil ){
11042 panic(GSHWS_MARK(what)+err.Error())
11043 }
11044 }
11045 func glog(what string, fmts string, params ...interface{}){
11046 fmt.Print(GSHWS_MARK(what))
11047 fmt.Printf(fmts+"\n",params...)
11048 }
11049
11050 var WSV = []*websocket.Conn{}
11051 func jsend(argv []string){
11052 if len(argv) <= 1 {
11053 fmt.Printf("--!j %v [-m] command arguments\n",argv[0])
11054 return
11055 }
11056 argv = argv[1:]
11057 if( len(WSV) == 0 ){
11058 fmt.Printf("--Ej-- No link now\n")
11059 return
11060 }
11061 if( 1 < len(WSV) ){
11062 fmt.Printf("--!j-- multiple links (%v)\n",len(WSV))
11063 }
11064 multicast := false // should be filtered with regexp
11065 if( 0 < len(argv) && argv[0] == "-m" ){
11066 multicast = true
11067 argv = argv[1:]
11068 }
11069 args := strings.Join(argv, " ")
11070
11071 now := time.Now()
11072 msec := now.UnixNano() / 1000000;
11073 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
11074 msg := fmtstring("%v SEND gshell* %v",tstamp,args)
11075
11076 if( multicast ){
11077 for i,ws := range WSV {
11078 wn,werr := ws.Write([]byte(msg))
11079 if( werr != nil ){
11080 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
11081 }
11082 glog("SQ",fmtstring("(%v) %v",wn,msg))
11083 }
11084 }else{
11085 i := 0
11086 ws := WSV[i]
11087 wn,werr := ws.Write([]byte(msg))
11088 if( werr != nil ){
11089 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
11090 }
11091 glog("SQ",fmtstring("(%v) %v",wn,msg))
11092 }
11093 }
11094
11095 func ws_broadcast(msg string)(wn int,werr error){
11096 for i,ws := range WSV {
11097 wn,werr := ws.Write([]byte(msg))
11098 if( werr != nil ){
11099 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
11100 }
11101 glog("SQ",fmtstring("(%v) %v",wn,msg))
11102 }
11103 return wn,werr;
11104 }
11105
11106 func servl(ws *websocket.Conn) {
11107 WSV = append(WSV,ws)
11108 //fmt.Print("\n")
11109 glog("CO", "accepted connections[%v]",len(WSV))
11110 //remoteAddr := ws.RemoteAddr
11111 //fmt.Printf("-- accepted %v\n",remoteAddr)
11112 //fmt.Printf("-- accepted %v\n",ws.Config().Header)
11113 //fmt.Printf("-- accepted %v // %v\n",ws,servl)
11114
11115 var reqb = make([]byte,GSHWS_MSGSIZE)
11116 for {
11117 rn, rerr := ws.Read(reqb)
11118 if( rerr != nil || rn < 0 ){
11119 glog("SQ",fmtstring("(%v,%v)",rn,rerr))
11120 break
11121 }
11122 req := string(reqb[0:rn])
11123 glog("SQ",fmtstring("(%v) %v",rn,req))
11124
11125 margv := strings.Split(req, " ");
11126 margv = margv[1:]
11127 if( 0 < len(margv) ){
11128 if( margv[0] == "RESP" ){
11129 // should forward to the destination
11130 continue;
11131 }
11132 }
11133 now := time.Now()
11134 msec := now.UnixNano() / 1000000;
11135 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
11136 res := fmtstring("%v "+CAST+" %v",tstamp,req)
11137
11138 wn := 0;
11139 werr := error(nil);
11140 if( 0 < len(margv) && margv[0] == "BCAST" ){
11141 wn, werr = ws_broadcast(res);
11142 }else{
11143 wn, werr = ws.Write([]byte(res))
11144 }
11145 gchk("SE",werr)
11146 glog("SR",fmtstring("(%v) %v",wn,string(res)))
11147 }
11148 glog("SF","WS response finish")
11149
11150 wsv := []*websocket.Conn{}

```

```

11151 wsx := 0
11152 for i,v := range WSV {
11153     if v != ws ){
11154         wsx = i
11155         wsv = append(wsv,v)
11156     }
11157 }
11158 WSV = wsv
11159 //glog("CO", "closed %v",ws)
11160 glog("CO", "closed connection [%v/%v]",wsx+1,len(WSV)+1)
11161 ws.Close()
11162 }
11163 // url := [scheme://host[:port][[/path]
11164 func decomp_url(url string){
11165 }
11166 func full_wsURL(){
11167 }
11168 func gj_server(argv []string) {
11169     gjserv := gshws_url
11170     gjport := gshws_server
11171     gjpath := gshws_path
11172     gjscheme := "ws"
11173 }
11174 //cmd := argv[0]
11175 argv = argv[1:]
11176 if ( 1 <= len(argv) ){
11177     serv := argv[0]
11178     if ( 0 < strings.Index(serv, "://") ){
11179         schemev := strings.Split(serv, "://")
11180         gjscheme = schemev[0]
11181         serv = schemev[1]
11182     }
11183     if ( 0 < strings.Index(serv, "/") ){
11184         pathv := strings.Split(serv, "/")
11185         serv = pathv[0]
11186         gjpath = pathv[1]
11187     }
11188     servv := strings.Split(serv, ":")
11189     host := "localhost"
11190     port := 9999
11191     if ( servv[0] != "" ){
11192         host = servv[0]
11193     }
11194     if ( len(servv) == 2 ){
11195         fmt.Sscanf(servv[1], "%d", &port)
11196     }
11197     //glog("LC", "hostport=%v (%v : %v)",servv,host,port)
11198     gjport = fmt.Sprintf("%v:%v",host,port)
11199     gjserv = gjscheme + "://" + gjport + "/" + gjpath
11200 }
11201 glog("LS",fmtstring("listening at %v",gjserv))
11202 http.Handle("/"+gjpath,websocket.Handler(serv))
11203 err := error(nil)
11204 if ( gjscheme == "wss" ){
11205     // https://golang.org/pkg/net/http/#ListenAndServeTLS
11206     //err = http.ListenAndServeTLS(gjport,nil)
11207 }else{
11208     err = http.ListenAndServe(gjport,nil)
11209 }
11210 gchk("LE",err)
11211 }
11212
11213 func gj_client(argv []string) {
11214     glog("CS",fmtstring("connecting to %v",gshws_url))
11215     ws, err := websocket.Dial(gshws_url, "", gshws_origin)
11216     gchk("C",err)
11217 }
11218 var resb = make([]byte, GSHWS_MSGSIZE)
11219 for qi := 0; qi < 3; qi++ {
11220     req := fmtstring("Hello, GShell! (%v)",qi)
11221     wn, werr := ws.Write([]byte(req))
11222     glog("QM",fmtstring("(%v) %v",wn,req))
11223     gchk("QE",werr)
11224     rn, rerr := ws.Read(resb)
11225     gchk("RE",rerr)
11226     glog("RM",fmtstring("(%v) %v",rn,string(resb)))
11227 }
11228 glog("CP", "WS request finish")
11229 }
11230 //</span><!-- end of class="gsh-src" -->
11231 //</details></span>
11232
11233 /*
11234 <details id="GJLink_Section"><summary>GJ Link</summary>
11235 <span id="GJLinkView" class="GJLinkView">
11236 <p>
11237 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
11238 </p>
11239
11240 <span id="GJLink_1">
11241 <div id="GJLink_ServerSet"></div>
11242
11243 <div>
11244 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
11245 <span id="GJLink_Account"></span>
11246 </div>
11247
11248 <div>
11249 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
11250 <span id="GJLink_SendArea"></span>
11251 </div>
11252
11253 <div id="ws0_log_container"></div>
11254 </span>
11255 </span>
11256
11257 <script>
11258 function setupGJLinkArea(){
11259     GJLink_ServerSet.innerHTML = '<'+<span id="gj_serv_label">
11260     + <'+<span id="gj_serv" class="textField textURL" contenteditable><'+</span>';
11261
11262     GJLink_Account.innerHTML = '<'+<textarea id="gj_user" class="textField"><'+</textarea>
11263     + <'+<textarea id="gj_ukey" class="textField"><'+</textarea>
11264     + <'+<textarea id="gj_chan" class="textField"><'+</textarea>
11265     + <'+<textarea id="gj_ckey" class="textField"><'+</textarea>';
11266
11267     GJLink_SendArea.innerHTML =
11268     '<'+<textarea id="gj_sendText" class="textField MsgText" cols=60 rows=2><'+</textarea>';
11269
11270     ws0_log_container.innerHTML = '<'+<textarea id="ws0_log" class="ws0_log"
11271     +<'+<div id="ws0_log" style="border: 1px solid black; height: 100px; width: 100%;></div>';
11272 }
11273
11274 function clearGJLinkArea(){
11275     GJLink_ServerSet.innerHTML = "";
11276     GJLink_Account.innerHTML = "";
11277     GJLink_SendArea.innerHTML = "";
11278     ws0_log_container.innerHTML = "";
11279 }
11280 </script>
11281
11282 <script>
11283 function SetupGJLink(){
11284     setupGJLinkArea();
11285     SetupVisibleText(GJLink_l_gj_serv, 'GJLinkServ');
11286     SetupVisibleText(GJLink_l_gj_user, 'UserName');
11287     SetupBlinderText(GJLink_l_gj_ukey, 'UserKey');
11288     SetupVisibleText(GJLink_l_gj_chan, 'ChannelName');
11289     SetupBlinderText(GJLink_l_gj_ckey, 'ChannelKey');
11290     SetupVisibleText(GJLink_l_gj_sendText, 'Message');
11291     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1';
11292 }
11293
11294 function GJLink_init(){
11295     SetupGJLink();
11296 }
11297
11298 function iselem(eid){
11299     return document.getElementById(eid);
11300 }
11301
11302 function DestroyGJLink1(){
11303     clearGJLinkArea();
11304     if ( !iselem('gj_user') ){
11305         return;
11306     }
11307     if ( gj_serv_label.parentNode != gj_user ){
11308         return;
11309     }
11310     if ( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
11311     if ( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
11312     if ( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
11313     if ( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
11314     if ( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
11315     if ( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
11316     if ( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
11317     if ( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
11318 }
11319
11320 DestroyGJLink1 = DestroyGJLink1;
11321 </script>
11322 </details>
11323 */
11324
11325 <style>
11326 .GJDigest {
11327     display:none;
11328 }
11329 </style>
11330 <script id="HtmlCodeview-script">

```



```

11328 function showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign){
11329   txa = document.createElement('textArea');
11330   txa.id = otxa.id;
11331   txa.setAttribute('class','HtmlCodeviewText');
11332   otxa.parentNode.replaceChild(txa,otxa);
11333   txa.setAttribute('spellcheck','false');
11334   //txa.value = code.innerHTML;
11335   //txa.innerHTML = code.innerHTML;
11336   txa.innerHTML = prefix + code.outerHTML + postfix;
11337   if( sign ){
11338     text = txa.value;
11339     tlen = txa.value.length;
11340     digest = strCRC32(text,tlen) + ' ' + tlen
11341     // + code.id + ' (' + DateShort() + ')';
11342     //alert('digest: '+digest);
11343     console.log('digest: '+digest);
11344     txa.innerHTML += '</><span class="GJDigest">'+digest+'</></span>\n';
11345   }
11346   txa.style.display = "block";
11347   txa.style.width = "100%";
11348   txa.style.height = "300px";
11349 }
11350 function showHtmlCodeX(otxa,code,prefix,postfix,sign){
11351   if( event.target.value == 'ShowCode' ){
11352     showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign);
11353     event.target.value = 'HideCode';
11354   }else{
11355     otxa.style.display = "none";
11356     event.target.value = 'ShowCode';
11357   }
11358 }
11359 function showNodeAsHtmlSource(otxa,code){
11360   showNodeAsHtmlSourceX(otxa,code,'','');
11361 }
11362 function showHtmlCode(otxa,code){
11363   if( event.target.value == 'ShowCode' ){
11364     showNodeAsHtmlSource(otxa,code);
11365     event.target.value = 'HideCode';
11366   }else{
11367     otxa.style.display = "none";
11368     event.target.value = 'ShowCode';
11369   }
11370 }
11371 </script>
11372 <style id="HtmlCodeview-style">
11373   .HtmlCodeviewText {
11374     font-size:10pt;
11375     font-family:Courier New;
11376     white-space:pre;
11377   }
11378   .HtmlCodeViewButton {
11379     padding:2pt 1important;
11380     line-height:1.1 1important;
11381     border:2px inset #bbb 1important;
11382     font-size:11pt 1important;
11383     font-weight:normal 1important;
11384     font-family:Georgia 1important;
11385     border-radius:3px 1important;
11386     color:#ddd; background-color:#228 1important;
11387   }
11388 </style>
11389 /*
11390 <details><summary>Live HTML Snapshot</summary>
11391 <span id="LiveHTML">
11392 <!-- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
11393 <div class="GshMenu">
11394 <span class="GshMenu" onclick="html_edit();">Edit</span>
11395 <span class="GshMenu" onclick="html_save();">Save</span>
11396 <span class="GshMenu" onclick="html_load();">Load</span>
11397 <span class="GshMenu" onclick="html_ver0();">Vers</span>
11398 </div>
11399 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()" >
11400 <span id="LiveHTML_Codeview"></span>
11401 </div>
11402 <script id="LiveHTMLScript">
11403 function showLiveHTMLCode(){
11404   showHtmlCode(LiveHTML_Codeview,LiveHTML);
11405 }
11406 var editable = false;
11407 var savSuppressGJShell = false;
11408 function ToggleEditMode(){
11409   editable = !editable;
11410   If( !editable ){
11411     savSuppressGJShell = SuppressGJShell;
11412     SuppressGJShell = true;
11413     gsh.setAttribute('contenteditable','true');
11414     GshMenuEdit.innerHTML = 'Lock';
11415     GshMenuEdit.style.color = 'rgba(255,0,0,1)';
11416     GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
11417   }else{
11418     SuppressGJShell = savSuppressGJShell;
11419     gsh.setAttribute('contenteditable','false');
11420     GshMenuEdit.innerHTML = 'Edit';
11421     GshMenuEdit.style.color = 'rgba(16,160,16,1)';
11422     GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
11423   }
11424 }
11425 function html_edit(){
11426   ToggleEditMode();
11427 }
11428 // Live HTML (DOM) Snapshot onto browser's localStorage
11429 // 2020-0923 SatoxITS
11430 var htRoot = gsh; // -- Element-ID, should be selectable
11431 const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
11432 // -- should be a [map] of URL
11433 // -- should be with CSSOM as inline script
11434 const htVersionTag = 'VersionTag'; // VesionTag Belmont-ID in the HTML (in DOM)
11435 function showVersion(note,w,v,u,t){
11436   w.alert(note+ ' + v + '\n'
11437     + '-- URL: ' + v + '\n'
11438     + '-- Time: ' + t + ' (' + DateLong(t*1000) + ')')
11439 }
11440 function html_save(){
11441   u = document.URL;
11442   t = new Date().getTime() / 1000;
11443   v = '<+>span id="'+htVersionTag+'"' data-url="'+u+'" data-time="'+t+'";'
11444   v += '<+>/span>';
11445   h += v + htRoot.outerHTML;
11446   localStorage.setItem(snappedHTML,h);
11447   showVersion('Saved',window,v,u,t);
11448 }
11449 function html_load(){
11450   h = localStorage.getItem(snappedHTML);
11451   if( h == null ){
11452     alert('No snapshot taken yet');
11453     return;
11454   }
11455   w = window.open('','','');
11456   d = w.document;
11457   d.write(h);
11458   w.focus();
11459   html_ver1("Loaded",w,d);
11460 }
11461 function html_ver1(note,w,d){
11462   if( ( v = d.getElementById(htVersionTag) ) != null ){
11463     h = v.outerHTML;
11464     u = v.getAttribute('data-url');
11465     t = v.getAttribute('data-time');
11466   }else{
11467     h = 'No version info. in the page';
11468     u = '';
11469     t = 0;
11470   }
11471   showVersion(note,w,v,u,t);
11472 }
11473 function html_ver0(){
11474   html_ver1("Version",window,document);
11475 }
11476 </script>
11477 <!-- LiveHTML -->
11478 </span>
11479 </details>
11480 /*
11481 <details><summary>Event sharing</summary>
11482 <span id="EventSharingCodeSpan">
11483 <!-- Event sharing // 2020-0925 SatoxITS { -->
11484 <div id="iftestTemplate" class="iftest" hidden="">
11485 <style>
11486 <span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
11487   function docadd(txt){
11488     document.body.append(txt);
11489     window.scrollTo(0,100000);
11490   }
11491   function frameClick(){
11492     xy = (x+'event.x + ' + y+'event.y');
11493     //docadd('Got Click on #' + event.target.id + ' +xy+ '\n');
11494     docadd('Got Click on #' + Fid.value + ' +xy+ '\n');
11495     window.scrollTo(0,100000);

```

