

```

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-0.7.4-2020-10-23-SatoxITS</span>
7 <title id="GshTitle">GShell-0.7.4 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="GshBanner" height="100px" onclick="shiftBG();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.7.4 // 2020-10-23 // SatoxITS</note></div>
17 </div>
18 </div>
19 //<-- ----- Work { ----- -->
20 //<span id="Topbar_WorkCodeSpan">
21 /* 
22 <details><summary>Topbar</summary>
23 <!-- Topbar // 2020-1008 SatoxITS ( -->
24 <h2>Topbar</h2>
25 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
26 <input id="Topbar_OpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
27 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
28 <span id="Topbar_WorkCodeView"></span>
29 </details>
30
31 <style>
32 #GshHeading {
33   display:inline;
34   overflow:visible;
35 }
36 .ConfigIcon {
37   position:absolute;
38   top:-6px;
39   left:92px;
40   width:32px;
41   height:32px;
42 }
43 .MetaWindow {
44   z-index:100;
45   position:relative;
46   display:block;
47   overflow:visible !important;
48   width:99.9%;
49   height:122px;
50   top:-22px;
51   left:-22px;
52   border:1px solid #22a;
53   margin:0px;
54   line-height:1.0;
55   font-family:Georgia;
56   color:#fff;
57   font-size:12pt;
58   text-align:center;
59   vertical-align:middle;
60   padding:4px;
61   background-color:rgba(0,8,170,0.8);
62   background-color:#3d861;xxx-PBlue;
63   vertical-align:middle;
64 }
65 .MetaWindow:hover {
66   color:#000;
67   border:1px solid #22a;
68   background-color:rgba(255,255,255,1.0);
69 }
70 }#GshBanner {
71   overflow:visible;
72   display:block;
73   width:100px;
74   height:100px;
75   left:inherent !important;
76 }
77 </style>
78 <script>
79 function Topbar_openWorkCodeView(){
80   function Topbar_showWorkCode(){
81     showHTMLCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
82   }
83   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
84 }
85 Topbar_openWorkCodeView();
86 function ConfigClick(){
87   if( 0 <> AffView.style.zIndex ){
88     AffView.style.saved_zIndex = AffView.style.zIndex;
89     AffView.style.zIndex = -1000;
90     GshSidebar.style.zIndex = -1;
91     GshPerfMon.style.zIndex = -1;
92     GMenu.style.zIndex = -1;
93   }else{
94     /AffView.style.zIndex = AffView.style.saved_zIndex;
95     AffView.style.zIndex = 1;
96     GshSidebar.style.zIndex = 1;
97     GshPerfMon.style.zIndex = 1;
98     GMenu.style.zIndex = 10000000;
99   }
100  console.log('AffZIndex'+AffView.style.zIndex);
101 }
102 function Gshell_initTopbar(){
103   GshTopbar.innerHTML = GshTitle.innerHTML;
104   /<div id="ConfigIcon" class="ConfigIcon">
105   if(true){
106     cfgi = document.createElement('img');
107     cfgi.className = "ConfigIcon";
108     cfgi.setAttribute('class','ConfigIcon');
109     GshTopbar.appendChild(cfgi);
110     cfgi.src = ConfigICON_DATA;
111   }
112   //cfgi.style.zIndex = 1000000000;
113   //cfgi.addEventListener('click',ConfigClick);
114   GshTopbar.addEventListener('click',ConfigClick);
115 }
116 }
117 </script>
118 <!-- Topbar_WorkCodeSpan ) -->
119 /* //<span>
120 //<-- ----- Work ) ----- -->
121
122 //<-- ----- Work { ----- -->
123 //<span id="Indexer_WorkCodeSpan">
124 /* 
125 <details><summary>Indexer</summary>
126 <!-- Indexer // 2020-1007 SatoxITS ( -->
127 <h2>Indexer</h2>
128 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
129 <input id="Indexer_OpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
130 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
131 <span id="Indexer_WorkCodeView"></span>
132 </details>
133 <style id="SidebarIndex">
134 #gsh {
135   display:block;
136   display:flex;
137   flex-direction:column;
138   flex-grow:1;
139 }
140 #GshMain {
141   z-index:1;
142   position:relative;
143   display:block;
144   width:80% !important;
145   left:19.5% !important;
146 }
147 #GshSidebar {
148   z-index:0;
149   position:relative !important;
150   position:fixed;
151   resize:both !important;
152   xoverflow-y:hidden !important;
153   xxxheight:100px !important;
154   xxxdisplay:inline !important;
155   left:0px;
156   top:0px;
157   width:19.5% !important;
158   min-width:80px;
159   xxxheight:100% !important;
160   height:0px;
161   color:#f00;
162   xxbackground-color:rgba(64,64,64,0.5);
163 }
```

```

162 xxbackground-color:#DFE3EB;xxx-PBlue;
163 }
164 }
165 #GshPerfMon {
166 position:relative;
167 display:block;
168 overflow:visible;
169 z-index:0 !important;
170 width:12px;
171 font-family:monospace, Courier New !important;
172 font-size:9pt !important;
173 color:#f84;
174 top:-20px;
175 }
176 #GshPerfMon:hover {
177 z-index:3 !important;
178 }
179 #GshSidebar:hover {
180 z-index:2;
181 overflow-x:visible !important;
182 background-color:rgba(255,255,255,0.7);
183 width:50%;
184 }
185 #GshIndexer {
186 z-index:0;
187 position:relative;
188 resize:both !important;
189 height:100px;
190 left:0px;
191 top:0px;
192 scroll-behavior: overflow !important;
193 padding-left:4pt;
194 font-size:0.5em;
195 white-space:nowrap;
196 xxx-background-color:rgba(64,160,64,0.6) !important;
197 color:#7794c6;xxx-PBlue;
198 xxbackground-color:#F0F0F0;xxx-PBlue;
199 background-color:#eeeeee;xxx-PBlue;
200 }
201 #GshIndexer:hover {
202 z-index:10000000;
203 overflow-x:visible !important;
204 color:#7794c6;xxx-PBlue;
205 xxbackground-color:#FFFFFF;xxx-PBlue;
206 background-color:rgba(255,255,255,0.7);
207 padding-right:0px;
208 width:80px;
209 }
210 #GshIndexer select {
211 color:#000000 !important;xxx-PBlue;
212 background-color:#FFFFFF;xxx-PBlue;
213 }
214 .IndexLine {
215 font-size:8pt !important;
216 font-family:Georgia;
217 display:block;
218 xxx-color:#fff;
219 xxx-color:#fffff5;xxx-PBlue;
220 xxx-color:#41516d;xxx-PBlue;
221 xxx-color:#7794c6;xxx-PBlue;
222 padding-right:4pt;
223 }
224 .IndexLine:hover {
225 font-size:10pt !important;
226 xxx-color:#228B22;
227 xxx-color:#fffff5;xxx-PBlue;
228 color:#516487;xxx-PBlue;
229 background-color:rgba(220,220,225,1.0);xxx-PBlue;
230 xxtext-shadow:1px 1px #333;
231 text-shadow:1px 1px #eee;
232 xxbackground-color:#516487;xxx-PBlue;
233 xxtext-decoration:underline !important;
234 }
235 
```

```

236 </style>
237
238 <script id="Indexer_WorkScript">
239 function Indexer_openWorkCodeView(){
240     function Indexer_showWorkCode(){
241         showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
242     }
243     Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
244 }
245 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
246 Indexer_openWorkCodeView();
247
248 var startPerfDate = new Date();
249 var prevPerfDate = startPerfDate;
250 function ShowResourceUsage(){
251     var newDate();
252     perf += '<br>';
253     perf += '<font color="gray">UA:' + window.navigator.userAgent + '<'+ '/font><br>\n';
254     perf += DateShort0(startPerfDate) + '<br>\n';
255     perf += DateShort() + '<br>\n';
256     elpa = d.getMemory() - startPerfDate.getTime();
257     itvl = d.getTime() - prevPerfDate.getTime();
258     perf += 'Elapsed: ' + elpa/1000 + ' s<br>\n';
259     perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
260     prevPerfDate = d;
261
262     if( performance.memory != undefined ){
263         mo = performance.memory;
264         mu0 = ((mo.usedJSHeapSize / 1000000.0) //toFixed(6));
265         perf += 'Memory: '+mu0+' MB<br>\n';
266     }
267     perf += '<br>\n';
268
269 //GshSidebar.innerHTML = perf;
270 GshPerfMon.innerHTML = perf;
271 //GshIndexer.innerHTML = "Memory: "+mu0+" MB";
272 //console.log('-- Perfmon heap: '+mu0+'/'+mo.totalHeapSize+'/'+mo.jsHeapSizeLimit);
273 if( true ) {
274     GshSidebar.style.zIndex = 1000;
275     GshIndexer.style.zIndex = 0;
276     GshPerfMon.style.zIndex = 1;
277     //GshSidebar.appendChild(GshPerfMon);
278     if( document.getElementById('primary') == null ) { // not in WordPress
279         GshPerfMon.style.position = 'absolute';
280     }
281     GshPerfMon.style.display = 'block';
282     GshPerfMon.style.marginLeft = '4px';
283     //GshPerfMon.style.top = '45px';
284     GshPerfMon.style.position = 'relative';
285     GshPerfMon.style.left = 'absolute';
286     copy = GshTopbar.getBoundingClientRect().top;
287     //copy = parseInt(copy) + 40;
288     //GshPerfMon.style.top = copy + 'px';
289     GshPerfMon.style.left = '0px';
290
291     GshMain.style.top = -GshPerfMon.getBoundingClientRect().height;
292 }
293 }
294 function ResetPerfMon(){
295     GshPerfMon.removeAttribute('style');
296     GshSidebar.removeAttribute('style');
297 }
298
299 var iserno = 0;
300 var GeneratedId = 0;
301 function generateIndex(ni,e,chv,nch,ht){
302     /> https://developer.mozilla.org/en-US/docs/Web/API/Element
303     e = document.createElement('div');
304     if( e.classList != null ){
305         c = e.classList.value;
306     }
307     //console.log('<'+e.nodeName+'> #'+'e.id'+'.'+c+' '+e.attributes);
308     if( e.nodeName == '#text'){ return ''; }
309     if( e.nodeName == '#comment' ){ return ''; }
310     if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
311         id = e.innerHTML;
312         GeneratedId += 1;
313         eid = 'GeneratedId-' + GeneratedId;
314         e.id = eid;
315     }
316     else{
317         if( e.nodeName == 'SUMMARY' ){
318             id = e.innerHTML;
319             GeneratedId += 1;
320             eid = 'GeneratedId-' + GeneratedId;
321             e.id = eid;
322         }
323     }
324     if( andie.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxxxxxentry-content' ) {
325         console.log('-- DIV entry-content begin');
326     }
}