```

11505 window.parent.postMessage('OnClick: ' + xy, '*');
11506 }
11507 function frameMouseMove(){
11508     if( false ){
11509         document.body.append('MouseMove on #' + event.target.id + ' '
11510             + 'x=' + event.x + ' y=' + event.y + '\n');
11511         peerWin = window.frames[frame1];
11512         document.body.append('Send to peer #' + peerWin + ' ' + '\n');
11513         window.scrollTo(0,100000);
11514         peerWin.postMessage('Hi!', '*');
11515     }
11516 }
11517 function frameKeyDown(){
11518     msg = 'Got Keydown: #' + fid.value + ' (' + event.code + ')';
11519     docadd(msg + '\n');
11520     window.parent.postMessage(msg, '*');
11521 }
11522 function frameOnMessage(){
11523     docadd('Message ' + event.data + '\n');
11524     window.scrollTo(0,100000);
11525 }
11526 if( document.getElementById('Fid') ){
11527     frameBody.id = Fid.value;
11528     h =
11529     h += '<' + 'style*' + ' ';
11530     h += 'font-size:10pt;white-space:pre-wrap;';
11531     h += 'font-family:Courier New;';
11532     h += '<' + '*/style*' + ' ';
11533     h += 'I am ' + Fid.value + '\n';
11534     document.write(h);
11535     window.addEventListener('click', frameClick);
11536     window.addEventListener('keydown', frameKeyDown);
11537     window.addEventListener('message', frameOnMessage);
11538     window.addEventListener('mousemove', frameMouseMove);
11539     window.parent.postMessage('Hi parent, I am ' + Fid.value, '*');
11540 }
11541 </script></span></div>
11542 </div>
11543 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px}</style>
11544 <h2>Inter-window communicaiton</h2>
11545 <note>
11546 frame0 >>> frame1 and frame2<br>
11547 frame1 >>> frame0 and frame2<br>
11548 frame2 >>> frame0 and frame1<br>
11549 </note>
11550 <div id="iframe-test">
11551 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
11552 <iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
11553 <iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
11554 <iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
11555 </div>
11556 <script id="if0-test-script">
11557 function InterFrameComm_init(){
11558     setupFrames0();
11559     setupFrames1();
11560     setupFrames2();
11561 }
11562 function setFrameSrcdoc(dst,src){
11563     if( true ){
11564         dst.contentWindow.document.write(src);
11565         // this makes browser wait close, and crash if accumulated !?
11566         // so it should be closed after write
11567         dst.contentWindow.document.close();
11568     }else{
11569         // to be erased before source dump
11570         // but should be set for live snapshot
11571         dst.srcdoc = src;
11572     }
11573 }
11574 function setupFrames0(){
11575     body = iframe0.contentWindow.document.body;
11576     iframe0.style.width = "755px"
11577     //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
11578     window.addEventListener('message', messageFromChild);
11579 }
11580 if0 = '';
11581 if0 += '<' + 'pre style="font-family:Courier New;">';
11582 if0 += '<input id="Fid" value="iframe0">';
11583 if0 += iftestTemplate.innerHTML;
11584 setFrameSrcdoc(iframe0,if0);
11585 }
11586 function clickOnChild(){
11587     console.log('clickOn #' + this.id);
11588 }
11589 function moveOnChild(){
11590     console.log('moveOn #' + this.id);
11591 }
11592 iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
11593 iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
11594 }
11595 function setupFrames1(){
11596     if1 = '<input id="Fid" value="iframe1">';
11597     if1 += iftestTemplate.innerHTML;
11598     setFrameSrcdoc(iframe1,if1);
11599     //iframe1.name = 'iframe1'; // this seems break contentWindow
11600 }
11601 if2 = '<input id="Fid" value="iframe2">';
11602 if2 += iftestTemplate.innerHTML;
11603 setFrameSrcdoc(iframe2,if2);
11604 }
11605 iframe1.addEventListener('message', messageFromChild);
11606 //iframe1.addEventListener('mouseover', moveOnChild);
11607 iframe2.addEventListener('message', messageFromChild);
11608 //iframe2.addEventListener('mouseover', moveOnChild);
11609 iframe1.contentWindow.postMessage([parent] Hi iframe1 -- from parent.', '*');
11610 //iframe1.contentWindow.postMessage('Your peer is ' + iframe2.contentWindow, '*');
11611 //iframe2.contentWindow.postMessage([parent] Hi iframe2 -- from parent.', '*');
11612 //iframe2.contentWindow.postMessage('Your peer is ' + iframe1.contentWindow, '*');
11613 }
11614 function messageFromChild(){
11615     from = null;
11616     forw = null;
11617     if( event.source == iframe0.contentWindow ){
11618         from = [iframe0];
11619         forw = 'iframe12';
11620     }else
11621     if( event.source == iframe1.contentWindow ){
11622         from = [iframe1];
11623         forw = 'iframe2';
11624     }else
11625     if( event.source == iframe2.contentWindow ){
11626         from = [iframe2];
11627         forw = 'iframe1';
11628     }else
11629     {
11630         iframeHost.innerHTML += 'Message [unknown] '
11631         + ' orig=' + event.origin
11632         + ' data=' + event.data
11633         //+ ' from=' + event.source
11634         ;
11635     }
11636     msglogl = from + event.data + ' -- '
11637     + ' from=' + event.source
11638     + ' orig=' + event.origin
11639     + ' name=' + event.source.name
11640     //+ ' port=' + event.ports
11641     //+ ' euid=' + event.lastEventId
11642     + '\n'
11643     ;
11644     if( true ){
11645         if( forw == 'iframe1' || forw == 'iframe12' ){
11646             iframe1.contentWindow.postMessage(from+event.data);
11647         }
11648         if( forw == 'iframe2' || forw == 'iframe12' ){
11649             iframe2.contentWindow.postMessage(from+event.data);
11650         }
11651     }
11652     txtadd0(msglogl);
11653 }
11654 function txtadd0(txt){
11655     iframe0.contentWindow.document.body.append(txt);
11656     iframe0.contentWindow.scrollTo(0,100000);
11657 }
11658 }
11659 function es_ShowSelf(){
11660     IFrame1.setAttribute('src', document.URL);
11661     IFrame2.setAttribute('src', document.URL);
11662 }
11663 </script>
11664 </div>
11665 <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
11666 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
11667 <span id="EventSharingCodeView"></span>
11668 <script id="EventSharingScript">
11669 function es_showHtmlCode(){
11670     showHtmlCode(EventSharingCodeview, EventSharingCodeSpan);
11671 }
11672 DestroyEventSharingCodeview = function(){
11673     //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
11674     EventSharingCodeview.innerHTML = "";
11675     iframe0.style = "";
11676     //iframe0.srcdoc = "erased";
11677     //iframe1.srcdoc = "erased";
11678     //iframe2.srcdoc = "erased";
11679 }
11680 </script>
11681 <!-- EventSharing -->

```

```

1166</span>
1166</details>
1166/*
1166
1166
1166
1166</-- ----- "GShell Inside" Modification { -->
1166<script id="script-gshell-inside">
1166var notices = 0;
1166function noticeGShellInside(){
1166  var = "";
1166  if( ver = document.getElementById('GshVersion') ){
1166    ver = ver.innerHTML;
1166  }
1166  console.log('GJShell Inside ("~"//'+ver);
1166  notices += 1;
1166  if( 2 <= notices ){
1166    document.removeEventListener('mousemove',noticeGShellInside);
1166  }
1166}
1166document.addEventListener('mousemove',noticeGShellInside);
1166noticeGShellInside();
1166
1166const FooterName = 'GshFooter'
1166function DestroyFooter(){
1166  if( footer = document.getElementById(FooterName) != null ){
1166    //footer.parentNode.removeChild(footer);
1166    empty = document.createElement('div');
1166    empty.id = 'GshFooter0';
1166    footer.parentNode.replaceChild(empty, footer);
1166  }
1166}
1166function showFooter(){
1166  footer = document.createElement('div');
1166  footer.id = FooterName;
1166  footer.style.backgroundColor = "url("+ITSmoreQR+")";
1166  //GshFooter0.parentNode.appendChild(footer);
1166  GshFooter0.parentNode.replaceChild(footer, GshFooter0);
1166}
1166</script>
1166</-- } -->
1166
1166border:20px inset #888;
1166-->
1166
1166<span id="VirtualDesktopCodeSpan">
1166/*
1166<details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
1166</-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
1166<style>
1166.VirtualSpace {
1166  z-index:0;
1166  width:1280px !important; xheight:720px !important;
1166  width:5120px; height:2880px;
1166  border-width:0px;
1166  xxbackground-color:rgba(32,32,160,0.8);
1166  xxbackground-image:url("WD-WallPaper03.png");
1166  xxbackground-size:100% 100%;
1166  color:#22a;xfont-family:Georgia;font-size:10pt;
1166  xoverflow:scroll;
1166}
1166.VirtualGrid {
1166  z-index:0;
1166  position:absolute;
1166  width:800px; height:500px;
1166  border:1px inset #fff;
1166  color:rgba(192,255,192,0.8);
1166  font-family:Georgia, Courier New;
1166  text-align:right;
1166  vertical-align:middle;
1166  font-size:200px;
1166  text-shadow:4px 4px #ccc;
1166}
1166.WD_GridScroll {
1166  z-index:100000;
1166  background-color:rgba(200,200,200,0.1);
1166}
1166.VirtualDesktop {
1166  z-index:0;
1166  position:relative;
1166  resize:both !important;
1166  overflow:scroll;
1166  display:block;
1166  min-width:120px !important; min-height:60px !important;
1166  width:600px;
1166  height:500px;
1166  border:10px inset #228;
1166  border-width:30px; border-radius:20px;
1166  background-image:url("WD-WallPaper03.png");
1166  background-size:100% 100%;
1166  color:#22a;font-family:Georgia;font-size:10pt;
1166}
1166/*comment {
1166  // overflow-scroll seems to bound childrens' view in the element span
1166  // specifying overflow seems fix the position of the element
1166}
1166.VirtualBrowserSpan {
1166  z-index:10;
1166  xxxborder:0.5px dashed #fff !important;
1166  border-color:rgba(255,255,255,0.5) !important;
1166  position:relative;
1166  left:100px;
1166  top:100px;
1166  display:block;
1166  resize:both !important;
1166  width:540px;
1166  height:320px;
1166  min-width:40px !important; min-height:20px !important;
1166  max-width:5120px !important; max-height:2880px !important;
1166  background-color:rgba(255,200,255,0.1);
1166  xoverflow:scroll;
1166}
1166.xVirtualBrowserLocationBar:focus {
1166  color:#f00;
1166  background-color:rgba(255,128,128,0.2);
1166}
1166.xVirtualBrowserLocationBar:active {
1166  color:#f00;
1166  background-color:rgba(128,255,128,0.2);
1166}
1166.a.VirtualBrowserLocation {
1166  color:#ccc !important;
1166  text-decoration:none !important;
1166}
1166.a.VirtualBrowserLocation:hover {
1166  color:#fff !important;
1166  text-decoration:underline;
1166}
1166.VirtualBrowserLocationBar {
1166  position:absolute;
1166  z-index:100000;
1166  display:block;
1166  width:400px;
1166  height:20px;
1166  padding-left:2px;
1166  line-height:1.1;
1166  vertical-align:middle;
1166  font-size:14px;
1166  color:#fff;
1166  background-color:rgba(128,128,128,0.2);
1166  font-family:Georgia;
1166}
1166.VirtualBrowserCommandBar {
1166  position:absolute;
1166  z-index:200000;
1166  xxdisplay:inline;
1166  display:block;
1166  width:60px;
1166  height:20px;
1166  line-height:1.1;
1166  vertical-align:middle;
1166  font-size:14px;
1166  color:#f00;
1166  background-color:rgba(128,128,128,0.1);
1166  font-family:Georgia;
1166  text-align:left;
1166  left:404px;
1166}
1166.VirtualBrowserFrame {
1166  xposition:relative;
1166  position:absolute;
1166  xxdisplay:inline;
1166  display:block;
1166  z-index:10;
1166  resize:both !important;
1166  width:480px; height:240px;
1166  min-width:60px; min-height:30px;
1166  max-width:5120px; max-height:2880px;
1166  border-radius:6px;
1166  background-color:rgba(255,255,255,0.9);
1166  border-top:20px solid;
1166  border-right:4px solid;
1166  border-bottom:10px solid;
1166}
1166.WinFavicon {
1166  width:16px;
1166  height:16px;

```