```

```

324     id = e.innerHTML;
325     GeneratedId += 1;
326     eid = "GeneratedId-' + GeneratedId;
327     e.id = eid;
328     console.log('-- DIV entry-content end hash-child=' + e.hasChildNodes());
329   }else{
330     if( e.id == '' || e.id == 'undefined' ){
331       return '';
332     }else{
333       id = "#"+e.id;
334       eid = e.id;
335     }
336     iserno += 1;
337     ht += <div id="SeparatedRef" >iserno</div> class="IndexLine" href="#'+eid+'>';
338     ht += iserno + 'n'+e.nodeName + ':' + id;
339     if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+>div>; }
340     if( !e.hasChildNodes() ){ return ht + '<'+>div>; }
341     chv = e.childNodes;
342     nch = e.childNodes.length;
343     if( chv != null ) nch = chv.length;
344     ht += '<'+>chv + '<'+>div>;
345     for( let i = 0; i < chv.length; i++ ){
346       sec = nch + i;
347       if( ni == '' ){ sec = i; }
348       ht += generateIndex(sec,chv[i],null,0);
349     }
350   }
351 }
352 function onClickIndex(e){
353   tid = e.target.id;
354   tge = document.getElementById(tid);
355   eid = tge.getAttribute('href');
356   left = tge.getBoundingClientRect().left.toFixed(0)
357   top = tge.getBoundingClientRect().top.toFixed(0)
358   if( false ){
359     alert('index clicked mouse(x="'+e.clientX+', y="'+e.clientY+')'
360           + 'n'+id#'+ tid +' rx#'+rx + ',ry#'+ry
361           + '\neid#'+ eid + '\n'
362           + '\nhtml#'+ tge.outerHTML);
363   }
364   e = document.getElementById(eid);
365   sx = 'NaN';
366   sy = e.getBoundingClientRect().top;
367   console.log('sx#'+sx,'sy#'+sy);
368   ee.scrollTo(sx,sy);
369   window.scrollTo(sx,0);
370   //window.scrollTo(0,sy);
371 }
372 function Indexer_afterLoaded(){
373   sideIndex = document.getElementById('GshIndexer');
374   ht = '<'+>h3+>Index</h3>';
375   ht += generateIndex("",document.getElementById('gsh'),null,0,'');
376   if( (pri = document.getElementById('primary')) != null ){
377     ht += generateIndex("",pri,null,0,'');
378   }
379   ht += '<'+>br>';
380   ht += '<'+>br>';
381   ht += '<'+>br>';
382   ht += '<'+>br>';
383   sideIndex.innerHTML = ht;
384   sideIndex.addEventListener('click',onClickIndex);
385
386   if( (pri = document.getElementById('primary')) != null ){
387     console.log('-- Seems in WordPress');
388     pri.style.zIndex = 2000;
389
390     GshSidebar.style.setProperty('position','relative','important');
391     GshSidebar.style.top = '-140px';
392     //GshSidebar.style.setProperty('position','absolute','important');
393     //GshSidebar.style.top = '0px';
394
395     GshSidebar.style.setProperty('width','200px','important');
396     GshSidebar.style.setProperty('overflow','scroll','important');
397     GshSidebar.style.resize = 'both';
398
399     GshSidebar.style.left = '-100px';
400     GshIndexer.style.left = '100px';
401     GshIndexer.style.height = '140px';
402     gsh.appendChild(GshSidebar); // change parent
403   }else{
404     console.log('-- Seems not in WordPress');
405     GshSidebar.style.setProperty('position','fixed','important');
406   }
407   //document.addEventListener('load',Indexer_afterLoaded);
408
409 DestroyIndexBar = function(){
410   sideIndex = document.getElementById('GshIndexer');
411   sideIndex.innerHTML = "";
412   sideIndex.style = "";
413 }
414 
```

```

415 </script>
416
417 <!-- Indexer_WorkCodeSpan -->
418 </-- //<span>
419 //<!-- ----- Work -->
420
421
422 /*
423 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
424 <p>
425 <note>
426 It is a shell for myself, by myself, of myself. --SatoITS(~)
427 <a href="gsh-0.6.2.go.html">prev.</a>
428 </note>
429 </p>
430 <div id="GJFactory_x"></div>
431
432 <div>
433 <span id="GshMenu" class="GshMenu">
434 <span class="GshMenu1" id="GshMenuedit" onclick="html_edit();>Edit</span>
435 <span class="GshMenu1" id="GshMenuread" onclick="html_read();>Read</span>
436 <span class="GshMenu1" id="GshMenuLoad" onclick="html_load();>Load</span>
437 <span class="GshMenu1" id="GshMenuVers" onclick="html_ver0();>Vers</span>
438 <span id="gsh-WinId" onclick="winJump('0.1');>0</span>
439 <span class="GshMenu1" id="gsh-menu-exit" onclick="html_close();>xit</span>
440 <span class="GshMenu1" id="gsh-menu-for" onclick="html_for();>for</span>
441 <span class="GshMenu1" id="gsh-menu-Stop" onclick="html_stop(this,true);>Stop</span>
442 <span class="GshMenu1" id="GshMenuStop" onclick="html_stop(this);>Stop</span>
443 <span class="GshMenu1" id="GshMenuFold" onclick="html_fold(this);>Infold</span>
444 <span class="GshMenu1" id="gsh-menu-cksum" onclick="html_digest();>Digest</span>
445 <span class="GshMenu1" id="GshMenuSign" onclick="html_sign(this);>Sign</span>
446 <!-- / <span id="gsh-menu-pure" onclick="html_pure(this);>Pure</span -->
447 </span>
448 </div>
449 </>
450
451 /*
452 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
453 <h3>Fun to create a shell</h3>
454 <p>For a programmer, it must be far easy and fun to create his own simple shell
455 rightly fitting to his favor and necessities, than learning existing shells with
456 complex full features that he never use.
457 I, as one of programmers, am writing this tiny shell for my own real needs,
458 totally from scratch, with fun.
459 </p>
460 For a programmer, it is fun to learn new computer languages. For long years before
461 writing this software, I had been specialized to C and early HTML2 ::).
462 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
463 on demand as a novice of these, with fun.
464 </p>
465 The single file "gsh.go", that is executable by Go, contains all of the code written
466 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
467 HTML file that works as the viewer of the code of itself, and as the "home page" of
468 this software.
469 </p>
470 Because this HTML file is a Go program, you may run it as a real shell program
471 on your computer.
472 But you must be aware that this program is written under situation like above.
473 Needless to say, there is no warranty for this program in any means.
474 </p>
475 <address>Aug 2020, SatoITS (sato@its-more.jp)</address>
476 </details>
477 </>
478 /*
479 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
480 <h3>Cross-browser communication</h3>
481 <p>
482 <!-- to be written ...
483 </p>
484 </h3>
485 <h3>Vi compatible command line editor</h3>
486

```

```

486 <p>
487 The command line of GShell can be edited with commands compatible with
488 <a href="https://www.susmission.edu/computing/unix/vi.html"><b>vi</b></a>.
489 As a vi, you can enter <b></b> command mode<b></b> by <b></b>ESC<b></b> key,
490 then move around in the history by <b></b>j k / ? n <b></b>code<b></b>, or so.
491 or within the current line by <b></b>l h f w b 0 $ %<b></b> or so.
492 </p>
493 </details>
494 /
495 <div id="gsh-gindex">
496 <summary>Index</summary><div class="gsh-src">
497 Documents
498 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
500 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
501 Package structures
502 <a href="#">import</a>
503 <a href="#">#struct</a>
504 Main functions
505 <a href="#">#cmd-expansion</a> str-expansion</a> // macro processor
506 <a href="#">#find</a> <a href="#">#finder</a> // builtin find + du
507 <a href="#">#grep</a> <a href="#">grep</a> // builtin grep + wc + cksum + ...
508 <a href="#">#plugin</a> <a href="#">plugin</a> // plugin commands
509 <a href="#">#ex-commands</a> system</a> // external commands
510 <a href="#">#builtin</a> <a href="#">builtin</a> // builtin commands
511 <a href="#">#network</a> <a href="#">network</a> // network header
512 <a href="#">#remote</a> <a href="#">remote-sh</a> // remote shell
513 <a href="#">#redirect</a> <a href="#">redirect</a> // StdIn/out redirection
514 <a href="#">#history</a> <a href="#">history</a> // command history
515 <a href="#">#usage</a> <a href="#">usage</a> // resource usage
516 <a href="#">#encode</a> <a href="#">encode</a> // encode / decode
517 <a href="#">#decode</a> <a href="#">decode</a> // decode / encode
518 <a href="#">#getline</a> <a href="#">getline</a> // line editor
519 <a href="#">#scanf</a> <a href="#">scanf</a> // string decomposer
520 <a href="#">#interpreter</a> <a href="#">interpreter</a> // command interpreter
521 <a href="#">#main</a>
522 </span>
523 JavaScript part
524 <a href="#">#script-src-view</a> class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>
525 <a href="#">#gsh-data-frame</a> class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
526 CSS part
527 <a href="#">#style-src-view</a> class="gsh-link" onclick="jumpto_StyleView();">Source</a>
528 Reference
529 <a href="#">#gsh</a> class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
530 <a href="#">#gsh-reference</a> class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
531 Whole parts
532 <a href="#">#whole-src-view</a> class="gsh-link" onclick="jumpto_WholeView();">Source</a>
533 <a href="#">#whole-src-view</a> class="gsh-link" onclick="jumpto_WholeView();">Download</a>
534 <a href="#">#whole-src-view</a> class="gsh-link" onclick="jumpto_WholeView();">Dump</a>
535
536 </div>
537 </details>
538 /
539 //<details id="gsh-gocode">
540 // Go lang - So you can copy <div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
541 // gsh - Go lang based Shell
542 // (C) 2020 ITS more Co., Ltd.
543 // 2020-0807 created by satoxITS (sato@its-more.jp)
544
545 package main // gsh main
546
547 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
548 import (
549   "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
550   "errors"
551   "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
552   "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
553   "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
554   "time" // <a href="https://golang.org/pkg/time/">time</a>
555   "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
556   "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
557   "os" // <a href="https://golang.org/pkg/os/">os</a>
558   "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
559   "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
560   "net" // <a href="https://golang.org/pkg/net/">net</a>
561   "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
562   "/> http" // <a href="https://golang.org/pkg/html/">html</a>
563   "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
564   "go/types" // <a href="https://golang.org/pkg/types/">types</a>
565   "go/token" // <a href="https://golang.org/pkg/token/">token</a>
566   "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
567   "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
568   "/> gshdata" // gshell's logo and source code
569   "hash/crc32" // <a href="https://golang.org/pkg/hash/crc32/">crc32</a>
570   "golang.org/x/net/websocket"
571   "runtime"
572 )
573
574 /*
575 #include <stdio.h> // </stdio.h> to be closed as HTML tag :-p
576 #ifndef WIN32
577 #include <windows.h> // </windows.h>
578 // 2020-1022 added -- terminal mode on Windows
579 https://msdn.microsoft.com/en-us/windows/console/setconsolemode
580 https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
581 int setTermRaw(){
582     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
583     DWORD tmode = 0;
584     if( GetConsoleMode(hStdin,&tmode) ){
585         tmode |= ~ENABLE_ECHO_INPUT;
586         tmode |= ~ENABLE_LINE_INPUT;
587         tmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
588         if( SetConsoleMode(hStdin,tmode) ){
589             return tmode;
590         }
591     }
592 }
593
594 return 0;
595 }
596 int setTermMode(int tmode){
597     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
598     SetConsoleMode(hStdin,tmode);
599     return 0;
600 }
601 #else
602 int setTermRaw(){
603     return -1;
604 }
605 int setTermMode(int tmode){
606     return 0;
607 }
608 #endif
609 */
610 import "C"
611
612 /*
613 // 2020-0906 added
614 // <a href="https://golang.org/cmd/cgo/">cgo</a>
615 // <a href="https://golang.org/cmd/poll/">poll</a> // </poll.h> to be closed as HTML tag :-p
616 // typedef struct { struct pollfd fdv[8]; } pollFd;
617 // int pollx(pollFd *fdv, int nfds, int timeout);
618 // return poll(fdv->fdv,nfds,timeout);
619 */
620 import "C"
621
622 // 2020-1021 replaced poll() with channel/select
623 // 2020-0906 added,
624 func CPollInit(f*os.File, timeoutUs int)(ready uintptr){
625     var fdv = C.pollFd{ }
626     var nfds = 1
627     var timeout = timeoutUs/1000
628
629     fdv.fd[0].fd = C.int(fd)
630     fdv.fd[0].events = C.POLLIN
631     if 0 < EventRecvFd {
632         fdv.fd[1].fd = C.int(EventRecvFd)
633         fdv.fd[1].events = C.POLLIN
634         nfds += 1
635     }
636     r := C.polx(&fdv,C.int(nfds),C.int(timeout))
637     if r <= 0 {
638         return 0
639     }
640     if (int(fdv.fd[1].events) & int(C.POLLIN)) != 0 {
641         //fprintf(stderr,"--> got Event\n");
642         return uintptr(EventFdOffset + fdv.fd[1].fd)
643     }
644     if (int(fdv.fd[0].events) & int(C.POLLIN)) != 0 {
645         return uintptr(NormalFdOffset + fdv.fd[0].fd)
646     }
647     return 0
}

```

```

648 }
649 */
650
651 const {
652     NAME = "gsh"
653     VERSION = "0.7.4"
654     DATE = "2020-10-23"
655     AUTHOR = "SatoxiT$^=)"//"
656 }
657 var {
658     GSH_HOME = ".gsh" // under home directory
659     GSH_PORT = 9999
660     MaxStreamSize = int64(128*1024*1024) // 128GiB is too large?
661     FLSIZE = int64(1024*1024)
662     PATHSEP = ":" // should be ";" in Windows
663     DIRSEP = "/" // canbe \ in Windows
664     OnWindows = false;
665 }
666 func initGshEnv(){
667     if(runtime.GOOS == "windows"){
668         PATHSEP = ";";
669         DIRSEP = "\\";
670         OnWindows = true;
671     }else{
672     }
673 }
674 */
675 // -XX logging control
676 // --A-- all
677 // --I-- info
678 // --D-- debug
679 // --T-- time and resource usage
680 // --W-- warning
681 // --E-- error
682 // --F-- fatal error
683 // --Xn network
684 /*
685 <a name="Struct">Structures</a>
686 */
687 // 2020-1022 Unix/Windows
688 // -----
689 type astat syscall.Stat_t;
690 type astat_t struct syscall.Stat_t {
691     astat_t struct {
692         Size int64
693         Mode os.FileMode
694         Rdev int64
695         Blocks int64
696         Nlink int64
697     }
698 }
699 func alstat(path string, astat *astat_t)(error){
700     /*
701     stat := syscall.Stat_t{};
702     err := syscall.Lstat(path,&stat);
703     *astat = astat_t(stat);
704     */
705     fi,err := os.Stat(path);
706     if(err == nil){
707         astat.Mode = fi.Mode();
708         astat.Size = fi.Size();
709     }
710     return err;
711 }
712 func arstat(fd int, astat *astat_t)(error){
713     /*
714     stat := syscall.Stat_t();
715     err := syscall.Fstat(fd,&stat);
716     *astat = astat_t(stat);
717     */
718     err := errors.New("NotImplemented-Fstat");
719     //fmt.Printf("--- E-- fstat(%v,%v)\n",fd,err);
720     return err;
721 }
722 func aAccess(path string, mode uint32)(error){
723     /*
724     /err := syscall.Access(path,mode);
725     //err := errors.New("NotImplemented-Access");
726     /err = os.Stat(path);
727     //fmt.Printf("-- Access(%v,%v)\n",path,mode,err);
728     if(err == nil){
729         mode := fi.Mode();
730         if( mode.IsRegular() ){
731             perm := fi.Mode().Perm();
732             if( (uint32(perm) & mode) != 0 ){
733                 return nil;
734             }
735             return errors.New("NotAccessible");
736         }
737         return errors.New("NotRegularFile");
738     }
739     return err;
740 }
741 */
742 // 2020-1022 Unix/Windows
743 // -----
744 type aRusage struct {
745     syscall.Rusage
746     Utme time.Duration
747     Stime time.Duration
748     /Sys interface{}
749 }
750 /*
751 const aRUSAGE_SELF = syscall.RUSAGE_SELF
752 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
753 */
754 const aRUSAGE_SELF = 0
755 const aRUSAGE_CHILDREN = 1
756 func aGetrusage(sel int, ru *aRusage){
757     /*
758     sysru := syscall.Rusage{};
759     syscall.Getrusage(sel,&sysru);
760     ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
761     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
762     */
763     /*
764     func aSetrusage(ru *aRusage, ps *os.ProcessState){
765         ru.Utime = ps.UserTime();
766         ru.Stime = ps.SystemTime();
767     }
768     */
769 func showRusage(what string,argv []string, ru *aRusage){
770     fmt.Println("#",what)
771     /ru.Utime = time.Duration(int64(ru.Utime.Sec)*1000000000+int64(ru.Utime.Usec))
772     /ru.Stime = time.Duration(int64(ru.Stime.Sec)*1000000000+int64(ru.Stime.Usec))
773     fmt.Println("Usec=%d.%06ds",ru.Utime/1000000000,(ru.Utime/1000)%1000000);
774     fmt.Println(" Sys=%d.%06ds ",ru.Stime/1000000000,(ru.Stime/1000)%1000000);
775     /*
776     fmt.Println(" Rss=%v",ru.Maxrss)
777     if(isin("-l",argv)){
778         fmt.Printf(" Minflt=%v",ru.Minflt)
779         fmt.Printf(" Majflt=%v",ru.Majflt)
780         fmt.Printf(" IxRss=%v",ru.Ixrss)
781         fmt.Printf(" IdRSS=%v",ru.Idrss)
782         fmt.Println(" Nswap=%v",ru.Nswap)
783         fmt.Println(" Read=%v",ru.Inblock)
784         fmt.Println(" Write=%v",ru.Outblock)
785     }
786     fmt.Println(" Snd=%v",ru.Msgsnd)
787     fmt.Println(" Rcv=%v",ru.Msgrcv)
788     /ru.Isignals = argv[1]
789     fmt.Println(" Sig=%v",ru.Nsignals)
790     */
791 }
792 */
793 */
794 */
795 type GCommandHistory struct {
796     StartAt time.Time // command line execution started at
797     EndAt time.Time // command line execution ended at
798     ResCode int // exit code of (external command)
799     CmdStr string // error string
800     OutData *os.File // output - result of the command
801     Foundfile []string // output - result of ufind
802     Rusageev [2]aRusage // Resource consumption, CPU time or so
803     Cmdid int // maybe with identified with arguments or impact
804     Workdir string // redirection commands should not be the Cmdid
805     WorkdirX int // index in ChdirHistory
806     CmdLine string // command line
807 }
808 */
809 type GCkdirHistory struct {