```

11859 margin:1px;
11860 margin-right:3px;
11861 vertical-align:middle;
11862 background-color:rgba(255,255,255,1.0);
11863 }
11864 .VirtualDesktopMenuBar {
11865   xposition:absolute;
11866   color:#fff;
11867   font-size:7pt;
11868   text-align:right;
11869   padding-right:4px;
11870   background-color:rgba(128,128,128,0.7);
11871 }
11872 .VirtualDesktopCalendar {
11873   color:#fff;
11874   font-size:22pt;
11875   text-align:right;
11876   padding-right:4px;
11877   xbackground-color:rgba(255,255,255,0.2);
11878 }
11879 .xxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
11880 display:inline !important; font-size:10pt !important; padding:1px !important;
11881 }
11882 .WD_Config {
11883   display:inline !important;
11884   padding:2px !important;
11885   font-size:10pt !important;
11886   width:60pt !important;
11887   height:12pt !important;
11888   line-height:1.0pt !important;
11889   height:15pt !important;
11890 }
11891 .WD_Button {
11892   display:inline !important;
11893   padding:1px !important;
11894   color:#fff !important;
11895   background-color:#228 !important;
11896   font-size:10pt !important;
11897   width:60pt !important;
11898   height:12pt !important;
11899   line-height:1.0pt !important;
11900   height:16pt !important;
11901   border:2px inset #44a !important;
11902 }
11903 .WD_Href {
11904   display:inline !important;
11905   padding:2px !important;
11906   font-size:9pt !important;
11907   width:120pt !important;
11908   height:12pt !important;
11909   line-height:1.0pt !important;
11910   height:15pt !important;
11911 }
11912 }
11913 .LiveHtmlCodeviewText {
11914   font-size:10pt;
11915   font-family:Courier New;
11916   xwhite-space:pre;
11917 }
11918 }
11919 .WD_Panel {
11920   x-index:100 !important;
11921   color:#000 !important;
11922   margin-left:25px !important;
11923   width:800px !important;
11924   padding:6px !important;
11925   border:1px solid #888 !important;
11926   border-radius:6px !important;
11927   background-color:rgba(220,220,220,0.9) !important;
11928   font-size:9pt;
11929   font-family:Courier New;
11930 }
11931 .WD_Help {
11932   font-size:10pt !important;
11933   font-family:Courier New;
11934   line-height:1.2 !important;
11935   color:#000 !important;
11936   width:100% !important;
11937   background-color:rgba(240,240,255,0.8) !important;
11938 }
11939 }
11940 .WB_Zoom {
11941 }
11942 }
11943 </style>
11944 <h2>CosmoScreen 0.0.8</h2>
11945 <menu>
11946 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
11947 g ... grid on/off<br>
11948 i ... zoom in<br>
11949 o ... zoom out<br>
11950 s ... save current scope<br>
11951 r ... restore saved scope<br>
11952 </span>
11953 </menu>
11954 <div class="WD_Panel" draggable="true">
11955 <p>-- should be on the frame of the WD -->
11956 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11957 <input id="WD_Width_1" class="WD_Config" type="text">
11958 <input id="WD_Height_1" class="WD_Config" type="text">
11959 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
11960 </p>
11961 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
11962 <input id="WS_1_Width" class="WD_Config" type="text">
11963 x <input id="WS_1_Height" class="WD_Config" type="text">
11964 </p>
11965 <p>
11966 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
11967 <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
11968 x <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
11969 <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
11970 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
11971 </p>
11972 <p>
11973 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
11974 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
11975 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
11976 shift+wheel for horizontal scroll
11977 </p>
11978 <p>
11979 Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
11980 <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
11981 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
11982 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11983 </p>
11984 <p>
11985 Scopeyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
11986 <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
11987 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
11988 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
11989 </p>
11990 <p>
11991 Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
11992 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
11993 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
11994 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
11995 </p>
11996 <p>
11997 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
11998 </p>
11999 <p>
12000 Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
12001 "scroll" imprisons windows inside the display
12002 </p>
12003 </div>
12004 </div>
12005 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
12006 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
12007 <div id="VirtualDesktop_1_Clock" class="VirtualDesktopClock"></div>
12008 </div>
12009 <div id="VirtualDesktop_1_Calendar" class="VirtualDesktopCalendar">>00:00</div>
12010 <div align="right"><div id="VirtualSpace_1" class="VirtualSpace"></div>
12011 <div id="VirtualDesktop_1_Content" class="VirtualSpace">
12012 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
12013 <div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
12014 <span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
12015 <iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></iframe>
12016 </div>
12017 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
12018 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
12019 <span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
12020 <iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></iframe>
12021 </div>
12022 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
12023 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
12024 <span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
12025 <iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></iframe>
12026 </div>
12027 </div>
12028 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
12029 </div>
12030 </div>
12031 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">

```

```

1203 <span id="VirtualDesktopCodeview"></span>
1203 <script id="VirtualDesktopScript">
1203 function vVirtualDesktopCodeview() {
1203     codespan = document.getElementById('VirtualDesktopCodeSpan');
1204     showHtmlCode(VirtualDesktopCodeview,codespan);
1204     VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
1204 }
1204 DestroyEventSharingCodeview = function(){
1204     VirtualDesktopCodeview.innerHTML = "";
1204 }
1204 }
1204 function wdlog(log){
1204     if( GJ_Channel != null ){
1204         GJ_SendMessage('WD '+log);
1205     }
1205     console.log(log);
1205 }
1205 var topMostWin = 10000;
1205 function onEnterWin(e){
1205     t = e.target;
1205     oindex = t.style.zIndex;
1205     //if( oindex == '' ) oindex = 0;
1205     //t.saved_zIndex = oindex;
1205     //t.style.zIndex = 10000;
1206     topMostWin += 1;
1206     t.style.zIndex = topMostWin;
1206     nindex = t.style.zIndex;
1206     wdlog('Enter '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
1206     e.stopPropagation();
1206     e.preventDefault();
1206 }
1206 function onClickWin(e) { // can detect click on the thick border? t = e.target;
1206     oindex = t.style.zIndex;
1206     topMostWin += 1;
1206     t.style.zIndex = topMostWin;
1207     nindex = t.style.zIndex;
1207     wdlog('Click '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
1207     //e.stopPropagation();
1207     //e.preventDefault();
1207 }
1207 function onLeaveWin(e){
1207     t = e.target;
1207     //oindex = t.style.zIndex;
1207     //nindex = t.saved_zIndex;
1207     //t.style.zIndex = nindex;
1208     //wdlog('Leave '+e.target+' #'+e.target.id+'('+oindex+'->'+nindex+')');
1208     e.stopPropagation();
1208     e.preventDefault();
1208 }
1208 }
1208 var WinDragstartX; // event
1208 var WinDragstartY;
1208 var WinDragstartTX; // target
1208 var WinDragstartTY;
1209 function onWinDragstart(e){
1209     WinDragstartX = e.x;
1209     WinDragstartY = e.y;
1209     t = e.target;
1209     //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
1209     //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
1209     if( t.style.left == '' ){
1210         WinDragstartTX = x0 = 0;
1210         t.style.left = '0px';
1210     }else{
1210         //WinDragstartTX = x0 = Number(t.style.left);
1210         WinDragstartTX = x0 = parseInt(t.style.left);
1210     }
1210     if( t.style.top == '' ){
1210         WinDragstartTY = y0 = 0;
1210         t.style.top = '0px';
1210     }else{
1210         //WinDragstartTY = y0 = Number(t.style.top);
1210         WinDragstartTY = y0 = parseInt(t.style.top);
1210     }
1211     if( true ){ // to be undo
1211         t.wasAtX = WinDragstartTX;
1211         t.wasAtY = WinDragstartTY;
1211     }
1211     wdlog('DragSTA #'+t.id
1211         + ' event('+e.x+', '+e.y+')'
1211         + ' position=' + t.style.position
1211         + ' style left,top('+t.style.left+', '+t.style.top+')'
1211     );
1212     e.stopPropagation();
1212     //e.preventDefault();
1212     return true;
1212 }
1212 function onWinDragEvent(wh,e,set,dolog){
1212     t = e.target;
1212     dx = e.x - WinDragstartX;
1212     dy = e.y - WinDragstartY;
1212     nx = WinDragstartTX + dx;
1212     ny = WinDragstartTY + dy;
1212     log = 'Drag'+wh+' #'+t.id
1212         + ' event0('+WinDragstartX+', '+WinDragstartY+')'
1212         + ' event1('+e.x+', '+e.y+')'
1212         + ' diff('+dx+', '+dy+')'
1212         + ' (' + nx + ', ' + ny + ') '
1212         + ' (' + t.style.left + ', ' + t.style.top + ') '
1212         + ' wasAt('+ t.wasAtX + ', ' + t.wasAtY + ') '
1212     ;
1213     if( e.x != 0 || e.y != 0 ){
1213         if( set == true ){
1213             //t.style.x = nx + 'px'; // not effective
1213             //t.style.y = ny + 'px'; // not effective
1213             t.style.left = nx + 'px';
1213             t.style.top = ny + 'px';
1213             log += ' Set';
1213         }else{
1213             log += ' NotSet';
1213             if( !dolog ){
1213                 log = '';
1213             }
1213         }
1213     }else{
1213         log += ' What?'; // the type is event start?
1213         if( !dolog ){
1213             log = '';
1213         }
1213     }
1213     if( and(dolog, log != '') ){
1213         wdlog(log);
1213     }
1213     if( true ){
1213         // should be propagated to parent in FireFox ?
1213         e.stopPropagation();
1213     }
1213     e.preventDefault();
1213     return false;
1213 }
1213 }
1213 function onWinDrag(e){
1213     return onWinDragEvent('Ing',e,true,false);
1213 }
1213 function onWinDragend(e){
1213     return onWinDragEvent('End',e,false,true);
1213 }
1213 function onWinDragexit(e){
1213     return onWinDragEvent('Exit',e,false,true);
1213 }
1213 function onWinDragover(e){
1213     return onWinDragEvent('Over',e,false,true);
1213 }
1213 function onWinDragenter(e){
1213     return onWinDragEvent('Enter',e,false,true);
1213 }
1213 function onWinDragleave(e){
1213     return onWinDragEvent('Leave',e,false,true);
1213 }
1213 function onWinDragdrop(e){
1213     return onWinDragEvent('Drop',e,false,true);
1213 }
1213 function onFaviconChange(e){
1213     wdlog('--Favicon #'+e.target.id+' href='+e.details.href);
1213 }
1213 var savedSuppressGJShell = false;
1213 function stopGShell(e){
1213     //wdlog('enter Gsh STOP');
1213     savedSuppressGJShell = SuppressGJShell;
1213     SuppressGJShell = true;
1213     e.stopPropagation();
1213     e.preventDefault();
1213 }
1213 function contGShell(e){
1213     //wdlog('leave Gsh STOP');
1213     SuppressGJShell = savedSuppressGJShell;
1213     e.stopPropagation();
1213     e.preventDefault();
1213 }
1213 }
1213 function WD_onkeydown(e){
1213     keycode = e.code;
1213     console.log('Keydown #'+e.target.id+' '+keycode);
1213     if( keycode == 'KeyG' ){
1213         WD_setGrid1(WD_Grid_1);
1213     }
1213 }

```