```

```

810     Dir      string
811     MovedAt   time.Time
812     CmdIndex  int
813   }
814   type CmdMode struct {
815     BackGround bool
816   }
817   type Event struct {
818     when    time.Time
819     event   int
820     evarg  int64
821     CmdIndex int
822   }
823 var CmdIndex int
824 Events []Event
825 type EPluginInfo struct {
826   Spec   *plugin.Plugin
827   Addr   plugin.Symbol
828   Name   string // maybe relative
829   Path   string // this is in Plugin but hidden
830 }
831 type GServer struct {
832   host   string
833   port   string
834 }
835
836 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
837 const ( // SumType
838   SUM_ITEMS  = 0x0000001 // items count
839   SUM_SIZE   = 0x0000002 // data length (simply added)
840   SUM_SIZEHASH = 0x0000004 // data length (hashed sequence)
841   SUM_DATASHASH = 0x0000008 // sum of data (hashed sequences)
842   // also envelope attributes like time stamp can be a part of digest
843   // hashed value of sizes or mod-date of files will be useful to detect changes
844
845   SUM_WORDS  = 0x0000010 // word count is a kind of digest
846   SUM_LINES   = 0x0000020 // line count is a kind of digest
847   SUM_SUM64  = 0x0000040 // simple add of bytes, useful for human too
848
849   SUM_SUM32_BITS = 0x000100 // the number of true bits
850   SUM_SUM32_2BYTE = 0x000200 // 16bits words
851   SUM_SUM32_4BYTE = 0x000400 // 32bits words
852   SUM_SUM32_8BYTE = 0x000800 // 64bits words
853
854   SUM_SUM16_BSD = 0x010000 // UNIXsum -sum -bsd
855   SUM_SUM16_SYSV = 0x020000 // UNIXsum -sum -sysv
856   SUM_UNIXFILE  = 0x040000
857   SUM_CRCIEEE  = 0x080000
858 )
859 type CheckSum struct {
860   Files   int64 // the number of files (or data)
861   Size    int64 // content size
862   Words   int64 // word count
863   Lines   int64 // line count
864   SumType int
865   Sum64  uint64
866   Crc32Table crc32.Table
867   Crc32Val  uint32
868   Sum16  uint
869   CTime   time.Time
870   ATime   time.Time
871   MTime   time.Time
872   Start   time.Time
873   Done    time.Time
874   RusgAStart [2]uintptr
875   RusgAEnd  [2]uintptr
876 }
877 type ValueStack [][]string
878 type GshContext struct {
879   Startdir  string // the current directory at the start
880   GetLine   string // gsh-line command as a input line editor
881   CommandHistory []GCommandHistory // the 1st entry is wd at the start
882   /gshPA   syscall.ProcAttr
883   gshPA   os.ProcAttr
884   CmdCommandHistory []GCommandHistory
885   CmdCurrent GCommandHistory
886   Background bool
887   BackgroundJobs []ios.ProcessState; //[]int
888   LastUsage  ausage
889   GahHomeDir string
890   TerminalId int
891   CmdTrace  bool // should be [map]
892   CmdTime   bool // should be [map]
893   PrintLines []GPrintInfo
894   iValues  []string
895   iDelimeter string // field separator of print out
896   iFormat   string // default print format (of integer)
897   iValStack ValueStack
898   LastServer GServer
899   PRWD   string // {gsh://host:port}
900   PRWD   string // remote (target, there) working directory
901   lastCheckSum  CheckSum
902 }
903
904 func nsleep(ns time.Duration){
905   time.Sleep(ns)
906 }
907 func usleep(ns time.Duration){
908   msleep(ns*1000)
909 }
910 func msleep(ns time.Duration){
911   msleep(ns*1000000)
912 }
913 func sleep(ns time.Duration){
914   msleep(ns*1000000000)
915 }
916
917 func strBegins(str, pat string)(bool){
918   if len(pat) <= len(str){
919     yes := str[0:len(pat)] == pat
920     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
921     return yes
922   }
923   //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
924   return false
925 }
926 func isin(what string, list []string) bool {
927   for v := range list {
928     if v == what {
929       return true
930     }
931   }
932   return false
933 }
934 func isinX(what string,list[]string)(int{
935   for i,v := range list {
936     if v == what {
937       return i
938     }
939   }
940   return -1
941 }
942 func env(opts []string) {
943   env := os.Environ()
944   if isn("-s", opts){
945     sort.Slice(env,func(i,j int) bool {
946       return env[i] < env[j]
947     })
948   }
949   for _, v := range env {
950     fmt.Printf("%v\n",v)
951   }
953 }
954
955 // - rewriting should be context dependent
956 // - should postpone until the real point of evaluation
957 // - should rewrite only known notation of symbol
958 func scanInt(str string)(val int,int,int){
959   leng = -1
960   for i,ch := range str {
961     if '0' <= ch && ch <= '9' {
962       leng = i+1
963     }else{
964       break
965     }
966   }
967   if 0 < leng {
968     val,_ := strconv.Atoi(str[0:leng])
969     return val,leng
970   }else{
971     return 0,0
972   }

```

```

972     }
973 }
974 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
975     if len(str[i+1]) == 0 {
976         return 0,rstr
977     }
978     hi := 0
979     histlen := len(gshCtx.CommandHistory)
980     if str[i+1] == `!` {
981         hi = histlen - 1
982         leng = 1
983     }else{
984         hi,leng = scanInt(str[i+1:])
985         if leng == 0 {
986             return 0,rstr
987         }
988         if hi < 0 {
989             hi = histlen + hi
990         }
991     }
992     if 0 <= hi && hi < histlen {
993         var ext byte
994         if i < len(str[i+leng]) {
995             ext = str[i+leng][1]
996         }
997         //fmt.Println("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
998         if ext != `!` {
999             leng += 1
1000         }
1001         xlist := [i]string{}
1002         list := gshCtx.CommandHistory[hi].FoundFile
1003         for _,v := range list {
1004             //list[i] = escapeWhiteSP(v)
1005             xlist = append(xlist,escapeWhiteSP(v))
1006         }
1007         //rstr += strings.Join(list, " ")
1008         rstr += strings.Join(xlist, " ")
1009     }else{
1010         if ext == `!` || ext == `d` {
1011             // !N@ .. workdir at the start of the command
1012             leng += 1
1013             rstr += gshCtx.CommandHistory[hi].Workdir
1014         }else{
1015             rstr += gshCtx.CommandHistory[hi].CmdLine
1016         }
1017     }
1018     leng = 0
1019 }
1020 return leng,rstr
1021 }
1022 func escapeWhiteSP(str string)(string){
1023     if len(str) == 0 {
1024         return "\\\z" // empty, to be ignored
1025     }
1026     rstr := ""
1027     for _,ch := range str {
1028         switch ch {
1029             case `\\`:
1030                 rstr += `\\\\`
1031             case `\'`:
1032                 rstr += `\\\'`
1033             case `\"`:
1034                 rstr += `\\\"`
1035             default:
1036                 rstr += string(ch)
1037         }
1038     }
1039     return rstr
1040 }
1041 func unescapeWhiteSP(str string)(string){ // strip original escapes
1042     rstr := ""
1043     for i := 0; i < len(str); i++ {
1044         ch := str[i]
1045         if ch == `\\` {
1046             if i+1 < len(str) {
1047                 switch str[i+1] {
1048                     case `z`:
1049                         continue;
1050                     }
1051             }
1052         }
1053     }
1054     return rstr
1055 }
1056 func unescapeWhiteSPv(strv []string)([]string){ // strip original escapes
1057     ustrv := []string{}
1058     for _,v := range strv {
1059         ustrv = append(ustrv,unescapeWhiteSP(v))
1060     }
1061     return ustrv
1062 }
1063 // <a name="comexpansion">str-expansion</a>
1064 func substSubst(gshCtx *GshContext,str string,histonly bool) string {
1065     rbuf := []byte{}
1066     if false{
1067         // Unicode should be cared as a character
1068         return str
1069     }
1070     //rstr := ""
1071     inEsc := 0 // escape character mode
1072     for i := 0; i < len(str); i++ {
1073         //fmt.Println("--D--Subst %v:%v\n",i,str[i:])
1074         ch := str[i]
1075         if inEsc == 0 {
1076             if ch == `!` {
1077                 inEsc,inxstr := substHistory(gshCtx,str,i,rstr)
1078                 leng := substHistory(gshCtx,str,i,"")
1079                 if 0 < leng {
1080                     if _,rs := substHistory(gshCtx,str,`!`,"");
1081                     rbuf = append(rbuf,[byte(rs)])
1082                     i += leng
1083                     //rstr = xstr
1084                     continue
1085                 }
1086                 switch ch {
1087                     case `\\`:
1088                         inEsc = `\\`;
1089                         //case `%`:
1090                         //inEsc = `%`;
1091                         case `$`:
1092                         }
1093                 switch inEsc {
1094                     case `\\`:
1095                         switch ch {
1096                             case `\\`:
1097                                 case `s`:
1098                                     case `t`:
1099                                         case `r`:
1100                                         case `n`:
1101                                         case `z`:
1102                                         inEsc = 0; continue // empty, to be ignored
1103                         }
1104                     case `%`:
1105                         switch {
1106                             case ch == `s`:
1107                                 case ch == `t`:
1108                                     //rstr = rstr + time.Now().Format(time.Stamp)
1109                                     rs := time.Now().Format(time.Stamp)
1110                                     rbuf = append(rbuf,[byte(rs)])
1111                                     inEsc = 0
1112                                     continue;
1113                                     default:
1114                                         // postpone the interpretation
1115                                         //rstr = rstr + "%" + string(ch)
1116                                         rbuf = append(rbuf,ch)
1117                                         inEsc = 0
1118                                         continue;
1119                                     }
1120                     inEsc = 0
1121                 }
1122                 //rstr = rstr + string(ch)
1123                 rbuf = append(rbuf,ch)
1124             }
1125             //fmt.Println("--D--subst(%s)(%s)\n",str,string(rbuf))
1126             return string(rbuf)
1127         }
1128     }
1129     func showFileInfo(path string, opts []string) {
1130         if isnin("-l",opts) || isnin("-ls",opts) {
1131             fi, err := os.Stat(path)
1132             if err != nil {
1133                 fmt.Printf("----- (%v)",err)

```

```

1134     }else{
1135         mod := fi.ModTime()
1136         date := mod.Format(time.Stamp)
1137         fmt.Printf("%v %s",fi.Mode(),date)
1138     }
1139 }
1140 fmt.Println("#",path)
1141 if isnin("-sp",opts) {
1142     fmt.Println(" ")
1143 }else{
1144     if ! isnin("-n",opts) {
1145         fmt.Println("\n")
1146     }
1147 }
1148 func userHomeDir()(string,bool){
1149 /*
1150 homedir,_ = os.UserHomeDir() // not implemented in older Golang
1151 */
1152 homedir,found := os.LookupEnv("HOME")
1153 //fmt.Println("--> HOME=%v(%n)",homedir,found)
1154 if !found {
1155     return "/tmp",found
1156 }
1157 return homedir,found
1158 }
1159
1160 func toFullPath(path string) (fullpath string) {
1161     if path[0] == '/' {
1162         return path
1163     }
1164     pathv := strings.Split(path,DIRSEP)
1165     switch len(pathv) {
1166     case pathv[0] == ":":
1167         pathv[0] = os.Getenv("HOME")
1168     case pathv[0] == ".": // all ones should be interpreted
1169         cwd, _ := os.Getwd()
1170         ppathv := strings.Split(cwd,DIRSEP)
1171         pathv[0] = strings.Join(ppathv,DIRSEP)
1172     case pathv[0] == userHomeDir():
1173         pathv[0] = userHomeDir()
1174     default:
1175         cwd, _ := os.Getwd()
1176         pathv[0] = cwd + DIRSEP + pathv[0]
1177     }
1178     return strings.Join(pathv,DIRSEP)
1179 }
1180
1181 func IsRegFile(path string)(bool){
1182     fi, err := os.Stat(path)
1183     if err != nil {
1184         fm := fi.Mode()
1185         return fm.IsRegular()
1186     }
1187     return false
1188 }
1189
1190 // <a name="encode">Encode / Decode</a>
1191 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1192 func (gshctx *GshContext)Enc(argv[]string){
1193     file := os.Stdin
1194     buff := make([]byte,LINESIZE)
1195     li := 0
1196     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1197     for li = 0; ; li++ {
1198         count, err := file.Read(buff)
1199         if count <= 0 {
1200             break
1201         }
1202         if err != nil {
1203             break
1204         }
1205         encoder.Write(buff[0:count])
1206     }
1207     encoder.Close()
1208 }
1209 func (gshctx *GshContext)Dec(argv[]string){
1210     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1211     li := 0
1212     buff := make([]byte,LINESIZE)
1213     for li = 0; ; li++ {
1214         count, err := decoder.Read(buff)
1215         if count <= 0 {
1216             break
1217         }
1218         if err != nil {
1219             break
1220         }
1221         os.Stdout.Write(buff[0:count])
1222     }
1223 }
1224 // lnsp [N] [-crlf]\n
1225 func (gshctx *GshContext)SplitLine(argv[]string){
1226     strRep := isnin("-str",argv) // "...+
1227     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1228     n := 0
1229     toI := 0
1230     for ni := 0; ; ni++ {
1231         line, err := reader.ReadString('\n')
1232         if len(line) <= 0 {
1233             if err != nil {
1234                 fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d (%v)\n",ni,toI,err)
1235                 break
1236             }
1237         }
1238         off := 0
1239         ilen := len(line)
1240         remlen := len(line)
1241         if strRep { os.Stdout.WriteString([]byte("\n")) }
1242         for oi := 0; < remlen; oi++ {
1243             olen := remlen
1244             addnl := false
1245             if 72 < olen {
1246                 olen /= 72
1247                 addnl = true
1248             }
1249             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d\n",
1250                         toI,ni,oi,off,olen,remlen,ilen)
1251             toI += olen
1252             os.Stdout.WriteString([]byte(line[0:olen]))
1253             if addnl {
1254                 if strRep {
1255                     os.Stdout.WriteString([]byte("\n"))
1256                 }else{
1257                     /*os.Stdout.WriteString([]byte("\r\n"))
1258                     os.Stdout.WriteString([]byte("\n"))
1259                     os.Stdout.WriteString([]byte("\n"))
1260                 }*/
1261                 line = line[olen:]
1262                 off += olen
1263                 remlen -= olen
1264             }
1265             if strRep { os.Stdout.WriteString([]byte("\n")) }
1266         }
1267     }
1268     fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d\n",ni,toI)
1269 }
1270
1271 // CRC32 <a href="https://golang.jp/pkg/hash-crc32">crc32</a>
1272 // 1 0000 0100 1100 0001 0001 1101 0111
1273 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
1274 var CRC32NET uint32 = uint32(0x04C11DB7) // Net cksum
1275 func bytencrc32add(crc uint32,si[]byte,len uint64)(uint32){
1276     var ol uint64
1277     for ol = 0; ol < len; ol++ {
1278         var oct = str[ol]
1279         for bi := 0; bi < 8; bi++ {
1280             crc = crc ^ byte(crc<<1)^ byte((ol&1)&oct)
1281             if ((crc & 0x80000000) != 0) {
1282                 ovf1 := (oct & 0x80) != 0
1283                 ovf2 := (oct & 0x80) != 0
1284                 ovf := (ovf1 && ovf2) || (!ovf1 && ovf2)
1285                 oct <<= 1
1286                 crc <<= 1
1287             }
1288         }
1289     }
1290     //printf(stderr,"--CRC32 return %d\n",crc,len)
1291     return crc;
1292 }
1293 func bytencrc32end(crc uint32, len uint64)(uint32){
1294     var silen = make([]byte,4)
1295     var li = 0
1296     for li = 0; li < 4; {

```

```

1296     slen[li] = byte(len)
1297     li += 1
1298     len >>= 8
1299     if( len == 0 ){
1300         break
1301     }
1302     crc = byteCRC32add(crc,slen,uint64(li))
1303     crc ^= 0xFFFFFFFF
1304     return crc
1305 }
1306 }
1307 func strcRC32(st string,len uint64)(crc uint32){
1308     crc = byteCRC32add(0,[]byte(st),len)
1309     crc = byteCRC32end(crc,len)
1310     //printf(stderr,"-CRC32 %d\n",crc,len)
1311     return crc
1312 }
1313 func CRC32FFinish(crc uint32, table *crc32.Table, len uint64)(uint32){
1314     var slen = make([]byte,4)
1315     var li = 0
1316     for li = 0; li < 4; {
1317         slen[li] = byte(len & 0xFF)
1318         li += 1
1319         len >>= 8
1320         if( len == 0 ){
1321             break
1322         }
1323     }
1324     crc = crc32.Update(crc,table,slen)
1325     crc ^= 0xFFFFFFFF
1326     return crc
1327 }
1328
1329 func (gsh*GshContext)xCksum(path string,argv[]string, sum*CheckSum)(int64){
1330     if isin("-type/f",argv) && !IsRegFile(path){
1331         return 0
1332     }
1333     if isin("-type/d",argv) && IsRegFile(path){
1334         return 0
1335     }
1336     file, err := os.OpenFile(path,os.O_RDONLY,0)
1337     if err != nil {
1338         fmt.Printf("--E-- cksum tv (%v)\n",path,err)
1339         return -1
1340     }
1341     defer file.Close()
1342     if gsh.CmdTrace { fmt.Printf("--I-- cksum tv %v\n",path,argv) }
1343
1344     bi := 0
1345     var buff = make([]byte,32*1024)
1346     var total int64 = 0
1347     var initTime = time.Time{}
1348     if sum.Start == initTime {
1349         sum.Start = time.Now()
1350     }
1351     for bi = 0; ; bi++ {
1352         count,err := file.Read(buff)
1353         if count <= 0 || err != nil {
1354             break
1355         }
1356         if (sum.SumType & SUM_SUM64) != 0 {
1357             s := sum.Sum64
1358             for ,c := range buff[0:count] {
1359                 s += uint64(c)
1360             }
1361             sum.Sum64 = s
1362         }
1363         if (sum.SumType & SUM_UNIXFILE) != 0 {
1364             sum.Crc32Val = byteCRC32add(sum.Crc32Val,buff,uint64(count))
1365         }
1366         if (sum.SumType & SUM_CRC16EE) != 0 {
1367             sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
1368         }
1369         // <a href="https://en.wikipedia.org/wiki/BSB_checksum">BSD checksum</a>
1370         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1371             s := sum.Sum16
1372             for ,c := range buff[0:count] {
1373                 s += ((s & 1) + ((s & 1)<<15)
1374                 s += int(c)
1375                 s &= 0xFFFF
1376             }
1377             //fmt.Printf("BSDsum: %d(%d) %d\n",sum.Size+int64(i),i,s)
1378             sum.Sum16 = s
1379         }
1380         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1381             for bj := 0; bj < count; bj++ {
1382                 sum.Sum16 += int(buff[bj])
1383             }
1384         }
1385         total += int64(count)
1386     }
1387     sum.Done = time.Now()
1388     sum.Files += 1
1389     sum.Size += total
1390     if isin("-s",argv) {
1391         fmt.Printf("%v ",total)
1392     }
1393     return 0
1394 }
1395
1396 // <a name="grep">grep</a>
1397 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
1398 // a*,lab,c... ... sequential combination of patterns
1399 // what "LINE" is should be definable
1400 // generic line-by-line processing
1401 // grep [-v]
1402 // grep [-c]
1403 // uniq [-c]
1404 // tail [-f]
1405 // sed s/x/y/ or awk
1406 // grep with line count like wc
1407 // rewrite contents is not specified
1408 func (gsh*GshContext)xgrep(path string,rexpv[]string)(int){
1409     file, err := os.OpenFile(path,os.O_RDONLY,0)
1410     if err != nil {
1411         fmt.Printf("--E-- grep tv (%v)\n",path,err)
1412         return -1
1413     }
1414     defer file.Close()
1415     if gsh.CmdTrace { fmt.Printf("--I-- grep tv %v\n",path,rexpv) }
1416     //reader := bufio.NewReaderSize(file,LINESIZE)
1417     reader := bufio.NewReaderSize(file,80)
1418     li := 0
1419     found := 0
1420     for li = 0; ; li++ {
1421         line, err := reader.ReadString('\n')
1422         if len(line) <= 0 {
1423             break
1424         }
1425         if 150 < len(line) {
1426             // maybe binary
1427             break;
1428         }
1429         if err != nil {
1430             break
1431         }
1432         if 0 < strings.Index(string(line),rexpv[0]) {
1433             found += 1
1434             fmt.Printf("%s:%d: %s",path,li,line)
1435         }
1436     }
1437     //fmt.Printf("total %d lines %d\n",li,path)
1438     //if 0 < found { fmt.Printf("(found %d lines %s))\n",found,path); }
1439     return found
1440 }
1441
1442 // <a name="finder">Finder</a>
1443 // finding files with it name and contents
1444 // file names are ORed
1445 // show the content with % fmt list
1446 // ls -R
1447 // tar command by adding output
1448 type fileStruct struct{
1449     Err int64 // access error or so
1450     Size int64 // content size
1451     DupSize int64 // content size from hard links
1452     Blocks int64 // number of blocks (of 512 bytes)
1453     DirectBlocks int64 // Blocks pointed from hard links
1454     MLinks int64 // hard links
1455     Words int64
1456     Lines int64
1457     Files int64

```

```

1458     Dirs int64 // the num. of directories
1459     SLink int64
1460     Flat int64 // the num. of flat files
1461     MaxDepth int64
1462     MaxNameLen int64 // max. name length
1463     nextRepo time.Time
1464 }
1465 func showUsage(dir string,fusage *fileSum){
1466     btime := float64((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1467     /bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1468
1469     fmt.Printf("v%v %v files (%vd vs %vb) %.6f MB (%.2f MBK)\n",
1470             dir,
1471             fusage.Files,
1472             fusage.Dirs,
1473             fusage.Symlink,
1474             fusage.HLinks,
1475             float64(fusage.Size)/1000000.0,bsumdup);
1476 }
1477 const {
1478     S_IFMT = 0170000
1479     S_IFCHR = 0020000
1480     S_IFDIR = 0040000
1481     S_IFREG = 0100000
1482     S_IFLNK = 0120000
1483     S_IFSOCK = 0140000
1484 }
1485 func cumInfo(fsum *fileSum, path string, staterr error, fstat aStat_t, argv[]string, verb bool)(*fileSum){
1486     now := time.Now()
1487     if time.Second < now.Sub(fsum.nextRepo) {
1488         if fsum.nextRepo.IsZero(){
1489             tstamp := now.Format(time.Stamp)
1490             showUsage(tstamp,fsum)
1491         }
1492         fsum.nextRepo = now.Add(time.Second)
1493     }
1494     if staterr != nil {
1495         fsum.Err += 1
1496     }
1497     return fsum
1498 }
1499 fsum.Files += 1
1500 if 1 < fstat.Nlink {
1501     // must ignore only ones...
1502     // at least ignore ones in the same directory
1503     if fstat.Mode & S_IFMT == S_IFREG {
1504         if (fstat.Mode & S_IFMT) == S_IFREG {
1505             fsum.HLinks += 1
1506             fsum.DupBlocks += int64(fstat.Blocks)
1507             //fmt.Printf("---Dup HardLink %v %s\n",fstat.Nlink,path)
1508         }
1509     }
1510     fsum.Size += fstat.Size
1511     fsum.Blocks += int64(fstat.Blocks)
1512     //if verb { fmt.Printf("(%dBk) %s",fstat.Blocks/2,path) }
1513     if isn("-ls",argv) {
1514         //if "-ls" {verb { fmt.Println("%d %s",fstat.Blksize,fstat.Blocks) }
1515     }
1516     //if finfo.IsDir()
1517     if (fstat.Mode & S_IFMT) == S_IFDIR {
1518         fsum.Dirs += 1
1519     }
1520     //if (finfo.Mode() & os.ModeSymlink) != 0
1521     if (fstat.Mode & S_IFMT) == S_IFLNK {
1522         //if verb { fmt.Println("%slink(%v,%s)",fstat.Mode,finfo.Name()) }
1523         //if verb { fmt.Println("%slink(%o,%s)",fstat.Mode,finfo.Name()) }
1524         fsum.Symlink += 1
1525     }
1526     return fsum
1527 }
1528
1529 func (gsh*GshContext)xxFindEnv(depth int,total *fileSum,dir string,dstat aStat_t, ei int, entv []string,npatv[]string,argv[]string)(*fileSum){
1530     nols := isn("-grep",argv)
1531     // sort entv
1532     /*
1533     if isn("-t",argv){
1534         sort.Slice(filev, func(i,j int) bool {
1535             return 0 < filev[i].ModTime().Sub(filev[j].ModTime()))
1536         })
1537     */
1538     /*
1539     if isn("-a",argv){
1540         sort.Slice(filev, func(i,j int) bool {
1541             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1542         })
1543     }
1544     if isn("-U",argv){
1545         sort.Slice(filev, func(i,j int) bool {
1546             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1547         })
1548     */
1549     /*
1550     */
1551     if isn("-S",argv){
1552         sort.Slice(filev, func(i,j int) bool {
1553             return filev[j].Size() < filev[i].Size()
1554         })
1555     */
1556     /*
1557     for _,filename := range entv{
1558         for _,npat := range npatv {
1559             match := true
1560             if npat == "*" {
1561                 match = true
1562             }else{
1563                 match, _ = filepath.Match(npat,filename)
1564             }
1565             path := dir + DIRSEP + filename
1566             if !match {
1567                 continue
1568             }
1569             var fstat aStat_t
1570             staterr := alstat(path,&fstat)
1571             if staterr != nil {
1572                 if isn("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
1573                 continue;
1574             }
1575             if isn("-du",argv) && (fstat.Mode & S_IFDIR == S_IFDIR {
1576                 // should not show size of directory in "-du" mode ...
1577             }else{
1578                 if !nols && isn("-s",argv) && ((isn("-du",argv) || isn("-a",argv)) {
1579                     if isn("-du",argv) {
1580                         fmt.Println("%dt",fstat.Blocks/2)
1581                     }
1582                 }
1583                 showFileInfo(path,argv)
1584             }
1585             if true { // && isn("-du",argv)
1586                 total = cumInfo(total,path,staterr,fstat,argv,false)
1587             }
1588             if isn("-wc",argv) {
1589             }
1590             /*
1591             if gsh.lastCheckSum.SumType != 0 {
1592                 gsh.XChecksum(path,argv,&gsh.lastCheckSum);
1593             }
1594             x := isnX("-grep",argv); // -grep will be convenient like -ls
1595             if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1596                 if IsRegfile(path){
1597                     found := gsh.Grep(path,argv[x+1:])
1598                     if true < found {
1599                         foundv := gsh.CmdCurrent.FoundFile
1600                         if len(foundv) < 10 {
1601                             gsh.CmdCurrent.FoundFile =
1602                             append(gsh.CmdCurrent.FoundFile,path)
1603                         }
1604                     }
1605                 }
1606             }
1607             if !isIn("-r0",argv) { // -d 0 in du, -depth n in find
1608                 //total.Depth += 1
1609                 if (fstat.Mode & S_IFMT) == S_IFLNK {
1610                     continue
1611                 }
1612                 if dstat.Rdev != fstat.Rdev {
1613                     fmt.Printf("---- don't follow differnet device %v(%v) %v(%v)\n",
1614                         dir,dstat.Rdev,path,fstat.Rdev)
1615                 }
1616                 if (fstat.Mode & S_IFMT) == S_IFDIR {
1617                     total = gsh.xxFind(depth+1,total,path,npav,argv)
1618                 }
1619             }
1620         }
1621     }
1622 }