```

12213 }else
12214 if( keycode == 'KeyI' ){
12215     WD_doZoomIn();
12216 }else
12217 if( keycode == 'KeyO' ){
12218     WD_doZoomOut();
12219 }else
12220 if( keycode == 'KeyR' ){
12221     WD_RestoreScope(null);
12222 }else
12223 if( keycode == 'KeyS' ){
12224     WD_SaveScope(null);
12225 }
12226 e.stopPropagation();
12227 e.preventDefault();
12228 }
12229 function WD_onKeyUp(e){
12230     e.stopPropagation();
12231     e.preventDefault();
12232 }
12233 function WD_EventSetup1(){
12234     WirtualDesktop_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
12235     WirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeyDown(e); });
12236     WD_Help_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
12237     WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
12238 }
12239 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
12240     function WirtualBrowserCommand(e,s,l,cmd,f){
12241         command = cmd.innerHTML;
12242         if( command == "Reload" ){
12243             href_id = e.target.href_id;
12244             d = document.getElementById(href_id);
12245             wlog('href_tag='+href_id+'\n elem=#'+href_id+'\n href='+d);
12246             url = d.innerHTML;
12247             wlog('---- Load href_tag='+href_id+'\n elem=#'+href_id+'\n href='+d
12248                 + '\n url='+url);
12249             wlog('---- Load target #' + f.id + ' with url=' + url);
12250             f.src = url;
12251         }else{
12252             alert('unknown command'+command+' '+e.target.id+', '+l.id+', '+f.id);
12253         }
12254     }
12255     function onKeyDown(e){
12256         if( e.code == 'Enter' ){
12257             e.stopPropagation();
12258             e.preventDefault();
12259         }
12260     }
12261     function onKeyUp(e){
12262         if( e.code == 'Enter' ){
12263             e.stopPropagation();
12264             e.preventDefault();
12265             // should reload immediately ?
12266         }
12267     }
12268     if( false ){
12269         wlog('start settle WirtualBrowser url='+u + '\n'
12270             + 'id=' + s.id + '\n'
12271             + 'width=' + s.style.width + '\n'
12272             + 'height=' + s.style.height
12273         );
12274     }
12275     // very important for WordPress ??
12276     s.style.width = f.style.width = 501; // for WordPress ...??
12277     s.style.height = f.style.height = 271; // for WordPress ...??
12278     if( false ){
12279         wlog('midway settle WirtualBrowser url='+u + '\n'
12280             + 'id=' + s.id + '\n'
12281             + 'width=' + s.style.width + '\n'
12282             + 'height=' + s.style.height
12283         );
12284     }
12285     s.width = 502; // for WordPress ...??
12286     s.height = 272; // for WordPress ...??
12287     if( false ){
12288         wlog('midway-2 settle WirtualBrowser url='+u + '\n'
12289             + 'id=' + s.id + '\n'
12290             + 'span-width=' + s.width + '\n'
12291             + 'span-height=' + s.height
12292         );
12293     }
12294     s.style.width = w + 'px';
12295     s.style.height = h + 'px';
12296     f.style.width = w + 'px';
12297     f.style.height = h + 'px';
12298     //f.style.setProperty('-webkit-transform', 'scale('+scale+')');
12299     f.style.setProperty('transform', 'scale('+scale+')');
12300     //wlog('---x1-- u='+u+ ' width s='+s.style.width+f, f='+f.style.width);
12301     //wlog('---x2-- u='+u+ ' width s='+s.style.width+f, f='+f.style.width);
12302     s.setAttribute('draggable', 'true');
12303     f.setAttribute('draggable', 'false'); // why necessary?
12304     l.setAttribute('draggable', 'false'); // why necessary?
12305     cmd.setAttribute('draggable', 'false'); // why necessary?
12306     s.addEventListener('dragstart', e => { onWinDragstart(e); });
12307     s.addEventListener('drag', e => { onWinDrag(e); });
12308     s.addEventListener('dragend', e => { onWinDragend(e); });
12309     s.addEventListener('dragexit', e => { onWinDragexit(e); });
12310     s.addEventListener('dragover', e => { onWinDragover(e); });
12311     s.addEventListener('dragleave', e => { onWinDragleave(e); });
12312     s.addEventListener('drop', e => { onWinDragdrop(e); });
12313     s.addEventListener('mouseenter', e => { onEnterWin(e); });
12314     s.addEventListener('mouseleave', e => { onLeaveWin(e); });
12315     if( false ){
12316         s.style.position = "absolute";
12317         s.style.x = x+'px';
12318         s.style.left = x+'px';
12319         s.style.y = y+'px';
12320         s.style.top = y+'px';
12321     }else{
12322         s.style.setProperty('position', 'absolute', 'important');
12323         s.style.setProperty('x', x+'px', 'important');
12324         s.style.setProperty('left', x+'px', 'important');
12325         s.style.setProperty('y', y+'px', 'important');
12326         s.style.setProperty('top', y+'px', 'important');
12327     }
12328     favicon = './favicon.ico';
12329     uv1 = u.split('/');
12330     if( 2 <= uv1.length ){
12331         uv2 = uv1[1].split('/');
12332         if( 2 <= uv2.length ){
12333             if( uv1[0] == 'file' ){
12334                 //favicon = 'file://' + uv2.slice(0, uv2.length-1).join('/');
12335                 // + './favicon.ico';
12336                 favicon = './favicon.ico';
12337             }else{
12338                 favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
12339             }
12340         }
12341     }
12342     wlog('---- favicon-url='+favicon);
12343     href_id = l.id + ' href';
12344     l.innerHTML = ''+u+'</a>';
12346     //l.addEventListener('click', e => { onClickWin(e); });
12347     l.addEventListener('mouseenter', e => { stopShell(e); });
12348     l.addEventListener('mouseleave', e => { contShell(e); });
12349     l.addEventListener('keydown', e => { onKeyUp(e); });
12350     l.addEventListener('keyup', e => { onKeyUp(e); });
12351     cmd.href_id = href_id;
12352     wlog(' (0)cmd=#'+cmd.id);
12353     wlog(' (1)href_id=#'+href_id);
12354     wlog(' (2)href_id=#'+cmd.href_id);
12355     cmd.addEventListener('click', e => { WirtualBrowserCommand(e,s,l,cmd,f); });
12356     f.style.borderColor = c;
12357     f.src = u;
12358     //f.addEventListener('mouseenter', e => { onEnterWin(e); });
12359     //f.addEventListener('mouseleave', e => { onLeaveWin(e); });
12360     //s.addEventListener('click', e => { onClickWin(e); });
12361     //f.addEventListener('click', e => { wlog('click w1'); });
12362     f.addEventListener('mouseover', e => { onFaviconChange(e); });
12363     wlog('done settle WirtualBrowser url='+u + '\n'
12364         + 'id=' + s.id + ' '
12365         + 'width=' + s.style.width + ' '
12366         + 'height=' + s.style.height + ' '
12367         + 'cmd' + cmd.id
12368     );
12369 }
12370 function WD_EventSetup2(){
12371     dt = WirtualDesktop_1;
12372     dt.style.width = "800px";
12373     dt.style.height = "500px";
12374     dt.addEventListener('dragstart', e => { onWinDragstart(e); });
12375     dt.addEventListener('drag', e => { onWinDrag(e); });
12376     dt.addEventListener('exit', e => { onWinDragexit(e); });

```

```

12390}
12391
12392function GRonClick(){
12393    WD_SaveScope(null); // should be push
12394    t = event.target;
12395    x = t.getAttribute('data-leftx');
12396    y = t.getAttribute('data-topy');
12397    zoom = WD_Zoom_1_XY.value;
12398    x *= zoom;
12399    y *= zoom;
12400    WD_doScrollXY(event,x,y);
12401    //alert('scroll #' + t.id + ' x=' + x + ', y=' + y);
12402}
12403
12404function WD_setGrid(e){
12405    t = e.target;
12406    WD_setGrid(t);
12407}
12408
12409function WD_setGrid(t){
12410    //ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
12411    ds = WirtualDesktop_1_GridPlane;
12412    if( t.value == 'GridOn' ){
12413        for( col = 0; col < 16; col++ ){
12414            for( row = 0; row < 16; row++ ){
12415                gl = document.createElement('span');
12416                gl.setAttribute('class', 'WirtualGrid');
12417                leftx = col * 800;
12418                topy = row * 500;
12419                gid = col + ' ' + row
12420                label = '<+span '
12421                    + 'id="' + gid + '" ' + 'class="WD_GridScroll" '
12422                    + 'contenteditable="false" onclick="GRonClick()" '
12423                    + 'data-leftx=" ' + leftx + ' ' + 'data-topy=" ' + topy + ' '
12424                    + '>';
12425                console.log('grid ' + label);
12426                gl.innerHTML = label;
12427                gl.position = 'relative';
12428                gl.leftx = leftx;
12429                gl.topy = topy;
12430                gl.style.left = gl.leftx + 'px';
12431                gl.style.top = gl.topy + 'px';
12432                if( col % 2 == row % 2 ){
12433                    gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
12434                }
12435                ds.appendChild(gl);
12436            }
12437            t.value = 'GridOff';
12438        }else{
12439            ds.innerHTML = '';
12440            t.value = 'GridOn';
12441        }
12442    }
12443}
12444
12445var sav_scrollLeft;
12446var sav_scrollTop;
12447var sav_nscales;
12448function WD_SaveScope(e){
12449    sav_scrollLeft = WD_Left_1.value;
12450    sav_scrollTop = WD_Top_1.value;
12451    sav_nscales = WD_Zoom_1_XY.value;
12452    //console.log('saved zoom=' + sav_oscales + ', ' + sav_nscales);
12453}
12454
12455function WD_RestoreScope(e){
12456    WD_Zoom_1_XY.value = sav_nscales;
12457    WD_doZoom();
12458}
12459
12460WD_Left_1.value = sav_scrollLeft;
12461WD_Top_1.value = sav_scrollTop;
12462WD_doScroll(null);
12463}
12464
12465function ignoreEvent(e){
12466    e.stopPropagation();
12467    //e.preventDefault();
12468}
12469
12470function zoomMag(){
12471    return WD_Zoom_1_MAG.value;
12472}
12473
12474function WD_EventSetup3(){
12475    WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
12476    WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
12477    WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
12478    WD_Width_1.value = dt.style.width;
12479    WD_Width_1.addEventListener('keydown', ignoreEvent);
12480    WD_Height_1.value = dt.style.height;
12481    WD_Height_1.addEventListener('keydown', ignoreEvent);
12482    WD_Height_1.addEventListener('keyup', ignoreEvent);
12483    WD_Height_1.addEventListener('keydown', ignoreEvent);
12484    WD_Height_1.addEventListener('keyup', ignoreEvent);
12485    WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
12486    WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
12487}
12488
12489function escale1(e,oscale,nscale){
12490    e.style.setProperty('transform', 'scale('+nscale+')');
12491    rscale = oscale / nscale;
12492    w = parseInt(e.style.width);
12493    h = parseInt(e.style.height);
12494    w = w * rscale; //(oscale/nscale);
12495    h = h * rscale; //(oscale/nscale);
12496    e.style.width = w + 'px';
12497    e.style.height = h + 'px';
12498}
12499
12500function scaleWD(ds,oscale,nscale){
12501    if( true ){
12502        escale1(WirtualBrowser_1,oscale,nscale);
12503        escale1(WirtualBrowser_1_Location,oscale,nscale);
12504        escale1(WirtualBrowser_1_Command,oscale,nscale);
12505        escale1(WirtualBrowser_1_Frame,oscale,nscale);
12506        escale1(WirtualBrowser_2,oscale,nscale);
12507        escale1(WirtualBrowser_2_Location,oscale,nscale);
12508        escale1(WirtualBrowser_2_Command,oscale,nscale);
12509        escale1(WirtualBrowser_2_Frame,oscale,nscale);
12510        escale1(WirtualBrowser_3,oscale,nscale);
12511        escale1(WirtualBrowser_3_Location,oscale,nscale);
12512        escale1(WirtualBrowser_3_Command,oscale,nscale);
12513        escale1(WirtualBrowser_3_Frame,oscale,nscale);
12514    }
12515}
12516
12517function WD_doZoom(){
12518    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
12519    oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
12520    nscale = WD_Zoom_1_XY.value;
12521    ds.style.zoom = nscale;
12522    WD_Zoom_1_XY.value = ds.style.zoom;
12523    WD_Zoom_1_XY.ovalue = ds.style.zoom;
12524    scaleWD(ds,oscale,nscale);
12525}
12526
12527function WD_EventSetup4(){
12528    WD_Zoom_1.addEventListener('click', WD_doZoom);
12529    WD_Zoom_1_XY.addEventListener('keydown', ignoreEvent);
12530    WD_Zoom_1_XY.addEventListener('keyup', ignoreEvent);
12531}
12532
12533function WD_doZoomOUT(){
12534    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
12535    oscale = WD_Zoom_1_XY.value;
12536    if( oscale == 0 || oscale == '' ){
12537        oscale = 1;
12538    }
12539    nscale = oscale / zoomMag();
12540    ds.style.zoom = nscale;
12541    WD_Zoom_1_XY.value = ds.style.zoom;
12542    WD_Zoom_1_XY.ovalue = ds.style.zoom;
12543    scaleWD(ds,oscale,nscale);
12544}
12545
12546function WD_doZoomIN(){
12547    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
12548    oscale = WD_Zoom_1_XY.value;
12549    if( oscale == 0 || oscale == '' ){
12550        oscale = 1;
12551    }
12552    nscale = oscale * zoomMag();
12553    if( 4 < nscale ){
12554        alert('maybe too large, zoom=' + nscale);
12555        return;
12556    }
12557    ds.style.zoom = nscale;
12558    WD_Zoom_1_XY.value = ds.style.zoom;
12559    WD_Zoom_1_XY.ovalue = ds.style.zoom;
12560    scaleWD(ds,oscale,nscale);
12561}
12562
12563function WD_EventSetup5(){
12564    WD_Zoom_1_OUT.addEventListener('click', WD_doZoomOUT);
12565    WD_Zoom_1_IN.addEventListener('click', WD_doZoomIN);
12566}
12567
12568function WD_doResize(e){
12569    dt = WirtualDesktop_1;
12570    dt.style.width = WD_Width_1.value;
12571    dt.style.height = WD_Height_1.value;
12572    WD_Width_1.value = dt.style.width;
12573    WD_Height_1.value = dt.style.height;
12574}
12575
12576WD_Resize_1.addEventListener('click', e => { WD_doResize(e); });
12577
12578

```

```

12567 function WD_dorSResize(e){
12568 //alert('Resize Space');
12570 ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
12571 ds.style.width = WS_1_Width.value;
12572 ds.style.height = WS_1_Height.value;
12573 WS_1_Width.value = ds.style.width;
12574 WS_1_Height.value = ds.style.height;
12575 }
12576 function WD_EventSetup6(){
12577 ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
12578 ds.style.width = '5120px';
12579 ds.style.height = '2880px';
12580 WS_1_Width.value = ds.style.width;
12581 WS_1_Height.value = ds.style.height;
12582 WS_1_Width.addEventListener('keydown', ignoreEvent);
12583 WS_1_Width.addEventListener('keyup', ignoreEvent);
12584 WS_1_Height.addEventListener('keydown', ignoreEvent);
12585 WS_1_Height.addEventListener('keyup', ignoreEvent);
12586 WS_Resize_1.addEventListener('click', e => { WD_dorSResize(e); });
12587 }
12588
12589 function WD_doScrollXY(e, left, stop){
12590 dt = VirtualDesktop_1;
12591 dt.scrollLeft = left;
12592 dt.scrollTop = stop;
12593 WD_Left_1.value = dt.scrollLeft;
12594 WD_Top_1.value = dt.scrollTop;
12595 console.log('--Scroll #' + dt.id + ' (' + left + ', ' + stop + ')');
12596 }
12597 function WD_doScroll(e){
12598 //dt = VirtualDesktop_1_Content;
12599 dt = VirtualDesktop_1;
12600 left = parseInt(WD_Left_1.value);
12601 stop = parseInt(WD_Top_1.value);
12602 dt.scrollLeft = left;
12603 dt.scrollTop = stop;
12604 WD_Left_1.value = dt.scrollLeft;
12605 WD_Top_1.value = dt.scrollTop;
12606 console.log('--Scroll #' + dt.id + ' (' + left + ', ' + stop + ')');
12607 }
12608 function showScrollPosition(){
12609 if( false ){
12610 console.log(
12611 'wtop' + VirtualDesktop_1.style.top + ', ' +
12612 'wxs' + VirtualDesktop_1.style.y + ', ' +
12613 'wss' + VirtualDesktop_1.scrollTop + ', ' +
12614 'wdtop' + VirtualDesktop_1_Content.style.top + ', ' +
12615 'wdxs' + VirtualDesktop_1_Content.style.y + ', ' +
12616 'wds' + VirtualDesktop_1_Content.scrollTop + ', ' +
12617 );
12618 WD_Left_1.value = VirtualDesktop_1.scrollLeft;
12619 WD_Top_1.value = VirtualDesktop_1.scrollTop;
12620 }
12621 function WD_EventSetup7(){
12622 WD_Scroll_1.addEventListener('click', e => { WD_doScroll(e); });
12623 WD_Left_1.addEventListener('keydown', ignoreEvent);
12624 WD_Left_1.addEventListener('keyup', ignoreEvent);
12625 WD_Top_1.addEventListener('keydown', ignoreEvent);
12626 WD_Top_1.addEventListener('keyup', ignoreEvent);
12627 }
12628 function WD_EventSetup8(){
12629 VirtualDesktop_1.addEventListener('scroll', showScrollPosition);
12630 VirtualDesktop_1_Content.addEventListener('scroll', showScrollPosition);
12631 }
12632
12633 if( false ){
12634 w = 1000 + 'px';
12635 dt.style.width = w;
12636 dt.style.height = '300px';
12637 dt.style.resize = 'both';
12638 dt.style.borderWidth = 50 + 'px';
12639 dt.style.borderRadius = 25 + 'px';
12640 console.log('---2----- #' + dt.id + ' style=' + dt.style);
12641 console.log('----- #' + dt.id + ' width=' + dt.style.width);
12642 console.log('----- #' + dt.id + ' left=' + dt.style.left);
12643 console.log('----- #' + dt.id + ' border=' + dt.style.border);
12644 }
12645 function onDTRize(e){
12646 dt = e.target;
12647 h = parseInt(dt.style.height);
12648 dt.style.border = (h * 0.075) + 'px';
12649 console.log('----- borderWidgh=' + dt.style.borderWidth);
12650 }
12651
12652 VirtualDesktopDetails.addEventListener('toggle', VirtualDesktop_init);
12653 function VirtualDesktop_init(){
12654 if( !VirtualDesktopDetails.open ){
12655 return;
12656 }
12657 //GJ_Join();
12658 VirtualDesktop_1.addEventListener('resize', e => { onDTRize(e); });
12659 //console.log('----- #' + dt.id
12660 // + ' borderWidgh=' + dt.style.getProperty('border-width'));
12661
12662 settleWin(
12663 VirtualBrowser_1,
12664 VirtualBrowser_1_Location,
12665 VirtualBrowser_1_Command,
12666 VirtualBrowser_1_Frame,
12667 document.URL,
12668 500, 280, 50, 20, '#262', 1.0);
12669 settleWin(
12670 VirtualBrowser_2,
12671 VirtualBrowser_2_Location,
12672 VirtualBrowser_2_Command,
12673 VirtualBrowser_2_Frame,
12674 'https://its-more.jp/ja_jp/',
12675 500, 280, 150, 100, '#448', 1.0);
12676 settleWin(
12677 VirtualBrowser_3,
12678 VirtualBrowser_3_Location,
12679 VirtualBrowser_3_Command,
12680 VirtualBrowser_3_Frame,
12681 '...', 'gshell/gsh.go.html',
12682 'http://gshell.org/gshell/gsh.go.html',
12683 'https://golang.org',
12684 500, 280, 250, 180, '#444', 1.0);
12685 //1000, 720, 0, 0, '#444', 0.125);
12686 //1200, 720, -100, -50, '#444', 0.4);
12687 function WD_ClockUpdate(e){
12688 VirtualDesktop_1_Clock.innerHTML = DateShort();
12689 VirtualDesktop_1_Calender.innerHTML = DateHourMin();
12690 }
12691 window.setInterval(WD_ClockUpdate, 500);
12692
12693 WD_EventSetup1();
12694 WD_EventSetup2();
12695 WD_EventSetup3();
12696 WD_EventSetup4();
12697 WD_EventSetup5();
12698 WD_EventSetup6();
12699 WD_EventSetup7();
12700 WD_EventSetup8();
12701 }
12702 //VirtualDesktop_init();
12703
12704 _Destroy_VirtualDesktop = function(){
12705 VirtualDesktop_1.style = '';
12706
12707 VirtualBrowser_1.removeAttribute('style');
12708 VirtualBrowser_1_Location.innerHTML = '';
12709 VirtualBrowser_1_Frame.removeAttribute('src');
12710 VirtualBrowser_1_Frame.removeAttribute('style');
12711 VirtualBrowser_1_Frame.style = '';
12712
12713 VirtualBrowser_2.removeAttribute('style');
12714 VirtualBrowser_2_Location.innerHTML = '';
12715 VirtualBrowser_2_Frame.removeAttribute('src');
12716 VirtualBrowser_2_Frame.style = '';
12717
12718 VirtualBrowser_3.removeAttribute('style');
12719 VirtualBrowser_3_Location.innerHTML = '';
12720 VirtualBrowser_3_Frame.removeAttribute('src');
12721 VirtualBrowser_3_Frame.style = '';
12722
12723 GJFactory_1.style = '';
12724 iframe0.style = '';
12725 VirtualDesktop_1.style = '';
12726 }
12727
12728 </script>
12729 </-- VirtualDesktop -->
12730 <details>
12731 <span> *! //</span>
12732
12733 </-- Work { ===== -->
12734 <span id="SBSidebar_WorkCodeSpan">
12735 *
12736 <details><summary>SBSidebar</summary>
12737 <-- SBSidebar // 2020-0928 SatoxITS { -->
12738 <h2>SBSidebar</h2>
12739 <input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12740 <input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12741 <input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12742 <span id="SBSidebar_WorkCodeView"></span>
12743 <script id="SBSidebar_WorkScript">

```



```

12744 function SBSidebar_openWorkCodeView(){
12745   function SBSidebar_showWorkCode(){
12746     showWorkCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
12747   }
12748   SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
12749 }
12750 SBSidebar_openWorkCodeView(); // should be invoked by an event
12751
12752 console.log('-- Sbslider // 2020-1006-01 SatoxITS --');
12753 function SetSidebar(){
12754   sidebar = document.getElementById('secondary');
12755   console.log('primary='+primary+' + secondary='+sidebar+' + main='+main+' ');
12756   wrap = sidebar.parentNode;
12757   console.log('-- Sbslider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
12758   //wrap = wrap.parentNode;
12759   //console.log('-- Sbslider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
12760   //wrap = wrap.parentNode;
12761   //console.log('-- Sbslider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
12762   //nsb = sidebar.cloneNode();
12763   nsb = sidebar;
12764   nsb.style.width = '100%';
12765   slider = document.createElement('div');
12766   slider.id = 'Sbslider';
12767   slider.appendChild(nsb);
12768   slider.setAttribute('class', 'Sbslider');
12769   nsb.style.position = 'relative';
12770   slider.style.position = 'fixed';
12771   slider.style.display = 'block'; // 'inline';
12772   slider.style.zIndex = 100000;
12773   // nsb.style.zIndex = 200000;
12774   nsb.style.position = 'absolute';
12775   nsb.style.minWidth = '80px';
12776   nsb.style.left = '0px';
12777   nsb.style.top = '0px';
12778
12779   w = window.innerWidth;
12780   console.log('SliderWidth '+w+' ',(w/3)+'px');
12781   if( w < 640 ){
12782     slider.style.setProperty('width',(w/3) + 'px','important');
12783   }
12784   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
12785
12786   slider.style.resize = "both";
12787   slider.draggable = "true";
12788   wrap.appendChild(slider);
12789   console.log('-- added Sbslider');
12790   //nsb.addEventListener('scroll',SbScrolled);
12791
12792   buttons = document.createElement('div');
12793   buttons.id = 'NaviButtons';
12794   buttons.setAttribute('class','NaviButtons');
12795   buttons.align = "center";
12796   buttons.innerHTML = '<'+>p><a href="#TopOfPost" draggable="true">TOP<'+>'+>p>';
12797   buttons.innerHTML += '<'+>p><a href="#EndOfPost" draggable="true">END<'+>'+>p>';
12798   page.appendChild(buttons);
12799   buttons.style.position = 'fixed';
12800   buttons.style.zIndex = 30000;
12801   buttons.style.width = '180%';
12802   buttons.style.top = '320px';
12803   buttons.style.left = parseInt(w) * 1.0 + 'px';
12804   console.log('-- Sbslider installed (^~/ SatoxITS');
12805 }
12806 //window.addEventListener('load',SetSidebar); // after load
12807 DestroyNaviButtons = function(){
12808   nb = document.getElementById('NaviButtons');
12809   if( nb != null ){
12810     nb.parentNode.removeChild(NaviButtons);
12811   }
12812 }
12813 </script>
12814
12815 // 2020-1006 its-more.jp-blog-60000-style.css
12816 <!-- {
12817 <style>
12818 #NaviButtons {
12819   position:fixed;
12820   display:block;
12821   width:100%;
12822   xtop:100px;
12823   xxleft:10px;
12824   z-index:30000;
12825   font-size:10pt;
12826   color:#2ff !important;
12827   text-align:center;
12828   background-color:rgba(230,230,230,0.01);
12829 }
12830 #NaviButtons a {
12831   color:#2a2 !important;
12832   font-size:20px;
12833   text-align:center;
12834   xtext-shadow:2px 2px #8ff;
12835   resize:both;
12836   padding:6px;
12837   margin:10px;
12838   border:1px solid #288 !important;
12839   border-radius:3px;
12840   background-color:rgba(160,160,160,0.05);
12841 }
12842 #Sbslider {
12843   overflow:auto;
12844   resize:both !important;
12845   xoverflow-y:hidden !important;
12846   height:100px !important;
12847   display:inline !important;
12848   position:fixed !important;
12849   left:0px;
12850   top:0px;
12851   xxwidth:180px;
12852   width:244;
12853   min-width:80px;
12854   height:100% !important;
12855   background-color:rgba(100,100,200,0.1);
12856 }
12857 #secondary {
12858   position:fixed;
12859   left:0px;
12860   top:0px;
12861   xxxz-index:60000;
12862   scroll-behavior: overflow !important;
12863   xxxxxxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
12864   padding-left:4pt;
12865   color:#fff;
12866   font-size:0.5em;
12867   background-color:rgba(64,160,64,0.6) !important;
12868   white-space:nowrap;
12869 }
12870 #secondary a {
12871   color:#fff !important;
12872   text-decoration:disable !important;
12873 }
12874 #primary {
12875   position:relative;
12876   width:75% !important;
12877   left:25% !important;
12878 }
12879 #main {
12880   position:relative;
12881   width:75% !important;
12882   left:25% !important;
12883 }
12884 #site-navigation {
12885   position:relative;
12886   left:10px;
12887 }
12888 #adswac_countertext {
12889   color:#4169e1;
12890   font-size:16pt !important;
12891   xxfont-size:10% !important;
12892   font-weight:bold;
12893 }
12894 #nowTime {
12895   color:#a0ffa0;
12896   font-size:16pt !important;
12897   xxfont-size:10% !important;
12898   font-weight:bold;
12899   text-shadow:1px 1px #fff;
12900 }
12901 .navigation-top {
12902   color:#2a !important;
12903   border:0px;
12904   background-color:rgba(220,220,220,0.1);
12905 }
12906
12907 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
12908   display: block;
12909   xxwidth: 1em;
12910   xoverflow: auto;
12911   xxheight: 1em;
12912 }
12913 .invisible-scrollbar::-webkit-scrollbar {
12914   xdisplay: none;
12915   width:1px !important;
12916   height:1px !important;
12917 }
12918 .mostly-customized-scrollbar::-webkit-scrollbar {
12919   width: 2px;
12920   height: 2px;

```