```

```

1620     }
1621   }
1622 }
1623 return total
1624 }
1625 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum{
1626   nols := isin("-grep",argv)
1627   dirfile,err := os.OpenFile(dir,os.O_RDONLY,0)
1628   if err == nil {
1629     defer dirfile.Close()
1630   }else{
1631   }
1632 }
1633 prev := *total
1634 var dstat aStat_t
1635 staterr := alstat(dir,&dstat) // should be fstatat
1636 if staterr != nil {
1637   if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1638   return total
1639 }
1640 //filever,err := ioutil.ReadDir(dir)
1641 //_err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1642 /*
1643   if err != nil {
1644     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1645   return total
1646 */
1647 if depth == 0 {
1648   total = sumFileInfo(total,dir,staterr,dstat,argv,true)
1649   if !nols && !isin("-e",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1650     showFileInfo(dir,argv)
1651   }
1652 }
1653 // it is not a directory, just scan it and finish
1654 for ei := 0 ; ei++ {
1655   entv,rdrerr := dirfile.Readdirnames(8*1024)
1656   if len(entv) == 0 || rdrerr != nil {
1657     //if rdrerr != nil { fmt.Printf("%d len=%d (%v)\n",ei,len(entv),rdrerr) }
1658     break
1659   }
1660   if 0 < ei {
1661     fmt.Printf("-I-- xxFind[%d] # large-dir: %s\n",ei,len(entv),dir)
1662   }
1663   total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1664   if isin("-du",argv) {
1665     // if in "du" mode
1666     fmt.Printf("%d\t%s\n", (total.Blocks-prev.Blocks)/2,dir)
1667   }
1668   return total
1669 }
1670 }
1671 // {ufind}fulis [Files] [/ Names] [-- Expressions]
1672 // Files is "*" by default
1673 // Names is "*" by default
1674 // Expressions is "-print" by default for "ufind", or -du for "fu" command
1675 func (gsh*GshContext)cmd(argv[]string){
1676   if 0 < len(argv) && argv[0] != "?" {
1677     showFound(gsh,argv)
1678   }
1679   if isin("-cksum",argv) || isin("-sum",argv) {
1680     gsh.lastCheckSum = CheckSum{
1681       if isin("-sum",argv) && isin("-add",argv) {
1682         gsh.lastCheckSum.SumType |= SUM_SUM64
1683       else
1684         if isin("-sum",argv) && isin("-size",argv) {
1685           gsh.lastCheckSum.SumType |= SUM_SIZE
1686         else
1687           if isin("-sum",argv) && isin("-bsd",argv) {
1688             gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1689           else
1690             if isin("-sum",argv) && isin("-sysv",argv) {
1691               gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1692             else
1693               if isin("-sum",argv) {
1694                 gsh.lastCheckSum.SumType |= SUM_SUM64
1695               else
1696                 if isin("-unix",argv) {
1697                   gsh.lastCheckSum.SumType |= SUM_UNIXFILE
1698                   gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1699                 }
1700                 if isin("-ieee",argv){
1701                   gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1702                   gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
1703                 }
1704               }
1705             }
1706             gsh.lastCheckSum.RusghtStart = Getrusagev()
1707           }
1708         }
1709       }
1710     }
1711   }
1712   var total = fileSum()
1713   npats := []string{}
1714   for _,v := range argv {
1715     if 0 < len(v) & v[0] != '-' {
1716       npats = append(npats,v)
1717     }
1718     if v == "/" { break }
1719     if v == "-" { break }
1720     if v == "-grep" { break }
1721     if v == "-ls" { break }
1722   }
1723   if len(npats) == 0 {
1724     npats = []string{"*"}
1725   }
1726   rwd := "."
1727   // if to be fullpath :: cwd, cwd := os.Getwd()
1728   if len(npats) == 0 { npats = []string{"*"} }
1729   fusage := gsh.xxFind(0,*total,cwd,npats,argv)
1730   if gsh.lastCheckSum.SumType != 0 {
1731     var sumi uint64 = 0
1732     sum := gsh.lastCheckSum
1733     if (sum.SumType + SUM_SIZE) != 0 {
1734       sumi = uint64(sum.Size)
1735     }
1736     if (sum.SumType + SUM_SUM64) != 0 {
1737       sumi = sum.Sum64
1738     }
1739     if (sum.SumType + SUM_SUM16_SYSV) != 0 {
1740       s := uint32(sum.Sum16)
1741       r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
1742       s = (r & 0xFFFF) + (r >> 16)
1743       sum.Crc32Val = uint32(s)
1744       sumi = uint64(s)
1745     }
1746     if (sum.SumType + SUM_SUM16_BSD) != 0 {
1747       sum.Crc32Val = uint32(sum.Sum16)
1748       sumi = uint64(sum.Sum16)
1749     }
1750     if (sum.SumType + SUM_UNIXFILE) != 0 {
1751       sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
1752       sumi = uint64(bytCRC32end(sum.Crc32Val,uint64(sum.Size)))
1753     }
1754     if 1 < sum.Files {
1755       fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1756         sumi,sum.Size,
1757         absSize(sum.Size),sum.Files,
1758         absSize(sum.Size/sum.Files))
1759     }else{
1760       fmt.Printf("%v %v %v\n",
1761         sumi,sum.Size,npats[0])
1762     }
1763   }
1764   if !isin("-grep",argv) {
1765     showUsage("total",fusage)
1766   }
1767   if !isin("-s",argv){
1768     hits := len(gsh.CmdCurrent.FindFile)
1769     if 0 < hits {
1770       fmt.Printf("-I-- #d files hits // can be refered with !fdf\n",
1771         hits,len(gsh.CommandHistory))
1772     }
1773   }
1774   if gsh.lastCheckSum.SumType != 0 {
1775     if isin("-ru",argv) {
1776       sum := &gsh.lastCheckSum
1777       sum.SumType = SUM_SIZE
1778       gsh.lastCheckSum.RusghtEnd = Getrusagev()
1779       elps := sum.Done.Sub(gsh.Start)
1780       fmt.Printf("-cksum-size: %v (%v) / %v files, %v/file\r\n",
1781         sum.Size,absSize(sum.Size),sum.Files,absSize(sum.Size/sum.Files))

```

```

1782     nanos := int64(elps)
1783     dnanos := time.Duration(nanos);
1784     fmt.Printf("--cpu-time: %v/total, %v/file, %.1f files/s, %v\r\n",
1785         abbtme(dnanos),
1786         abbtme(time.Duration(nanos/sum.Files)),
1787         (float64(sum.Files)*1000000000.0)/float64(nanos),
1788         abbspeed(sum.Size,nanos))
1789     diff := RusgSubv(sum.RusgPEnd,sum.RusgStart)
1790     fmt.Printf("--cksum-rusg: %v\n",sRusagef(argv,diff))
1791 }
1792 }
1793 return
1794 }
1795
1796 func showFiles(files[]string){
1797     sp := ""
1798     for i,file := range files {
1799         if 0 < i { sp = " " } else { sp = "" }
1800         fmt.Printf(sp+"%s",escapeWhiteSP(file))
1801     }
1802 }
1803 func showFound(gshCtx *GshContext, argv[]string){
1804     for i,v := range gshCtx.CommandHistory {
1805         if 0 < len(v.FoundFile) {
1806             fmt.Printf("%td (%d) ",i,len(v.FoundFile))
1807             if i == -1 {argv[0]}
1808             fmt.Println(" ")
1809             for ,file := range v.FoundFile {
1810                 fmt.Printf("%s //sub number?\n",showFileInfo(file,argv))
1811             }
1812         }else{
1813             showFiles(v.FoundFile)
1814             fmt.Println("\n")
1815         }
1816     }
1817 }
1818 }
1819
1820 func showMatchFile(filev []ios.FileInfo, npat,dir string, argv[]string)(string,bool){
1821     fname := ""
1822     found := false
1823     for i := range filev {
1824         match := filepath.Match(npata,(v.Name()))
1825         if match {
1826             fname = v.Name()
1827             found = true
1828             //fmt.Printf("%td) %s\n",i,v.Name())
1829             showIfExecutable(fname,dir,argv)
1830         }
1831     }
1832 }
1833 return fname,found
1834 }
1835 func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
1836     var fullpath string
1837     if !strings.HasPrefix(name,DIRSEP){
1838         fullpath = name
1839     }else{
1840         if len(dir) == 0 ){
1841             fullpath = name;
1842         }else{
1843             fullpath = dir + DIRSEP + name
1844         }
1845     fi, err := os.Stat(fullpath)
1846     //fmt.Printf("--Dp-- %v\n",fullpath,err);
1847     if err != nil {
1848         fullpath += ".exe";
1849         fi, err = os.Stat(fullpath)
1850     }
1851     if err != nil {
1852         fullpath = dir + DIRSEP + name + ".go"
1853         fi, err = os.Stat(fullpath)
1854     }
1855     if err == nil {
1856         fm := fi.Mode()
1857         if fm.IsRegular() {
1858             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1859             if fs := fullpath[:5] == nil {
1860                 fullpath = fullpath
1861                 ffound = true
1862                 if !isin("-s", argv) {
1863                     showFileInfo(fullpath,argv)
1864                 }
1865             }
1866         }
1867     }
1868     return ffullpath, ffound
1869 }
1870 func which(list string, argv []string) (fullpath []string, itis bool){
1871     if len(argv) <= 1 {
1872         fmt.Println("Usage: which comand [-s] [-a] [-ls]\n")
1873         return []string{}, false
1874     }
1875     path := argv[1]
1876     if strings.HasPrefix(path,'/'){
1877         // show check if executable?
1878         _exOK := showIfExecutable(path,"",argv)
1879         fmt.Printf("---- %v exOK=%v\n",path,_exOK)
1880         return []string{path},_exOK
1881     }
1882     pathenv, efound := os.LookupEnv(list)
1883     if ! efound {
1884         fmt.Println("----: which: no \"%s\" environment\n",list)
1885         return []string{}, false
1886     }
1887     //fmt.Println("PATH=%v\n",pathenv);
1888     shwall := isin("-s",argv) || 0 < strings.Index(path,"*")
1889     dirv := strings.Split(pathenv,PATHSEP)
1890     ffound := false
1891     ffullpath := path
1892     for _,dir := range dirv {
1893         if 0 < strings.Index(path,"*") { // by wild-card
1894             list := initial.ReadDir(dir)
1895             ffullpath, ffound = showMatchFile(list,dir,argv)
1896         }else{
1897             ffullpath, ffound = showIfExecutable(path,dir,argv)
1898         }
1899         //if ffound && !shwall {
1900         if ffound && !shwall {
1901             break;
1902         }
1903     }
1904     return []string{ffullpath}, ffound
1905 }
1906
1907 func stripLeadingWSParg(argv[]string)([]string){
1908     for ; 0 < len(argv); {
1909         if len(argv[0]) == 0 {
1910             argv = argv[1:]
1911         }else{
1912             break
1913         }
1914     }
1915     return argv
1916 }
1917 func xval(argv []string, nlen bool){
1918     argv = stripLeadingWSParg(argv)
1919     if len(argv) == 0 {
1920         fmt.Printf("eval [%#format] [Go-expression]\n")
1921         return
1922     }
1923     fmt := "%v"
1924     if argv[0][0] == '%' {
1925         fmt = argv[0]
1926         argv = argv[1:]
1927     }
1928     if len(argv) == 0 {
1929         return
1930     }
1931     gocode := strings.Join(argv," ");
1932     //fmt.Printf("eval [%v] [%v]\n",fmt,gocode)
1933     fset := token.NewFileSet()
1934     rval, _ := types.Eval(fmt,nil,fset,tok.NoPos,gocode)
1935     fmt.Println(fmt,rval.Value)
1936     if nlen { fmt.Println("\n") }
1937 }
1938
1939 func getval(name string) (found bool, val int) {
1940     /* should expand the name here */
1941     if name == "gsh.pid" {
1942         return true, os.Getpid()
1943     }else