```

12921  xbackground-color: #aaa; xxx:or add it to the track;
12922 }
12923 .mostly-customized-scrollbar::-webkit-scrollbar-thumb {
12924   background: #000;
12925 }
12926 </style>
12927 } -->
12928
12929 </details>
12930 <!-- SBSidebar_WorkCodeSpan } -->
12931 <!-- /span> //</span>
12932 <!-- ===== Work } ===== -->
12933
12934
12935 <!-- ===== Work { ===== -->
12936 <span id="Affiliate_WorkCodeSpan">
12937 *
12938 <details id="Affiliate_Test"><summary>Affiliates</summary>
12939 <!-- ===== Affiliate // 2020-1010 SatoxITS { -->
12940 <div id="AffViewDock">
12941 <div id="AffView" class="AffView" draggable="true" style="">
12942 <div id="AffSet" class="AffPlate">
12943 <iframe id="aff_1" class="AffItem"></iframe>
12944 <iframe id="aff_2" class="AffItem"></iframe>
12945 <iframe id="aff_3" class="AffItem"></iframe>
12946 <iframe id="aff_4" class="AffItem"></iframe>
12947 <iframe id="aff_5" class="AffItem"></iframe>
12948 </div>
12949 </div></div>
12950 <h2>Supportive Affiliate</h2>
12951 <style>
12952 .AffView {
12953   z-index:0;
12954   overflow-x:scroll;
12955   overflow-y:scroll;
12956   position:fixed;
12957   max-width:2560px;
12958   max-height:100%;
12959   width:270px;
12960   left:75%;
12961   height:95%;
12962   resize:both;
12963   xleft:-10%;
12964   margin-top:40px;
12965   xleft:0;
12966   xxalign:right;
12967   display:block;
12968   border:4px inset rgba(255,255,255,0.1);
12969   background-color:rgba(255,255,255,0.1);
12970 }
12971 .AffView:hover {
12972   z-index:1;
12973   width:300px;
12974   overflow:scroll;
12975   border:4px inset #fcc;
12976   background-color:rgba(80,80,255,0.2);
12977   background-color:#ffc;
12978 }
12979 .AffPlate:hover {
12980   border-left:4px dashed #888;
12981 }
12982 .AffPlate {
12983   overflow-x:visible;
12984   border-left:4px dashed rgba(255,255,255,0.1);
12985   max-width:2560px;
12986   max-height:2880px;
12987   margin-top:10px;
12988   margin-bottom:10px;
12989   margin-left:4px;
12990   width:300px;
12991   xheight:1440px;
12992 }
12993 .AffItem {
12994   overflow-x:visible;
12995   xoverflow-y:scroll;
12996   max-width:2560px;
12997   max-height:1440px;
12998   z-index:0;
12999   display:block;
13000   position:fixed;
13001   xposition:absolute;
13002   position:relative;
13003   //left:300px;
13004   xresize:both;
13005   padding:0px;
13006   width:600px;
13007   height:400px;
13008   max-height:800px !important;
13009   margin-top:0%;
13010   margin-left:0%;
13011   margin-right:0% !important;
13012   border:16px inset #ccc;
13013   transform:scale(0.5);
13014   background-color:rgba(255,255,255,0.2);
13015   xxalign:right;
13016 }
13017 .AffItem:hover {
13018   border:16px inset #bbf;
13019 }
13020 </style>
13021 <input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13022 <input id="Affiliate_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13023 <input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13024 <span id="Affiliate_WorkCodeView"></span>
13025 <script id="Affiliate_WorkCodeView">
13026 function Affiliate_openWorkCodeView(){
13027   function Affiliate_showWorkCode(){
13028     showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
13029   }
13030   Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
13031 }
13032 Affiliate_openWorkCodeView(); // should be invoked by an event
13033
13034 <!--iframe id="aff_0" xxsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
13035 <!--iframe id="aff_1" xxsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
13036 var Aff_IsSetup = false;
13037 Affiliate_Test.addEventListener('click',Aff_Setup);
13038 function Aff_Setup(){
13039   if (Aff_IsSetup) { return; } Aff_IsSetup = true;
13040   parent = document.documentElement;
13041   parent.appendChild(AffView);
13042   AffView.style.top = '0px';
13043   AffView.style.left = (window.innerWidth - 280) + 'px';
13044   var off = 100;
13045   zoom = 0.5;
13046   ozoom = 0.3;
13047   leftx = window.innerWidth - 300;
13048   left = leftx + 'px';
13049   left = -130 + 'px';
13050   console.log('aff-init window.innerWidth='+window.innerWidth);
13051   w = 1000;
13052   h = 560;
13053   aff_0.src = './gshell/gsh.go.html';
13054   aff_1.src = 'https://golang.org';
13055   aff_2.src = 'https://drafts.csswg.org/';
13056   aff_3.src = 'https://html.spec.whatwg.org/dev/';
13057   aff_4.src = 'https://wikipedia.org';
13058   aff_5.src = 'https://www.bing.com/translator';
13059   //parent.appendChild(aff_0);
13060   aff_0.style.width = zoom*w+'px';
13061   aff_0.style.height = zoom*h+'px';
13062   aff_0.style.left = left;
13063   //aff_0.style.top = off+'px'; off += ozoom*h;
13064   aff_0.draggable = 'true';
13065   //parent.appendChild(aff_1);
13066   aff_1.style.width = zoom*w+'px';
13067   aff_1.style.height = zoom*h+'px';
13068   aff_1.style.left = left;
13069   //aff_1.style.top = off+'px'; off += ozoom*h;
13070   aff_1.style.top = -150px;
13071   aff_1.draggable = 'true';
13072   //parent.appendChild(aff_2);
13073   aff_2.style.width = zoom*w+'px';
13074   aff_2.style.height = zoom*h+'px';
13075   aff_2.style.left = left;
13076   //aff_2.style.top = off+'px'; off += ozoom*h;
13077   aff_2.style.top = -300px;
13078   aff_2.draggable = 'true';
13079   //parent.appendChild(aff_3);
13080   aff_3.style.transform = 'scale(0.25)';
13081   aff_3.style.width = 2*zoom*w+'px';
13082   aff_3.style.height = 2*zoom*h+'px';
13083   aff_3.style.border = '2px inset #ccc';
13084   //aff_3.style.left = -390 + 'px'; //left*2;
13085   //aff_3.style.left = (leftx - 265) + 'px';
13086   aff_3.style.left = -395 + 'px';
13087   //aff_3.style.top = (-155+off)+'px'; off += ozoom*h;
13088   aff_3.style.top = -600px;
13089   aff_3.draggable = 'true';
13090 }

```

```

13098
13099 //parent.appendChild(aff_4);
13100 aff_4.style.width = zoom*h+'px';
13101 aff_4.style.height = zoom*h+'px';
13102 aff_4.style.left = left;
13103 //aff_4.style.top = off+'px'; off += ozoom*h;
13104 aff_4.style.top = -900px;
13105 aff_4.draggable = 'true';
13106
13107 //parent.appendChild(aff_5);
13108 aff_5.style.transform = "scale(0.300)";
13109 aff_5.style.width = zoom*(w*1.67)+'px';
13110 aff_5.style.height = zoom*(h*1.67)+'px';
13111 aff_5.style.border = "2px inset #ccc";
13112 aff_5.style.left = -308+'px';
13113 //aff_5.style.left = (-175+leftx)+'px';
13114 //aff_5.style.top = (-95+off)+'px'; off += ozoom*h;
13115 //aff_5.style.left = '0px';
13116 //aff_5.style.top = '0px';
13117 aff_5.style.top = "-1150px";
13118 //aff_5.style-align = "right";
13119 aff_5.draggable = 'true';
13120
13121 window.addEventListener('resize',affresize);
13122}
13123function affresize(){
13124 AffView.style.left = (window.innerWidth - 280) + 'px';
13125 leftx = window.innerWidth - 400;
13126 left = leftx + 'px';
13127 console.log('aff-resize window.innerWidth='+window.innerWidth);
13128}
13129//window.addEventListener('resize',affresize);
13130//document.addEventListener('resize',affresize);
13131//gsh.addEventListener('resize',affresize);
13132
13133function ResetAffView(){
13134 AffViewDock.appendChild(AffView);
13135 AffView.removeAttribute('style');
13136 aff_0.removeAttribute('src');
13137 aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
13138 aff_1.removeAttribute('src');
13139 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
13140 aff_2.removeAttribute('src');
13141 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
13142 aff_3.removeAttribute('src');
13143 aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
13144 aff_4.removeAttribute('src');
13145 aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
13146 aff_5.removeAttribute('src');
13147 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
13148}
13149</script>
13150</details>
13151<!-- Affiliate WorkCodeSpan -->
13152*/ </span>
13153<!-- ===== Work } ===== -->
13154
13155
13156
13157<!-- ===== Work { ===== -->
13158<span id="TextCanvas_WorkCodeSpan">
13159*
13160<details id="TextCanvas_Section"><summary id="TextCanvas_Summary">TextCanvas</summary>
13161<!-- ===== TextCanvas // 2020-1013 status { -->
13162
13163<details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selection</summary>
13164<h2>Font Selection</h2>
13165<div id="FontList"></div>
13166</div>
13167<style>
13168#FontList {
13169 overflow:visible;
13170 background-color:rgba(240,245,255,1.0) !important;
13171}
13172#FontList td {
13173 font-size:16px;
13174 padding:0px;
13175 padding-left:2px;
13176 padding-right:2px;
13177 margin:0px;
13178 line-height:1.2;
13179 border:0px;
13180}
13181#FontList td:hover {
13182 background-color:#228;
13183}
13184#FontList tr:hover {
13185 color:#fff;
13186 background-color:#000;
13187 xborder:1px solid #000;
13188}
13189.xcourier { colr:#000; font-size:16px; font-family:courier; }
13190.xcursive { colr:#000; font-size:16px; font-family:cursive; }
13191.xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
13192.xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
13193.xmonospace { colr:#000; font-size:16px; font-family:monospace; }
13194</style>
13195<script>
13196function fontstr(name,text){
13197 //tr = '<tr style=\'font-family:"'+name+'";\'>\n';
13198 tr = '<tr style=\'font-family:"'+name+'";\'>\n';
13199 tr += '<td data-fsty="font-family:Arial;font-size:12pt;">'+name+'</td>';
13200 tr += '<td data-fsty="n">'+text+'</td>';
13201 tr += '<td data-fsty="b">'+text+'</td>';
13202 tr += '<td data-fsty="i">'+text+'</td>';
13203 tr += '<td data-fsty="u">'+text+'</td>';
13204 tr += '<td data-fsty="b">'+text+'</td>';
13205 tr += '</tr>';
13206 return tr;
13207}
13208function lfont(){
13209 text = 'GShell-Go012';
13210
13211 fl = '';
13212 fl += '<table>\n';
13213 fl += fontstr('Arial',text);
13214 fl += fontstr('Courier',text);
13215 fl += fontstr('Courier New',text);
13216 fl += fontstr('Georgia',text);
13217 fl += fontstr('Helvetica',text);
13218 fl += fontstr('Verdana',text);
13219 fl += fontstr('Times',text);
13220
13221 fl += fontstr('Osaka',text);
13222 fl += fontstr('Meiryo',text);
13223 fl += fontstr('YuMincho',text);
13224
13225 //fl += fontstr('Roman',text);
13226 //document.fonts.load("30px cursive");
13227 fl += fontstr('Serif',text);
13228 fl += fontstr('Sans-Serif',text);
13229 fl += fontstr('System-UI',text);
13230 fl += fontstr('Monospace',text);
13231 fl += fontstr('Cursive',text);
13232 fl += fontstr('Fantasy',text);
13233 fl += '</table>\n';
13234
13235 if( false ){
13236 fss = document.fonts.entries(); // FontFaceSet
13237 console.log('FSS='+fss);
13238 while( true ){
13239 font = fss.next();
13240 if( font.done ){
13241 break;
13242 }
13243 fl += font.value[0] + '<br>';
13244 }
13245 }
13246 FontList.innerHTML = fl;
13247}
13248function selectFont(e){
13249 t = e.target;
13250 let fsty = '';
13251 for( i = 0; i < 4; i++){
13252 //console.log('FontSelect '+t.nodeName+' #' + t.id+' '+t.style);
13253 if( t.nodeName == 'TD' ){
13254 //console.log('FontSelect '+t.outerHTML);
13255 if( t.hasAttribute('data-fsty') ){
13256 fsty = t.getAttribute('data-fsty');
13257 //console.log('FontSelect = '+ fsty);
13258 }
13259 }
13260 if( t.nodeName != 'TD' ){
13261 if( t.style != '' ){
13262 if( t.style.fontFamily != '' ){
13263 break;
13264 }
13265 }
13266 }
13267 t = t.parentNode;
13268 }
13269 if( t.style != '' ){
13270 font = t.style.fontFamily;
13271 //console.log('FontSelect: '+font);
13272 //console.log('FontSelect == '+ fsty);
13273 if( font != '' ){
13274 sel = document.getElementById("TextCanvas_1_Font");

```

```

13275     if( sel != null ){
13276         if( fsty != ' ' ){
13277             TextCanvas_1.Bold.checked = 0 <= fsty.indexOf('b');
13278             TextCanvas_1.Italic.checked = 0 <= fsty.indexOf('i');
13279         }
13280         sel.value = font;
13281         RedrawTextCanvas();
13282     }else{
13283         alert('Event: ' + e.target.nodeName + ' #' + font);
13284     }
13285 }
13286 }
13287 }
13288 }
13289 FontList.addEventListener('click',selectFont);
13290 document.fonts.onloadingdone = function(fsse){
13291     //alert('font-loaded '+fsse.fontfaces.length);
13292 }
13293 function FontList_Setup(){
13294     if( FontSelect_Summary.open ){
13295         lsfont();
13296     }
13297 }
13298 FontSelect_Summary.addEventListener('click',lsfont);
13299 </script>
13300 </details>
13301
13302 <h2>Drawing Text on Canvas</h2>
13303 <!-- 2020-1012 -- Drawing Text on Canvas // SatoxITS { -->
13304 <div id="TextCanvas_1_Panel" class="CanvasLabel">
13305 <input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
13306 <input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
13307 <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
13308 <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">Bold
13309 <input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">Italic
13310 <br>
13311 <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
13312 <input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">
13313 <!-- to be P216 series ? -->
13314 <p>
13315 <input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
13316 </p>
13317 </div>
13318 <p>
13319 <canvas id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></canvas>
13320 </p>
13321 <div class="CanvasLabel">
13322 <input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
13323 <input id="TextCanvas_1_ToPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
13324 <input id="TextCanvas_1_ToJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
13325 <input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
13326 <div class="CanvasInImage"><br><img id="TextCanvas_1_Image" class="CanvasImage" src="">(inline image)</div>
13327 <div id="TextCanvas_1_BgImage" class="CanvasInImage"><br>(background image)</div>
13328 (Data URL)
13329 <div id="TextCanvas_1_DataUrlView" class="DataUrlView"><span id="TextCanvas_1_DataUrlText"></span></div>
13330 </div>
13331 <style>
13332 .CommandUsageText {
13333     font-family:Courier New;
13334 }
13335 .TextCanvas {
13336     border:1px solid #000;
13337     resize:both;
13338     display:inline !important;
13339 }
13340 .DataUrlView {
13341     width:100%;
13342     font-size:10pt;
13343     font-family:Courier New, monospace;
13344     color:#000;
13345     xbackground-color:#eee;
13346     margin-bottom:10px;
13347     xdisplay:block;
13348     xoverflow:scroll;
13349 }
13350 .CanvasImage {
13351     border:1px dashed #000;
13352 }
13353 .TextCanvasText {
13354     font-size:12pt;
13355     width:100%;
13356 }
13357 .CanvasLabel {
13358     font-size:10pt;
13359     color:#000;
13360 }
13361 .CanvasInImage {
13362     width:100%;
13363     height:160px;
13364     margin-bottom:10px;
13365     color:rgba(32,160,32,0.5);
13366     text-shadow:3px 3px #eee;
13367     background-color:#eee;
13368     xborder:1px solid #000;
13369     font-size:18pt;
13370     vertical-align:middle;
13371 }
13372 .PanelRadio {
13373     font-size:12pt !important;
13374     color:#000 !important;
13375     vertical-align:middle;
13376 }
13377 .CanvasBox {
13378     vertical-align:middle;
13379     margin-left:4px !important;
13380     margin-right:2px !important;
13381 }
13382 .CanvasPanel {
13383     vertical-align:middle !important;
13384     height:14pt !important;
13385     width:inherit !important;
13386     padding:2px !important;
13387     margin:4px !important;
13388     margin-left:4px !important;
13389     margin-right:2px !important;
13390     font-size:10pt !important;
13391     font-family:Georgia !important;
13392     color:#000;
13393     display:inline !important;
13394 }
13395 .TextCanvasPanel {
13396     vertical-align:middle;
13397     font-size:10pt !important;
13398     font-family:Georgia !important;
13399     color:#000;
13400     display:inline !important;
13401 }
13402 .PanelButton {
13403     font-size:10pt !important;
13404     font-family:Georgia !important;
13405     vertical-align:middle;
13406     width:45pt !important;
13407     height:14pt !important;
13408     line-height:1.2 !important;
13409     padding:3px !important;
13410     margin:1px !important;
13411     display:inline !important;
13412     padding:1px !important;
13413     color:#fff !important;
13414     background-color:#228 !important;
13415 }
13416 </style>
13417 <script>
13418 function DrawTextCanvas(){
13419     ctx = TextCanvas_1.getContext('2d');
13420     var textfont = " ";
13421     if( TextCanvas_1.Italic.checked ) textfont += ' italic';
13422     if( TextCanvas_1.Bold.checked ) textfont += ' bold';
13423     textfont += " " + TextCanvas_1_Size.value + "px";
13424     textfont += " " + TextCanvas_1_Font.value;
13425     //ctx.font = "italic bold 64px Georgia";
13426     //console.log("TxFont="+textfont);
13427     ctx.fillStyle = TextCanvas_1_Color.value; //'#22a';
13428     ctx.font = textfont;
13429     ctx.fillText(TextCanvas_1_Text.value,10,80);
13430 }
13431 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
13432 function ClearTextCanvas(){
13433     cv = TextCanvas_1;
13434     ctx = cv.getContext('2d');
13435     ctx.clearRect(0,0,cv.width,cv.height);
13436 }
13437 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
13438 function RedrawTextCanvas(){
13439     ClearTextCanvas();
13440     DrawTextCanvas();
13441 }
13442 function ab2str(buf) {
13443     return String.fromCharCode.apply(null, new Uint16Array(buf));
13444 }
13445 }
13446 }
13447 // 2020-1024, canvas to image
13448 function CanvasToImage(){
13449     canvas = TextCanvas_1;
13450     // https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
13451     if( TextCanvas_1_ToPNG.checked ) {

```

```

13452     url = canvas.toDataURL("image/png");
13453 }else{
13454     url = canvas.toDataURL("image/jpeg",1.0);
13455 }
13456 //alert('CanvasToImage: length='+url.length+'\n'+url);
13457 TextCanvas_1_Image.src = url;
13458 TextCanvas_1_Image.style.backgroundColor = 'url('+url+')';
13459 if( TextCanvas_1_DataURL.checked ){
13460     //TextCanvas_1_DataURLView.innerHTML = url;
13461     txa = TextCanvas_1_DataURLText;
13462     utxa = document.createElement('textArea');
13463     utxa.id = "TextCanvas_1_DataURLText";
13464     utxa.style.width = '100%';
13465     utxa.style.height = '50pt';
13466     utxa.value = url;
13467     txa.parentNode.replaceChild(utxa,txa);
13468 }
13469 return TextCanvas_1_Image;
13470
13471 var image = new Image();
13472 image.src = url;
13473 urla = str2ab(url);
13474 blob = new Blob(urla,{type:'text/plain'});
13475 link = document.createElement('a');
13476 link.href = URL.createObjectURL(blob);
13477 link.download = 'character.txt';
13478 link.click();
13479 return image;
13480
13481 TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
13482
13483 if( TextCanvas_Section.open ){
13484     DrawTextCanvas();
13485 }
13486 TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
13487 </script>
13488 <!-- } -->
13489
13490 <script>
13491 //TextCanvas_1_Panel.addEventListener('mouseover',OffGJShell);
13492 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
13493 </script>
13494 <!-- Clicking the textarea is necessary to see upto the end of text. why? -->
13495 <input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13496 <input id="TextCanvas_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13497 <input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13498 <span id="TextCanvas_WorkCodeView"></span>
13499 <script id="TextCanvas_WorkScript">
13500 function TextCanvas_openWorkCodeView(){
13501     function TextCanvas_showWorkCode(){
13502         showHtmlCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
13503     }
13504     TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
13505 }
13506 TextCanvas_openWorkCodeView(); // should be invoked by an event
13507 </script>
13508 </details>
13509 <!-- TextCanvas_WorkCodeSpan } -->
13510 *// </span>
13511 //<!-- ===== Work } ===== -->
13512
13513
13514 <!-- ===== Work { ===== -->
13515 </span id="Shading_WorkCodeSpan">
13516 /*
13517 <details><summary>Shading Canvas</summary>
13518 <!-- ===== Shading Canvas // 2020-1011 SatorITS { -->
13519 <h2>Shading Canvas</h2>
13520 <note class="CommandUsageText">
13521 <b>Commands</b><br>
13522 Placement Node<br>
13523 a ... apply (into absolute position)<br>
13524 j ... bring down (ArrowDown)<br>
13525 k ... bring up (ArrowUp)<br>
13526 h ... bring left (ArrowLeft)<br>
13527 l ... bring right (ArrowRight)<br>
13528 0 ... z-index = 0<br>
13529 + ... z-index += 1<br>
13530 - ... z-index -= 1<br>
13531 r ... return to here (relative position)<br>
13532 c ... clear the log text<br>
13533 Note: the HTML text must be contenteditable to catch Key Event.<br>
13534 </note>
13535
13536
13537 <div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
13538 <div id="Shading_1_Html" class="ShadingHtml" draggable="true"></div>
13539 <div id="Shading_1_Log" class="ShadingLog" draggable="true" style=""></div>
13540
13541 <canvas id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</canvas></div>
13542 <style>
13543 .ShadingPlate {
13544     z-index:0;
13545     position:static;
13546     overflow:scroll;
13547     display:block;
13548     width:100%;
13549     height:400px;
13550     font-size:9pt;
13551     font-family:Courier New;
13552     border:1px dashed #000;
13553     color:#444;
13554 }
13555 .ShadingLog {
13556     z-index:0;
13557     position:relative;
13558     display:block;
13559     top:0px;
13560     left:0px;
13561     overflow:scroll;
13562     width:100%;
13563     font-size:9pt;
13564     font-family:Courier New;
13565     color:#666;
13566     overflow:scroll;
13567     background:rgba(200,255,200,0.4);
13568     height:400px;
13569 }
13570 .ShadingHtml {
13571     z-index:2;
13572     position:relative;
13573     display:block;
13574     top:0px;
13575     left:0px;
13576     overflow:scroll;
13577     width:100%;
13578     font-size:12pt;
13579     font-family:Courier New;
13580     color:#666;
13581     overflow:scroll;
13582     background:rgba(200,255,200,0.4);
13583     height:400px;
13584 }
13585 .ShadingCanvas {
13586     z-index:3;
13587     position:relative;
13588     xdisplay:block;
13589     top:0px;
13590     left:100px;
13591     resize:both;
13592     border:1px solid #000;
13593 }
13594 </style>
13595 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13596 <input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13597 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13598 <span id="Shading_WorkCodeView"></span>
13599 <script id="Shading_WorkScript">
13600 function Shading_openWorkCodeView(){
13601     function Shading_showWorkCode(){
13602         showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
13603     }
13604     Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
13605 }
13606 const BR = '<'+'>';
13607 Shading_openWorkCodeView(); // should be invoked by an event
13608 function sh_onClick(e){
13609     Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
13610     + ' offset('+e.offsetX+', '+e.offsetY+')'
13611     + ' client('+e.clientX+', '+e.clientY+')'
13612     + ' page('+e.pageX+', '+e.pageY+')'
13613     + ' screen('+e.screenX+', '+e.screenY+')'
13614     +BR;
13615     e.stopPropagation();
13616     e.preventDefault();
13617 }
13618 function sh_onKeyUp(e){
13619     if( Shading_1.style.zIndex == '' ){
13620         Shading_1.style.zIndex = 0;
13621     }
13622     zi = parseInt(Shading_1.style.zIndex);
13623
13624     if( e.key.length == 1 ){
13625         Shading_1_Html.innerHTML += e.key;
13626     }
13627
13628     if( e.key == '0' ){ zi = 0; }else

```

```

13629 if( e.key == '+' ) { zi += 1; }else
13630 if( e.key == '-' ) { zi -= 1; }else
13631 if( e.key == 'c' ) {
13632   Shading_1_Log.innerHTML = '';
13633 }else
13634 if( e.key == 'r' ) {
13635   Shading_1.style.position = "relative";
13636   Shading_1.style.top = '0px';
13637   Shading_1.style.left = '0px';
13638   zi = 0;
13639 }else
13640 if( e.key == 'j' || e.code == 'ArrowDown' ){
13641   topx = parseInt(Shading_1.style.top) + 50;
13642   Shading_1.style.top = topx + 'px';
13643 }else
13644 if( e.key == 'k' || e.code == 'ArrowUp' ){
13645   topx = parseInt(Shading_1.style.top) - 50;
13646   Shading_1.style.top = topx + 'px';
13647 }else
13648 if( e.key == 'l' || e.code == 'ArrowRight' ){
13649   lefty = parseInt(Shading_1.style.left) + 50;
13650   Shading_1.style.left = lefty + 'px';
13651 }else
13652 if( e.key == 'h' || e.code == 'ArrowLeft' ){
13653   lefty = parseInt(Shading_1.style.left) - 50;
13654   Shading_1.style.left = lefty + 'px';
13655 }else
13656 if( e.key == 'a' ) {
13657   Shading_1.style.position = "absolute";
13658   Shading_1.style.top = '0px';
13659   Shading_1.style.left = '0px';
13660 }else{
13661 }
13662 Shading_1.style.zIndex = zi;
13663 Shading_1_Log.innerHTML += 'Keyup..' + e.target.nodeName + '#' + e.target.id
13664   + 'Up(' + e.key + '/' + e.code + ')';
13665   + 'z-index:' + zi + '/' + Shading_1.style.zIndex
13666   + 'top:' + Shading_1.style.top
13667   + BR;
13668 e.stopPropagation();
13669 e.preventDefault();
13670 }
13671 function sh_onKeyDown(e){
13672   Shading_1_Log.innerHTML += 'Keydown' + e.target.nodeName + '#' + e.target.id
13673     + 'Down(' + e.key + '/' + e.code + ')'+BR;
13674   e.stopPropagation();
13675   e.preventDefault();
13676 }
13677 function Shading_Setup(){
13678   Shading_1_Log.innerHTML += '<+h4>Click here<+/'h4>';
13679   Shading_1.addEventListener('keydown', sh_onKeyDown);
13680   Shading_1.addEventListener('keyup', sh_onKeyUp);
13681   Shading_1.addEventListener('click', sh_onClick);
13682   Shading_1.addEventListener('click', sh_onClick);
13683 }
13684 Shading_1_Log.style.top = "-400px";
13685 Shading_1_Log.style.left = "200px";
13686
13687 Shading_1.appendChild(Shading_1_Canvas);
13688 Shading_1_Canvas.style.width = "300px";
13689 Shading_1_Canvas.style.height = "300px";
13690 Shading_1_Canvas.style.position = "relative";
13691 Shading_1_Canvas.style.top = "-750px";
13692 Shading_1_Canvas.style.left = "100px";
13693
13694 const ctx = Shading_1_Canvas.getContext('2d');
13695 ctx.fillStyle = "rgba(160,0,0,0.9)";
13696 ctx.fillRect(50,50,40,40);
13697 ctx.fillStyle = "rgba(0,160,0,0.9)";
13698 ctx.fillRect(60,60,40,40);
13699 ctx.fillStyle = "rgba(0,0,160,0.9)";
13700 ctx.fillRect(70,70,40,40);
13701 }
13702 function Reset_ShadingCanvas(){
13703   Shading_1_Log.removeAttribute('style');
13704   Shading_1_Log.innerHTML = '';
13705   Shading_1_Canvas.style = {};
13706   //Shading_1_Canvas.removeAttribute('style');
13707 }
13708 </script>
13709 </details>
13710 <!-- Shading_WorkCodeSpan -->
13711 *//</span>
13712 <!-- Work -->
13713
13714
13715
13716 <!-- Work { -->
13717 <span id="Charmap_WorkCodeSpan">
13718 *
13719 <details id="Charmap_Work"><summary>Character Map Mandala</summary>
13720 <!-- CharcodeCharmap // 2020-1016 SatoxIFS ( -->
13721 <h2>Unicode Character Map</h2>
13722 <note>code 0x0000 - 0xFFFF / 16px / 3200 x 3200 px / zoom:0.25</note>
13723 <div id="Charmap_1_Frame">
13724 <div id="Charmap_1_Text" class="Charmap">
13725 </div>
13726 </div>
13727 <br>
13728 <style>
13729 #Charmap_1_Frame {
13730   overflow:scroll;
13731   height:3200px; width:3200px;
13732   xtransform:scale(0.5);
13733   zoom:0.25;
13734   resize:both;
13735   background-color:#fff;
13736 }
13737 .Charmap {
13738   xzoom:0.25;
13739   font-size:16px;
13740   line-height:1.0;
13741   xfont-family:Georgia;
13742   color:#000;
13743 }
13744 </style>
13745 <script>
13746 function charmappen(){
13747   text = '';
13748   for( cc = 0; cc < 0x10000; cc++ ){
13749     text += String.fromCharCode(cc);
13750   }
13751   Charmap_1_Text.innerHTML = text;
13752 }
13753 </script>
13754 Charmap_Work.addEventListener('click', charmappen);
13755 </CharmapSpan>
13756 </script>
13757
13758 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13759 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13760 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13761 <span id="Charmap_WorkCodeView"></span>
13762 <script id="Charmap_WorkScript">
13763 function Charmap_showWorkCode(){
13764   showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
13765 }
13766 Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCode);
13767 </script>
13768 Charmap_openWorkCodeView(); // should be invoked by an event
13769 </script>
13770 </details>
13771 <!-- Charmap_WorkCodeSpan -->
13772 *//</span>
13773 <!-- Work -->
13774
13775
13776
13777 <!-- Work { -->
13778 <span id="Pointillism_WorkCodeSpan">
13779 *
13780 <details><summary>Collaborated Pointillism</summary>
13781 <!-- CollaboratedPointillism // 2020-1016 SatoxIFS ( -->
13782 <h2><a name="Pointillism_Share">Pointillism</a><a href="#Pointillism">Collaborated Pointillism</a></h2>
13783
13784 <input type="button" class="HtmlCodeViewButton" value="Share"> Share
13785 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
13786 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
13787 <input type="button" class="HtmlCodeViewButton" value="Replay">
13788 <input type="button" class="HtmlCodeViewButton" value="Repeat">
13789 <input type="button" class="HtmlCodeViewButton" value="Save">
13790 <input type="button" class="HtmlCodeViewButton" value="Load">
13791 <div id="Pointillism_1" class="Pointillism">
13792
13793 <span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
13794 <span id="Pointillism_1_XY_1" class="Pointillism_XY">XY1</span>
13795 <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
13796 <canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13797 </span>
13798
13799 <span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
13800 <span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
13801 <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
13802 <canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13803 </span>
13804 </div>

```