```

```

1944     if name == "gsh.ppid" {
1945         return true, os.Getppid()
1946     }
1947     return false, 0
1948 }
1949
1950 func echo(argv []string, nlen bool){
1951     for ai := 1; ai < len(argv); ai++ {
1952         if i < ai {
1953             fmt.Printf(" ")
1954         }
1955         arg := argv[ai]
1956         found, val := getopt(arg)
1957         if found {
1958             fmt.Printf("%d",val)
1959         }else{
1960             fmt.Printf("%s",arg)
1961         }
1962     }
1963     if nlen {
1964         fmt.Printf("\n");
1965     }
1966 }
1967
1968 func resfile() string {
1969     return "gsh.tmp"
1970 }
1971 //var refF *File
1972 func remap() {
1973     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1974     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1975     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1976     if err != nil {
1977         fmt.Printf("refF could not open: %s\n",err)
1978     }else{
1979         fmt.Printf("refF opened\n")
1980     }
1981 }
1982
1983 // @@2020-0821
1984 func gshScanArg(str string,strip int)(argv []string){
1985     var si = 0
1986     var sb = 0
1987     var inBracket = 0
1988     var argl = make([]byte,LINESIZE)
1989     var ax = 0
1990     debug := false
1991
1992     for ; si < len(str); si++ {
1993         if str[si] != ' ' {
1994             break
1995         }
1996     }
1997     sb = si
1998     for ; si < len(str); si++ {
1999         if sb < si {
2000             if debug {
2001                 fmt.Printf("--Da- %d %d-%d %s ... %s\n",
2002                     inBracket,sb,si,argl[0:ax],str[si:])
2003             }
2004         ch := str[si]
2005         if ch == '(' {
2006             inBracket += 1
2007             if 0 < strip && inBracket <= strip {
2008                 //fmt.Println("stripLEV #d <= #d\n",inBracket,strip)
2009                 continue
2010             }
2011         }
2012         if 0 < inBracket {
2013             if ch == ')' {
2014                 inBracket -= 1
2015                 if 0 < strip && inBracket <= strip {
2016                     //fmt.Println("stripLEV #d <= #d\n",inBracket,strip)
2017                     continue
2018                 }
2019             }
2020         argl[sx] = ch
2021         sx += 1
2022         continue
2023     }
2024     if str[si] == ' ' {
2025         argv = append(argv,str(string(argl[0:ax])))
2026         if debug {
2027             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2028                 -1+len(argv),sb,si,str[sb:si],string(str[si:]))
2029         }
2030         sb = si+1
2031         ax = 0
2032         continue
2033     }
2034     argl[ax] = ch
2035     ax += 1
2036 }
2037
2038 if sb < si {
2039     argv = append(argv,str(string(argl[0:ax])))
2040     if debug {
2041         fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2042             -1+len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
2043     }
2044 }
2045 if debug {
2046     fmt.Printf("--Da- %d [%s] > [%d]%%\n",strip,str,len(argv),argv)
2047 }
2048 return argv
2049 }
2050
2051 // should get stderr (into tmpfile ?) and return
2052 func (gsh*GshContext)Open(name,mode string)(pin*os.File,pout*os.File,err bool){
2053     //var pv = [jint-1,-1]
2054     //syscall.Pipe(pv)
2055
2056     xargs := gshScanArg(name,1)
2057     name = strings.Join(xargs, " ")
2058
2059     //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-["+name+"]")
2060     //pout = os.NewFile(uintptr(pv[1]),"StdinOf-["+name+"]")
2061     pin,pout,_ = os.Pipe();
2062
2063     fidx := 0
2064     dir := "?"
2065     if mode == "r" {
2066         dir = "<"
2067     }
2068     fidx = 1 // read from the stdout of the process
2069     if mode == "w" {
2070         dir = ">"
2071         fidx = 0 // write to the stdin of the process
2072     }
2073     gshPA := gsh.gshPA
2074     savfd := gshPA.Files[fidx]
2075
2076     var fd uintptr = 0
2077     if mode == "r" {
2078         //fd = pout.Fd()
2079         //gshPA.Files[fidx] = pout.Fd()
2080         gshPA.Files[fidx] = pout;
2081     }else{
2082         //fd = pin.Fd()
2083         //gshPA.Files[fidx] = pin.Fd()
2084         gshPA.Files[fidx] = pin;
2085     }
2086     // should do this by Goroutine?
2087     if false {
2088         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2089         fmt.Printf("--REDI [%d,%d,%d]>[%d,%d,%d]\n",
2090             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2091             pin.Fd(),pout.Fd(),pout.Fd())
2092     }
2093     sav1 := os.Stdin
2094     sav2 := os.Stdout
2095     sav3 := os.Stderr
2096     os.Stdout = pin
2097     os.Stderr = pout
2098     gsh.BackGround = true
2099     gsh.gshellLlh(name)
2100     gsh.BackGround = false
2101     os.Stdin = sav1
2102     os.Stdout = sav2
2103     os.Stderr = sav3
2104
2105     gshPA.Files[fidx] = savfd

```

```

2106     return pin,pout,false
2107   }
2108
2109 // <a name="ex-commands">External commands</a>
2110 func (gsh*GahContext)execCommand(exec bool, argv []string) (notf bool,exit bool) {
2111   if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
2112
2113   gshPA := gsh.gshPA
2114   fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
2115   if itis == false {
2116     return true,true
2117   }
2118   fullpath := fullpath[0]
2119   argv = unescapeapesh(argv)
2120   if 0 < len(argv)-1 {
2121     argv[0] = argv[1]
2122     argv[1] = argv[2]
2123     if itis == false {
2124       fmt.Printf("--F-- Go not found\n")
2125       return false,true
2126     }
2127     gofullpath := gofullpath[0]
2128     argv = unescapeapesh(argv)
2129     fmt.Printf("--I-- %s %s\n",gofullpath,
2130     argv[0],argv[1],argv[2])
2131     if exec {
2132       syscall.Exec(gofullpath,argv,os.Environ())
2133     }else{
2134       pid, _ := syscall.ForkExec(gofullpath,argv,&gshPA);
2135       proc,_ := os.StartProcess(gofullpath,argv,&gshPA);
2136       pstat,_ := proc.Wait();
2137       pid = pstat.Pid();
2138       if gsh.BackGround {
2139         fmt.Fprintf(stderr,"--Ip- in Background pid(%d)%d(%v)\n",pid,len(argv),argv)
2140         gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2141       }
2142     }else{
2143       rusage := aRusage {}
2144       syscall.Wait4(pid,nil,0,&rusage)
2145       gsh.LastUsage = rusage
2146       gsh.CmdCurrent.Rusagev[1] = rusage
2147     }
2148   }
2149   /*
2150   gsh.LastUsage = *pstat.SysUsage().(*aRusage);
2151   gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2152   */
2153   /*
2154   aSetRusage(&gsh.LastRusage,pstat);
2155   gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2156   */
2157   else{
2158     if exec {
2159       syscall.Exec(fullpath,argv,os.Environ())
2160     }else{
2161       pid, _ := syscall.ForkExec(fullpath,argv,&gshPA);
2162       proc,_ := os.StartProcess(fullpath,argv,&gshPA);
2163       pstat,_ := proc.Wait();
2164       pid := pstat.Pid();
2165     }
2166     //fmt.Printf("%d(%v)",pid); // '&' to be background
2167   if( false )
2168   fmt.Printf("Sys=%v\n",gshPA.Sys);
2169   if( gshPA.Sys != nil ){
2170     //fmt.Printf("inFG=%v\n",gshPA.Sys.Foreground);
2171   }
2172   if( gsh.BackGround {
2173     fmt.Fprintf(stderr,"--Ip- in Background pid(%d)%d(%v)\n",pid,len(argv),argv)
2174     gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2175   }
2176   gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
2177 }else{
2178   /*
2179   rusage := aRusage {}
2180   syscall.Wait4(pid,nil,0,&rusage);
2181   gsh.LastUsage = rusage
2182   gsh.CmdCurrent.Rusagev[1] = rusage
2183   */
2184   /*
2185   gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2186   gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2187   */
2188   aSetRusage(&gsh.LastRusage,pstat);
2189   gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2190   */
2191 }
2192 }
2193 return false,false
2194
2195 // <a name="builtin">Builtin Commands</a>
2196 func (gshctx *GshContext) sleep(argv []string) {
2197   if len(argv) < 2 {
2198     fmt.Println("Sleep 100ms, 100us, 100ns, ...\n")
2199     return
2200   }
2201   duration := argv[1];
2202   d, err := time.ParseDuration(duration)
2203   if err != nil {
2204     d, err = time.ParseDuration(duration+"s")
2205   }
2206   if err != nil {
2207     fmt.Printf("duration ? %s (%s)\n",duration,err)
2208     return
2209   }
2210   //fmt.Println("Sleep %v\n",duration)
2211   time.Sleep(d)
2212   if 0 < len(argv[2:]) {
2213     gshctx.gshelv(argv[2:])
2214   }
2215 }
2216 func (gshctx *GshContext) repeat(argv []string) {
2217   if len(argv) < 2 {
2218     return
2219   }
2220   start0 := time.Now()
2221   for ri,_ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2222     if 0 < len(argv[1]) {
2223       //start := time.Now()
2224       gshctx.gshelv(argv[2:])
2225       end := time.Now()
2226       elps := end.Sub(start0);
2227       if( 1000000000 < elps ) {
2228         fmt.Printf("(repeat)%d %v)\n",ri,elps);
2229       }
2230     }
2231   }
2232 }
2233 }
2234
func (gshctx *GshContext) gen(argv []string) {
2235   gshPA := gshctx.gshPA
2236   if len(argv) < 2 {
2237     fmt.Printf("Usage: %s N\n",argv[0])
2238     return
2239   }
2240   // should br repeated by "repeat" command
2241   count := strconv.Atoi(argv[1])
2242   //fd := gshPA.Files[1] // Stdout
2243   //file := os.NewFile(fd,"internalstdout")
2244   file := gshPA.Files[1]; // Stdout
2245   file.Truncate(0).Gen. Count=fd to [%d]\n",count,file.Fd())
2246   //buf := [byte]{} Gen. Count=fd to [%d]\n",count,file.Fd())
2247   outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2248   for gi := 0; gi < count; gi++ {
2249     file.WriteString(outdata)
2250   }
2251   //file.WriteString("\n")
2252   fmt.Printf("\n%d B\n",count*len(outdata));
2253   //file.Close()
2254 }
2255 }
2256
// <a name="remote">Remote Execution</a> // 2020-0820
2257 func (gshctx *GshContext) remote(argv []string){
2258   elapsed := time.Now().SubFrom()
2259   elps := time.Now().SubFrom()
2260   if 1000000000 < elps {
2261     return fmt.Sprintf("[%5d.%02ds]",elps/1000000000,(elps%1000000000)/10000000)
2262   }else{
2263     if 1000000 < elps {
2264       return fmt.Sprintf("[%3d.%03dms]",elps/1000000,(elps%1000000)/1000)
2265     }else{
2266       return fmt.Sprintf("[%3d.%03duse]",elps/1000,(elps%1000))
2267     }
2268 }

```

```

2268 }
2269 //func abstime(nanos int64)(string){
2270 func abstime(nanos time.Duration)(string){
2271     if 1000000000 < nanos {
2272         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%100000000)/10000000)
2273     }else{
2274         if 100000 < nanos {
2275             return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
2276         }else{
2277             return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
2278         }
2279     }
2280 func abssize(size int64)(string){
2281     fsize := float64(size)
2282     if 1024*1024*1024 < size {
2283         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2284     }else{
2285         if 1024*1024 < size {
2286             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2287         }else{
2288             return fmt.Sprintf("%.3fKiB",fsize/1024)
2289         }
2290     }
2291 func absize(size int64)(string){
2292     fsize := float64(size)
2293     if 1024*1024*1024 < size {
2294         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2295     }else{
2296         if 1024*1024 < size {
2297             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2298         }else{
2299             return fmt.Sprintf("%.3fKiB",fsize/1024)
2300         }
2301 }
2302 func abbspeed(totalB int64,ns int64)(string){
2303     MBS := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2304     if 1000 <= MBS {
2305         return fmt.Sprintf("%.3fGB/s",MBS/1000)
2306     }else{
2307         if 1 <= MBS {
2308             return fmt.Sprintf("%.3fMB/s",MBS)
2309         }else{
2310             return fmt.Sprintf("%.3fKB/s",MBS*1000)
2311         }
2312     }
2313 func abbspeed(totalB int64,ns time.Duration)(string){
2314     MBS := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2315     if 1000 <= MBS {
2316         return fmt.Sprintf("%.3fGBps",MBS/1000)
2317     }else{
2318         if 1 <= MBS {
2319             return fmt.Sprintf("%.3fMBps",MBS)
2320         }else{
2321             return fmt.Sprintf("%.3fKBps",MBS*1000)
2322         }
2323     }
2324 func fileRelay(what string,in*os.File,out*os.File,size int64,bsz int)(wcount int64){
2325     Start := time.Now()
2326     buff := make([]byte,bsz)
2327     var total int64 = 0
2328     var rem int64 = size
2329     nio := 0
2330     Prev := time.Now()
2331     var PreSize int64 = 0
2332     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
2333             what,absize(total),size,nio)
2334     for i:= 0 ; i++ {
2335         var len = bsz
2336         if int(rem) < len {
2337             len = int(rem)
2338         }
2339         Now := time.Now()
2340         Elps := Now.Sub(Prev);
2341         if 1000000000 < Now.Sub(Prev) {
2342             fmt.Println(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
2343             what,absize(total),size,lio,
2344             abbspeed((total-PreSize),Elps))
2345         }
2346         Prev = Now;
2347         PreSize = total
2348         rlen := len
2349         if in != nil {
2350             // should watch the disconnection of out
2351             rcc,err := in.Read(buff[0:rlen])
2352             if err != nil {
2353                 fmt.Println(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
2354                     what,rcc,err,in.Name())
2355                 break
2356             }
2357             rlen = rcc
2358             if string(buff[0:10]) == "((SoftEOF " {
2359                 var ecc int
2360                 fmt.Scanf(string(buff),"(SoftEOF %v",&ecc)
2361                 fmt.Println(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))%v\n",
2362                     what,ecc,total)
2363                 if ecc == total {
2364                     break
2365                 }
2366             }
2367         }
2368         wlen := rlen
2369         if out != nil {
2370             wcc,err := out.Write(buff[0:rlen])
2371             if err != nil {
2372                 fmt.Println(Elapsed(Start)+"--En- X: %s write(%v,%v)>%v\n",
2373                     what,wcc,err,out.Name())
2374                 break
2375             }
2376             wlen = wcc
2377         }
2378         if wlen < rlen {
2379             fmt.Println(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2380                     what,wlen,rlen)
2381             break;
2382         }
2383         nio += 1
2384         total += int64(rlen)
2385         rem -= int64(rlen)
2386         if rem <= 0 {
2387             break
2388         }
2389     }
2390     Done := time.Now()
2391     Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2392     Total := float64(total)/1000000 //MB
2393     Lmt := Total/Elps
2394     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) tv %.3fMB/s\n",
2395             what,total,size,nio,absize(total),MBps)
2396     return total
2397 }
2398 func tcppush(clnt *os.File){
2399     // shrink socket buffer and recover
2400     usleep(100);
2401 }
2402 func (gsh*GshContext)ReexecServer(argv[]string{
2403     debug := true
2404     Start := time.Now()
2405     Start := Start
2406     if local == ":" { local = "0.0.0.0:9999" }
2407     local := "0.0.0.0:9999"
2408     if 0 < len(argv) {
2409         if argv[0] == "-s" {
2410             debug = false
2411             argv = argv[1:]
2412         }
2413     }
2414     if 0 < len(argv) {
2415         argv = argv[1];
2416     }
2417     port, err := net.ResolveTCPAddr("tcp",local);
2418     if err != nil {
2419         fmt.Println("--En- S: Address error: %s(%s)\n",local,err)
2420         return
2421     }
2422     sconn, err := net.ListenTCP("tcp", port)
2423     if err != nil {
2424         fmt.Println(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2425     }
2426     sconn, err := net.ListenTCP("tcp", port)
2427     if err != nil {
2428