```

1380<br>
1380<style>
13809.Pointillism {
13810  display:block;
13811  resize:both;
13812  width:680px;
13813  height:380px;
13814  min-width:240px;
13815  min-height:270px;
13816  background-color:#eee;
13817  overflow:scroll;
13818  font-size:16px;
13819  font-family:Georgia;
13820  color:#000;
13821  vertical-align:middle;
13822 }
13823.Pointillism Unit {
13824  position:relative;
13825  top:0px;
13826  display:block;
13827  overflow:scroll;
13828  width:300px;
13829  height:350px;
13830  margin:5px;
13831  padding:10px;
13832  background-color:rgba(255,255,127,0.7);
13833 }
13834.Pointillism XY {
13835  display:block;
13836  vertical-align:middle;
13837  width:290px;
13838  xxheight:20px;
13839  font-size:12px;
13840  line-height:1.2;
13841  padding:5px;
13842  margin:0px;
13843  color:#fff;
13844  background-color:#44c;
13845 }
13846.Pointillism XY Remote {
13847  display:block;
13848  vertical-align:middle;
13849  width:290px;
13850  xxheight:20px;
13851  font-size:12px;
13852  line-height:1.2;
13853  padding:5px;
13854  color:#fff;
13855  background-color:#4a4;
13856 }
13857.Pointillism Canvas {
13858  display:block;
13859  position:relative;
13860  xpadding:20px;
13861  xleft:20px;
13862  xtop:20px;
13863  background-color:#333;
13864 }
13865</style>
13866<script>
13867 var points = [];
13868 var replay = [];
13869 var replayx = 0;
13870 function pClearCanvas(can){
13871  ctx = can.getContext('2d');
13872  ctx.clearRect(0,0,can.width,can.height);
13873 }
13874 function Pointillism_1_ClearCanvas(){
13875  pClearCanvas(Pointillism_1_Canvas_1);
13876  pClearCanvas(Pointillism_1_Canvas_2);
13877 }
13878 function PointsReset(){
13879  points = [];
13880  replay = [];
13881  inRepeat = false;
13882  inReplay = false;
13883  Pointillism_1_ClearCanvas();
13884 }
13885 function Pointillism_1_ResetCanvas(){
13886  PointsReset();
13887  if( Pointillism_1_Share.checked ){
13888    //alert('---Broad cast reset\n');
13889    GJ_BcastMessage("DRAW RESET");
13890  }
13891 }
13892 function Pointillism_1_ResetCanvasReceive(){
13893  //alert('---received reset\n');
13894  PointsReset();
13895 }
13896 function drawPoint(can,x,y,r,g,b){
13897  const ctx = can.getContext('2d');
13898  ctx.fillStyle = 'rgba('+r+', '+g+', '+b+', 0.7)';
13899  ctx.fillRect(x,y,8,8);
13900 }
13901 function waitMs(serno,ms){
13902  console.log('-- wait #' +serno+ ' '+ms+'ms');
13903  until = new Date();
13904  now = until.getTime();
13905  untilMs = now + ms;
13906  for( wi = 0; ; wi++){
13907    now = new Date();
13908    nowMs = now.getTime();
13909    remMs = untilMs - nowMs;
13910    //console.log('wait '+wi+': '+remMs+'/' +ms);
13911    if( remMs < 0 ){
13912      break;
13913    }
13914  }
13915 }
13916 var inReplay = false;
13917 function replay1(){
13918  rx = replayx;
13919  if( replay.length <= rx ){
13920    return;
13921  }
13922  replayx += 1;
13923  pl = replay[rx];
13924  if( pl[1] == 1 ){
13925    can = Pointillism_1_Canvas_1;
13926  }else{
13927    can = Pointillism_1_Canvas_2;
13928  }
13929  drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
13930  if( inReplay == false ){
13931    console.log('wait '+replayx+' Stopped');
13932    return;
13933  }
13934  if( rx < replay.length-1 ){
13935    prevMs = replay[rx][0].getTime();
13936    nextMs = replay[rx+1][0].getTime();
13937    delayMs = nextMs - prevMs;
13938    //console.log('wait '+replayx+' '+delayMs+'ms');
13939    window.setTimeout(replay1,delayMs);
13940  }else{
13941    console.log('wait '+replayx+' Finished');
13942    if( inRepeat ){
13943      window.setTimeout(repeat1,1000);
13944    }
13945  }
13946 }
13947 function Pointillism_1_ReplayCanvas(can){
13948  Pointillism_1_ClearCanvas();
13949  replay = points;
13950  replayx = 0;
13951  inReplay = true;
13952  replay1();
13953 }
13954 var inRepeat = false;
13955 function repeat1(){
13956  Pointillism_1_ClearCanvas();
13957  replay = points;
13958  replayx = 0;
13959  replay1();
13960  if( inRepeat ){
13961    //window.setTimeout(repeat1,1000);
13962  }
13963 }
13964 function Pointillism_1_RepeatCanvas(can){
13965  if( inRepeat ){
13966    inRepeat = false;
13967    inReplay = false;
13968  }else{
13969    inRepeat = true;
13970    inReplay = true;
13971    repeat1();
13972  }
13973 }
13974
13975 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
13976 function Pointillism_Setup(){
13977  var moveCount1 = 0;
13978  var moveCount2 = 0;
13979
13980  var gjlinked = false;
13981  function GJdraw(msg){
13982    if( gjlinked == false ){

```

```

13983 //GJLink_Section.open = true;
13984 GJ Join();
13985 gjlinked = true;
13986 }
13987 GJ_BcastMessage('DRAW '+msg);
13988 }
13989 function showXY1(e){
13990 moveCount1 += 1;
13991 x = e.offsetX;
13992 y = e.offsetY;
13993 Pointillism_1_XY_1.innerHTML = 'XY1: ' + 'x'+x + ', y'+y + ' '+moveCount1+'/'+points.length;
13994 Pointillism_1_XY_2.Remote.innerHTML = 'XY1: ' + 'x'+x + ', y'+y + ' '+moveCount1;
13995 if (e.buttons || CopyLocal()) {
13996 points.push([new Date(),x,y,64,64,255]);
13997 drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
13998 if( CopyLocal() ){
13999 drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
14000 }
14001 GJdraw('1','x+', 'y');
14002 }
14003 }
14004 function showXY2(e){
14005 moveCount2 += 1;
14006 x = e.offsetX;
14007 y = e.offsetY;
14008 Pointillism_1_XY_2.innerHTML = 'XY2: ' + 'x'+x + ', y'+y + ' '+moveCount2+'/'+points.length;
14009 Pointillism_1_XY_1.Remote.innerHTML = 'XY2: ' + 'x'+x + ', y'+y + ' '+moveCount1;
14010 if (e.buttons || CopyLocal()) {
14011 points.push([new Date(),x,y,64,255,64]);
14012 drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
14013 if( CopyLocal() ){
14014 drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
14015 }
14016 GJdraw('2','x+', 'y');
14017 }
14018 }
14019 Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
14020 Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);
14021 Pointillism_1_Unit_2.style.left = '340px';
14022 Pointillism_1_Unit_2.style.top = '-375px';
14023 }
14024 function Pointillism_RemoteDraw(arg){
14025 //alert('Draw at '+arg);
14026 //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
14027 if( arg == 'RESET' ){
14028 Pointillism_1_ResetCanvasReceive();
14029 }else{
14030 argv = arg.split(',');
14031 x = argv[1];
14032 y = argv[2];
14033 Pointillism_1_XY_2.Remote.innerHTML = 'XYR: ' + 'x'+x + ', y'+y + ' '+points.length;
14034 drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
14035 }
14036 }
14037 </script>
14038
14039
14040 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14041 <input id="Pointillism_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14042 <input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14043 <script id="Pointillism_WorkScript">
14044 function Pointillism_openWorkCodeView(){
14045 function Pointillism_showWorkCode(){
14046 showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
14047 }
14048 Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_showWorkCode);
14049 }
14050 Pointillism_openWorkCodeView(); // should be invoked by an event
14051 </script>
14052 </details>
14053 <!-- Pointillism_WorkCodeSpan -->
14054 * //</span>
14055 <!-- Work -->
14056
14057
14058
14059 <!-- Work { -->
14060 <span id="StatCounter_WorkCodeSpan">
14061 *
14062 <details><summary>StatCounter</summary>
14063 <!-- StatCounter // 2020-1018 SatoxITS { -->
14064 <h2>StatCounter</h2>
14065
14066 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
14067 </div>
14068
14069 <script>
14070 statcounter {
14071 vertical-align:middle;
14072 }
14073 #sc_SatoxITS {
14074 color:#000;
14075 font-size:12pt;
14076 height:30px;
14077 width:100%;
14078 background-color:#ddd;
14079 }
14080 </style>
14081
14082 <div>
14083 <script>
14084 var sc_project=12411639;
14085 var sc_invisible=0;
14086 var sc_security="1aeb2a3a";
14087 var sc_https=1;
14088 var scJsHost = "https://";
14089 </script>
14090 <!-- script src="https://statcounter.com/counter/counter.js" -->
14091 <!-- /script --> (counter by inline script)
14092 </div>
14093
14094 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14095 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14096 <input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14097 <script id="StatCounter_WorkScript">
14098 function StatCounter_openWorkCodeView(){
14099 function StatCounter_showWorkCode(){
14100 showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
14101 }
14102 StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
14103 }
14104 StatCounter_openWorkCodeView(); // should be invoked by an event
14105 </script>
14106 </details>
14107 <!-- StatCounter_WorkCodeSpan -->
14108 * //</span>
14109 <!-- Work -->
14110
14111
14112
14113
14114
14115
14116
14117
14118 <!-- Work { -->
14119 <span id="Template_WorkCodeSpan">
14120 *
14121 <details><summary>Work Template</summary>
14122 <!-- Template of Work // 2020-0928 SatoxITS { -->
14123 <h2>Template of Work</h2>
14124 <input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14125 <input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14126 <input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14127 <script id="Template_WorkScript">
14128 function Template_openWorkCodeView(){
14129 function Template_showWorkCode(){
14130 showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
14131 }
14132 Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
14133 }
14134 Template_openWorkCodeView(); // should be invoked by an event
14135 </script>
14136 </details>
14137 <!-- Template_WorkCodeSpan -->
14138 * //</span>
14139 <!-- Work -->
14140
14141
14142
14143
14144 <!-- Work { -->
14145 <span id="OriginalSource_WorkCodeSpan">
14146 *
14147 <details open=""><summary>Original Source</summary>
14148 <!-- Original Source of GShell/12P -->
14149 <input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14150 <input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14151 <input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14152 <span id="OriginalSource_TextElement"></span>
14153 <span id="OriginalSource_WorkCodeView"></span>
14154 <script id="OriginalSource_WorkScript">
14155 function OriginalSource_openWorkCodeView(){
14156 function OriginalSource_showWorkCode(){
14157 //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
14158 //OriginalSourceTextElement = OriginalSourceNode;
14159

```



```
14160 //console.log('src3=\n'+OriginalSourceNode.outerHTML);
14161 showHtmlCodeX(OriginalSource_WorkCodeView,OriginalSourceNode,
14162             '\n',
14163             '\n',true);
14164 }
14165 OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_showWorkCode);
14166 }
14167 //OriginalSourceNode = document.documentElement.cloneNode();
14168 //OriginalSourceNode = gsh.cloneNode(true); //=====
14169 //console.log('src0=\n'+document.documentElement.outerHTML);
14170 //console.log('src1=\n'+gsh.outerHTML);
14171 //console.log('src2=\n'+OriginalSourceNode.innerHTML);
14172 OriginalSource_openWorkCodeView(); // should be invoked by an event
14173 //showNodeAsHtmlSource(OriginalSource_WorkCodeView,OriginalSourceNode);
14174 function SaveOriginalNode(){
14175     if( false ){
14176         m0 = performance.memory;
14177         mu0 = m0.usedJSHeapSize;
14178         console.log('-- heap bef clone: '
14179                 +m0.usedJSHeapSize+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
14180     }
14181     OriginalSourceNode = gsh.cloneNode(true);
14182     if( false ){
14183         m1 = performance.memory;
14184         mu1 = m1.usedJSHeapSize;
14185         mu = mu1 - mu0;
14186         //alert('-- clone: used heap '+mu0+' -> '+mu1+' = '+mu+' bytes');
14187         console.log('-- heap aft clone: '
14188                 +m1.usedJSHeapSize+'/'+m1.totalHeapSize+'/'+m1.jsHeapSizeLimit);
14189         //OriginalSourceNode = document.documentElement.cloneNode(true);
14190     }
14191 }
14192 }
14193 }
14194 function Gsh_setupPage(){
14195     GshSetImages();
14196     //Indexer_afterLoaded();
14197     //GShell_initKeyCommands();
14198     //GJConsole_initConsole();
14199     GJConsole_initFactory();
14200     GJLink_init();
14201     InterFrameComm_init();
14202     Gshell_initTopbar();
14203     //Virtualdesktop_init();
14204     Banner_init();
14205     Afi_Setup();
14206     Shading_Setup();
14207     window.setInterval(ShowResourceUsage,1000);
14208     //document.addEventListener('keydown',jgshCommand); // should be applied later?
14209     Pointillism_Setup();
14210     FontList_Setup();
14211     showFooter();
14212     GshInsideIconSetup();
14213     SightClass_Setup();
14214     //spawnPackmonGo();
14215     //PackmonGo_Setup(null);
14216     DrawingCanvas_Setup();
14217 }
14218 }
14219 function OnLoad(){
14220     SaveOriginalNode();
14221     Gsh_setupPage();
14222 }
14223 document.addEventListener('load',Gsh_setupPage);
14224 </script>
14225 </details>
14226 <!-- OriginalSource_WorkCodeSpan -->
14227 * //</span>
14228 //<!-- ===== Work } ===== -->
14229 }
14230 </div>
14231 //<br><script>OnLoad();</script></span>
14232 }
```