```

```

2430     fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2431     return
2432   }
2433 
2434   reqbuf := make([]byte,LINESIZE)
2435   res := ""
2436   for {
2437     fmt.Println(Elapsed(Start)+"--In- S: Listening at "+s+"\\n",local);
2438     sconn, err := sconn.AcceptTCP()
2439     Start := time.Now()
2440     if err != nil {
2441       fmt.Println(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2442       return
2443     }
2444     cln := aconn.File()
2445     fd := cln.Fd()
2446     ar := aconn.RemoteAddr()
2447     if debug { fmt.Println(Elapsed(Start0)+"--In- S: Accepted TCP at "+s+" ["+d+"] <- "+v+"\n",
2448       local,fd,ar)
2449     res := fmt.Sprintf("%20s Gshell/%s Server\\r\\n",VERSION)
2450     fmt.Println(ar+":",res)
2451     if debug { fmt.Println(Elapsed(Start)+"--In- S: "+s+",res) }
2452     count, err := cln.Read(reqbuf)
2453     if err != nil {
2454       fmt.Println(Elapsed(Start)+"--En- C: ("+v+" "+v)+")\n",
2455       count,err,string(reqbuf))
2456     req := string(reqbuf[:count])
2457     if debug { fmt.Println(Elapsed(Start)+"--In- C: "+v",string(req)) }
2458     req := strings.Split(string(req),"\\r")
2459     cmdv := gshScanArg(reqv[0],0)
2460     //cmdv := gshArgs.Split(reqv[0]," ")
2461     switch cmdv[0] {
2462     case "HELLO":
2463       res = fmt.Sprintf("250 %v",req)
2464     case "GET":
2465       download {remotefile[-2N]} [localfile]
2466       var dsize int64 = 32*1024*1024
2467       var bsize int = 64*1024
2468       var fname string = ""
2469       var in *os.File = nil
2470       var pseudoEOF = false
2471       if l < len(cmdv) {
2472         fname = cmdv[1]
2473         if strBegins(fname,"-z") {
2474           fmt.Sscanf(cmdv[1],"%d",&dsize)
2475         }else{
2476           if strBegins(fname,"-") {
2477             xin,xout,err := gsh.Popen(fname,"r")
2478             if err != nil {
2479               if xin != nil {
2480                 defer xin.Close()
2481               if xout != nil {
2482                 defer xout.Close()
2483               }
2484             }
2485             dsiz = MaxStreamSize
2486             pseudoEOF = true
2487           }else{
2488             xin,err := os.Open(fname)
2489             if err != nil {
2490               fmt.Println("--En- GET "+v+"\n",err)
2491             }else{
2492               defer xin.Close()
2493               in = xin
2494               fi := xin.Stat()
2495               dsiz = fi.Size()
2496             }
2497           }
2498         }
2499       }
2500       res = fmt.Sprintf("200 %v\\r\\n",dsiz)
2501       fmt.Println("In-",s,"-",v)
2502       tcpPush(cln); // should be separated as line in receiver
2503       fmt.Println(Elapsed(Start)+"--In- S: "+v+",res)
2504       wcount := fileRelay("SendGET",in,clnt,dsiz,bsize)
2505       if pseudoEOF {
2506         in.Close() // close from the command
2507         // show end of stream data (its size) by EOF?
2508         SoftEOF := fmt.Sprintf("(SoftEOF %v)",wcount)
2509         fmt.Println(Elapsed(Start)+"--In- S: Send "+v+"\n",SoftEOF)
2510       }
2511       tcpPush(clnt); // let SoftEOF data apper at the top of received data
2512       fmt.Println("In-",s,"-",v,"SoftEOF")
2513       tcpPush(cln); // client will close alone in a packet (separate with 200 OK)
2514       // with client generated random?
2515       //fmt.Println("--In- L: close "+v+"\\n",in.Fd(),in.Name())
2516     }
2517     res = fmt.Sprintf("200 GET done\\r\\n")
2518   case "PUT":
2519     // upload {srcfile[-2N]} [dstfile]
2520     var dsize int64 = 32*1024*1024
2521     var bsize int = 64*1024
2522     var fname string = ""
2523     var out *os.File = nil
2524     if l < len(cmdv) { // localfile
2525       fmt.Sscanf(cmdv[1],"%d",&dsize)
2526     }
2527     if 2 < len(cmdv) {
2528       fname = cmdv[2]
2529       if fname == "-" {
2530         // nul dev
2531       }else{
2532         if strBegins(fname,"-") {
2533           xin,xout,err := gsh.Popen(fname,"w")
2534           if err {
2535             if xin != nil {
2536               defer xout.Close()
2537               out = xout
2538             }
2539           }else{
2540             // should write to temporary file
2541             // should supports '0' or '0x00'
2542             xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2543             //fmt.Println("--In- S: open("+v+") out("+v+"),err("+v+"),fname,xout,err)
2544             if err != nil {
2545               fmt.Println("--En- PUT "+v+"\n",err)
2546             }else{
2547               out = xout
2548             }
2549           }
2550         }
2551       fmt.Println(Elapsed(Start)+"--In- L: open("+v+",w) "+v+"\\n",
2552         fname,local,err)
2553     }
2554     fmt.Println(Elapsed(Start)+"--In- PUT "+v+"\\n",dsiz)
2555     fmt.Println(Elapsed(Start)+"--In- S: 200 "+v+" OK\\r\\n",dsiz)
2556     fmt.Println("200 "+v+" OK\\r\\n",dsiz)
2557     fileRecvPUT(clnt,out,dsiz,bsize)
2558     res = fmt.Sprintf("200 PUT done\\r\\n")
2559   default:
2560     res = fmt.Sprintf("400 What? "+v",req)
2561   }
2562   swcc,serr := clnt.Write([]byte(res))
2563   if serr != nil {
2564     fmt.Println(Elapsed(Start)+"--In- S: (wc="+v+" er="+v+" "+swcc,serr,res)
2565   }else{
2566     fmt.Println(Elapsed(Start)+"--In- S: "+v+",res)
2567   }
2568   aconn.Close();
2569   cint.Close();
2570 }
2571 sconn.Close();
2572 }
2573 func (gsh*GshContext)RexecClient(argv[]string)(int,string{
2574   debug := true
2575   Start := time.Now()
2576   if len(argv) == 1 {
2577     return -1, "EmptyARG"
2578   }
2579   argv = argv[1:]
2580   if argv[0] == "-serv" {
2581     gsh.RexecServer(argv[1:])
2582     return 0, "Server"
2583   }
2584   remote := "0.0.0.0:9999"
2585   if argv[0][0] == '@' {
2586     remote = argv[0][1:]
2587     argv = argv[1:]
2588   }
2589   if argv[0] == "-g" {
2590     debug = false
2591     argv = argv[1:]
2592   }
2593 })

```

```

2592 }
2593 dport, err := net.ResolveTCPAddr("tcp", remote);
2594 if err != nil {
2595     fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2596     return -1,"AddressError"
2597 }
2598 fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2599 serv, err := net.DialTCP("tcp",nil,dport)
2600 if err != nil {
2601     fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
2602     return -1,"CannotConnect"
2603 }
2604 if debug {
2605     al := serv.LocalAddr()
2606     fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2607 }
2608 req := ""
2609 res := make([]byte,LINESIZE)
2610 counter,err := serv.Read(res)
2611 if err != nil {
2612     fmt.Printf("--En- S: (%d,%v) %v",count,err,string(res))
2613 }
2614 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }
2615 if argv[0] == "GET" {
2616     serv := gsh.gshPA
2617     var bsize int = 64*1024
2618     req = fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2619     fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
2620     fmt.Fprintf(serv,req)
2621     count,err = serv.Read(res)
2622     if err != nil {
2623         if err != nil {
2624             var dszie int64 = 0
2625             var out *os.File = nil
2626             var out_tobeclosed *os.File = nil
2627             var fname string = ""
2628             var rcode int
2629             var pid int = -1
2630             fmt.Sscanf(string(res),"td %d",&rcode,&dszie)
2631             fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2632             if 3 < len(argv) {
2633                 if strBegins(argv[2],") {
2634                     xin,xout,err := gsh.Popen(fname,"w")
2635                     if err {
2636                         xin.Close()
2637                         defer xout.Close()
2638                         out = xout
2639                         out_tobeclosed = xout
2640                         pid = 0 // should be its pid
2641                     } else {
2642                         // should write to temporary file
2643                         // should suppress "C on tty"
2644                         xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2645                         if err != nil {
2646                             fmt.Println("--En- %v\n",err)
2647                         } else {
2648                             out = xout
2649                             //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2650                         }
2651                     }
2652                     in := serv.File()
2653                     fileRelay("RecvGET",in,out,dszie,bsize)
2654                     if 0 < pid {
2655                         gsh.gshPA = savPA // recovery of Fd(), and more?
2656                         fmt.Println(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2657                         out_tobeclosed.Close()
2658                         //syscall.Wait4(pid,nil,0,nil) //@@
2659                     }
2660                 }
2661             }
2662             if argv[0] == "PUT" {
2663                 remote, _ := serv.File()
2664                 var local *os.File = nil
2665                 var dszie int64 = 32*1024*1024
2666                 var bsize int = 64*1024
2667                 var ofile string = "-"
2668                 //fmt.Println("--I-- Rex %v\n",argv)
2669                 if 1 < len(argv) {
2670                     if strBegins(argv[1],"-z") {
2671                         fmt.Sscanf(fname[2:], "%d",&dszie)
2672                     } else {
2673                         if strBegins(argv[1],"{") {
2674                             xin,xout,err := gsh.Popen(fname,"r")
2675                             if err {
2676                                 xin.Close()
2677                                 defer xin.Close()
2678                                 local = xin
2679                                 fmt.Println("--In- %d < Upload output of %v\n",
2680                                         local.Fd(),fname)
2681                                 ofile = "-fromc."fname
2682                                 dszie = MaxStreamSize
2683                             } else {
2684                                 xlocal,err := os.Open(fname)
2685                                 if err != nil {
2686                                     fmt.Println("--En- (%s)\n",err)
2687                                 } else {
2688                                     local = xlocal
2689                                     fi := local.Stat()
2690                                     dszie = fi.Size()
2691                                     defer local.Close()
2692                                     //fmt.Println("--I-- Rex in(%v / %v)\n",ofile,dszie)
2693                                     if file = fname
2694                                         fmt.Println(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2695                                         fname,dszie,local,err)
2696                                     }
2697                                 }
2698                             }
2699                         }
2700                         if 2 < len(argv) && argv[2] != "" {
2701                             ofile = argv[2]
2702                             //fmt.Println("(d)%v B.ofile=%v\n",len(argv),argv,ofile)
2703                         }
2704                     }
2705                     if 2 < len(argv) && argv[2] != "" {
2706                         if debug { fmt.Println(Elapsed(Start)+"--I-- Rex out(%v)\n",file)
2707                         }
2708                         if debug { fmt.Println(Elapsed(Start)+"--In- PUT %v (%v)\n",dszie,bsize)
2709                         }
2710                         if debug { fmt.Println(Elapsed(Start)+"--In- R %v (%v)\n",dszie,file)
2711                         }
2712                         if debug { fmt.Println(Elapsed(Start)+"--In- C %v",req)
2713                         }
2714                         count,err = serv.Read(res)
2715                         if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2716                         }
2717                         if debug { fmt.Println(Elapsed(Start)+"--In- sending RexRequest(%v)\n",len(req))
2718                         }
2719                         fileRelay("SendPUT",local,remote,dszie,bsize)
2720                     }
2721                     if debug { fmt.Println(Elapsed(Start)+"--In- waiting RexResponse...\n")
2722                     }
2723                     count,err = serv.Read(res)
2724                     res := ""
2725                     if count == 0 {
2726                         res = "(nil)\r\n"
2727                     } else {
2728                         res = string(res[:count])
2729                     }
2730                     if err != nil {
2731                         fmt.Println(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
2732                     }
2733                     if err != nil {
2734                         fmt.Println(Elapsed(Start)+"--En- C: %v",req)
2735                     }
2736                     if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",res)
2737                     }
2738                     serv.Close()
2739                     //conn.Close()
2740                 }
2741                 var stat string
2742                 if err != nil {
2743                     fmt.Println(Elapsed(Start)+"--D- Client: %v (%v)",rcode,stat)
2744                 }
2745                 rcode,stat = res,stat
2746             }
2747             return rcode,stat
2748         }
2749     }
2750 // <a name="remote-sh">Remote Shell</a>
2751 // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
2752 func (gsh*GshContext)FileCopy(argv[]string){
2753     var host = ""

```

```

2754 var port = ""
2755 var upload = false
2756 var download = false
2757 var argc = []string{"rex-gcp"}
2758 var srcv = []string{}
2759 var dstv = []string{}
2760 argv = argv[1:]
2761
2762 for v := range argv {
2763     if v[0] == '-' { // might be a pseudo file (generated date)
2764         continue
2765     }
2766     obj := strings.Split(v,":")
2767     //fmt.Println("fd $v $v\n",len(obj),v,obj)
2768     if 1 < len(obj) {
2769         host = obj[0]
2770         file := ""
2771         if 0 < len(host) {
2772             gsh.LastServer.host = host
2773         }else{
2774             host = gsh.LastServer.host
2775             port = gsh.LastServer.port
2776         }
2777         if 2 < len(obj) {
2778             port = obj[1]
2779             if 0 < len(port) {
2780                 gsh.LastServer.port = port
2781             }else{
2782                 port = gsh.LastServer.port
2783             }
2784             file = obj[2]
2785         }else{
2786             file = obj[1]
2787         }
2788         if len(srcv) == 0 {
2789             download = true
2790             srcv = append(srcv,file)
2791             continue
2792         }
2793         upload = true
2794         dstv = append(dstv,file)
2795         continue
2796     }/*
2797     idx := strings.Index(v,":")
2798     if 0 <= idx {
2799         remote := v[0:idx]
2800         if len(remote) == 0 {
2801             download = true
2802             srcv = append(srcv,v[idx+1:])
2803             continue
2804         }
2805         upload = true
2806         dstv = append(dstv,v[idx+1:])
2807         continue
2808     }/*
2809     if download {
2810         dstv = append(dstv,v)
2811     }else{
2812         srcv = append(srcv,v)
2813     }
2814 }
2815 hostport := "*" + host + ":" + port
2816 if upload {
2817     if host != "" { argv = append(argv,hostport) }
2818     argv = append(argv,"PUT")
2819     argv = append(argv,srcv[0:1])
2820     argv = append(argv,dstv[0:1])
2821     //fmt.Printf("--I-- FileCopy PUT gsh://$v<$v> $v // $v\n",hostport,dstv,srcv,xargv)
2822     fmt.Println("--I-- FileCopy PUT gsh://$v<$v> $v // $v\n",hostport,dstv,srcv)
2823     gsh.RexecClient(argv)
2824 }else{
2825     if download {
2826         if host != "" { argv = append(argv,hostport) }
2827         argv = append(argv,"GET")
2828         argv = append(argv,srcv[0:1])
2829         argv = append(argv,dstv[0:1])
2830         //fmt.Printf("--I-- FileCopy GET gsh://$v<$v> $v // $v\n",hostport,srcv,dstv,xargv)
2831         fmt.Println("--I-- FileCopy GET gsh://$v<$v> $v // $v\n",hostport,srcv,dstv)
2832         gsh.RexecClient(argv)
2833     }
2834 }
2835 }
2836 */
2837 func (gsh*GshContext)Trelpath(rloc string)(string){
2838     cwd := os.Getwd()
2839     os.Cchdir(gsh.RND)
2840     os.Cchdir(rloc)
2841     twd, _ := os.Getwd()
2842     os.Cchdir(cwd)
2843     rpath := twd + "/" + rloc
2844     return rpath
2845 }
2846 // join to remote GShell - [user@host[:port] or cd host[:port]:path
2847 func (gsh*GshContext)Rjoin(argv[]string){
2848     if len(argv) <= 1 {
2849         fmt.Printf("--I-- current server = $v\n",gsh.RSERV)
2850         return
2851     }
2852     serv := argv[1]
2853     servv := strings.Split(serv,":")
2854     if 1 < len(servv) {
2855         if servv[0] == "lo" {
2856             servv[0] = "localhost"
2857         }
2858     }
2859     switch len(servv) {
2860     case 1:
2861         //if strings.Index(serv,":") < 0 {
2862             serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2863         //}
2864     case 2: // host:port
2865         serv = strings.Join(servv,:)
2866     }
2867     argv := []string{"rex-join","$"+serv,"HELO"}
2868     rcode,stat := gsh.RexecClient(argv)
2869     if (rcode / 100) == 2 {
2870         fmt.Printf("--I-- OK Joined ($v) $v\n",rcode,stat)
2871         gsh.RSERV = serv
2872     }else{
2873         fmt.Printf("--I-- NG, could not joined ($v) $v\n",rcode,stat)
2874     }
2875 }
2876 func (gsh*GshContext)Rexec(argv[]string){
2877     if len(argv) < 1 {
2878         fmt.Printf("--I-- rexec command [ | {file || {command} ]\n",gsh.RSERV)
2879     }
2880     return
2881 }
2882 /*
2883 argv := gshScanArg(strings.Join(argv," "),0)
2884 fmt.Println("--D-- nargc=$d $v\n",len(argv),nargc)
2885 if argv[1][0] != '{' {
2886     argv[1] = "(" + argv[1] + ")"
2887     fmt.Println("--D-- nargc=$d $v\n",len(argv),nargc)
2888 }
2889 argv = argv[1:]
2890 */
2891 argv := []string{"rex-exec","$"+gsh.RSERV,"GET"}
2892 argv = append(argv,argv...)
2893 argv = append(argv,"/dev/tty")
2894 rcode,stat := gsh.RexecClient(argv)
2895 if (rcode / 100) == 2 {
2896     fmt.Printf("--I-- OK Rexec ($v) $v\n",rcode,stat)
2897 }else{
2898     fmt.Printf("--I-- NG Rexec ($v) $v\n",rcode,stat)
2899 }
2900 */
2901 }
2902 func (gsh*GshContext)Rhmdir(argv[]string){
2903     if len(argv) <= 1 {
2904         return
2905     }

```

```

2916 cwd, _ := os.Getwd()
2917 os.Chdir(gsh.RWD)
2918 os.Chdir(argv[1])
2919 twd = cwd.os.Getwd()
2920 gsh.RWD = twd
2921 fmt.Printf("--I-- JWD=%v\n", twd)
2922 os.Chdir(cwd)
2923 }
2924 func (gsh*GshContext)Rpwd(argv[]string){
2925 fmt.Println(argv[0],gsh.RWD)
2926 }
2927 func (gsh*GshContext)Rls(argv[]string){
2928 cwd := os.Getwd()
2929 os.Cd(gsh.RWD)
2930 argv[0] = cwd
2931 gsh.Xbindargs(argv)
2932 os.Cd(cwd)
2933 }
2934 func (gsh*GshContext)Rput(argv[]string){
2935 var local string = ""
2936 var remote string = ""
2937 if 1 < len(argv) {
2938 local = argv[1]
2939 remote = local // base name
2940 }
2941 if 2 < len(argv) {
2942 remote = argv[2]
2943 }
2944 fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trelpath(remote))
2945 }
2946 func (gsh*GshContext)Rget(argv[]string){
2947 var remote string = ""
2948 var local string = ""
2949 if 1 < len(argv) {
2950 remote = argv[1]
2951 local = remote // base name
2952 }
2953 if 2 < len(argv) {
2954 local = argv[2]
2955 }
2956 fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trelpath(remote),local)
2957 }
2958 // <a name="network">network</a>
2959 // -s, -sl, -so // bi-directional, source, sync (maybe socket)
2960 func (gshCtx*GshContext)connect(intCP bool, argv []string) {
2961 gshPA := gshCtx.gshPA
2962 if len(argv) < 2 {
2963 fmt.Println("Usage: -s [host]:[port].[udp]\n")
2964 return
2965 }
2966 remote := argv[1]
2967 if remote == ":" { remote = "0.0.0.0:9999" }
2968 if intCP { // TCP
2969 dport, err := net.ResolveTCPAddr("tcp",remote);
2970 if err != nil {
2971 fmt.Println("Address error: %s (%s)\n",remote,err)
2972 return
2973 }
2974 conn, err := net.DialTCP("tcp",nil,dport)
2975 if err != nil {
2976 fmt.Println("Connection error: %s (%s)\n",remote,err)
2977 return
2978 }
2979 file, _ := conn.File();
2980 //fd := file.Fd();
2981 //fmt.Println("Socket: connected to %s, socket[%d]\n",remote,file.Fd())
2982 fmt.Println("Socket: connected to %s, socket[%d]\n",remote,conn)
2983 }
2984 safd := gshPA.Files[1]
2985 //gshPA.File[1] = fd;
2986 gshPA.File[1] = file;
2987 gshCtx.gshelly(argv[2])
2988 gshPA.Files[1] = safd
2989 file.Close()
2990 conn.Close()
2991 } else {
2992 //dport, err := net.ResolveUDPAddr("udp4",remote);
2993 dport, err := net.ResolveUDPAddr("udp",remote);
2994 if err != nil {
2995 fmt.Println("Address error: %s (%s)\n",remote,err)
2996 return
2997 }
2998 //conn, err := net.DialUDP("udp4",nil,dport)
2999 conn, err := net.DialUDP("udp",nil,dport)
3000 if err != nil {
3001 fmt.Println("Connection error: %s (%s)\n",remote,err)
3002 return
3003 }
3004 file, _ := conn.File();
3005 //fd := file.Fd();
3006 file, _ := conn.File();
3007 //fd := file.Fd();
3008 ar := conn.RemoteAddr()
3009 //ar := conn.LocalAddr()
3010 //fmt.Println("Socket: connected to %s [%s], socket[%d]\n",
3011 //remote,ar.String(),fd)
3012 remote,ar.String(),file.Fd())
3013 }
3014 safd := gshPA.Files[1]
3015 //gshPA.File[1] = fd;
3016 gshPA.File[1] = file;
3017 gshCtx.gshelly(argv[2])
3018 gshPA.Files[1] = safd
3019 file.Close()
3020 conn.Close()
3021 }
3022 }
3023 func (gshCtx*GshContext)saccept(intCP bool, argv []string) {
3024 gshPA := gshCtx.gshPA
3025 if len(argv) < 2 {
3026 fmt.Println("Usage: -ac [host]:[port].[udp]\n")
3027 return
3028 }
3029 local := argv[1]
3030 if local == ":" { local = "0.0.0.0:9999" }
3031 if intCP { // TCP
3032 port, err := net.ResolveTCPAddr("tcp",local);
3033 if err != nil {
3034 if err != nil {
3035 fmt.Println("Address error: %s (%s)\n",local,err)
3036 return
3037 }
3038 //fmt.Println("Listen at %s...\n",local);
3039 sconn, err := net.ListenTCP("tcp", port)
3040 if err != nil {
3041 fmt.Println("Listen error: %s (%s)\n",local,err)
3042 return
3043 }
3044 //fmt.Println("Accepting at %s...\n",local);
3045 aconn, err := sconn.AcceptTCP();
3046 if err != nil {
3047 fmt.Println("Accept error: %s (%s)\n",local,err)
3048 return
3049 }
3050 file, _ := aconn.File()
3051 //fd := file.Fd();
3052 //fmt.Println("Accepted TCP at %s [%d]\n",local,fd)
3053 fmt.Println("Accepted TCP at %s [%d]\n",local,file.Fd())
3054 safd := gshPA.Files[0]
3055 //gshPA.File[0] = fd;
3056 gshPA.File[0] = file;
3057 gshCtx.gshelly(argv[2])
3058 gshPA.Files[0] = safd
3059 file.Close();
3060 sconn.Close();
3061 aconn.Close();
3062 file.Close();
3063 } else{
3064 //port, err := net.ResolveUDPAddr("udp4",local);
3065 port, err := net.ResolveUDPAaddr("udp",local);
3066 if err != nil {
3067 fmt.Println("Address error: %s (%s)\n",local,err)
3068 return
3069 }
3070 fmt.Println("Listen UDP at %s...\n",local);
3071 //uconn, err := net.ListenUDP("udp4", port)
3072 uconn, err := net.ListenUDP("udp", port)
3073 if err != nil {
3074 fmt.Println("Listen error: %s (%s)\n",local,err)
3075 return
3076 }
3077 }
}

```

```

3078     file, _ := uconn.File()
3079     //fd := file.FD()
3080     ar := uconn.RemoteAddr()
3081     remote := ""
3082     if ar != nil { remote = ar.String() }
3083     if remote == "" { remote = "?" }
3084 
3085     // not yet received
3086     //fmt.Println("Accepted at "+s[4d] <- *s\n",local,fd,"")
3087 
3088     savfd := gshPA.Files[0]
3089     //gshPA.Files[0] = fd;
3090     gshPA.Files[0].file;
3091     savev := gshPA.Env
3092     gshEnv := append(gshPA.Env, "REMOTE_HOST"+remote)
3093     gshCtx.gshEnv = savev
3094     gshPA.Env = savev
3095     gshPA.Files[0] = savfd
3096 
3097     uconn.Close();
3098     file.Close();
3099   }
3100 }
3101 
3102 // empty line command
3103 func (gshCtx*GshContext)xPwd(argv []string){
3104   // execute context command, pwd = date
3105   // context notation representation scheme, to be resumed at re-login
3106   cwd, _ := os.Getwd()
3107   switch {
3108   case isnin("-a", argv):
3109     gshEnv.ShowChdirHistory(argv)
3110   case isnin("-la", argv):
3111     showFileInfo(cwd,argv)
3112   default:
3113     fmt.Printf("%s\n", cwd)
3114   case isnin("-c", argv): // obsolete empty command
3115     t := time.Now()
3116     date := t.Format(time.UnixDate)
3117     exe, _ := os.Executable()
3118     host, _ := os.Hostname()
3119     fmt.Printf("(PWD=%s,%s,%s", cwd,
3120     fmt.Printf("(PID=%s,%s,%s", os.Getpid(),
3121     fmt.Printf("( DATE=%s,%s", date)
3122     fmt.Printf("( TIME=%s", t.String())
3123     fmt.Printf(" PID=%s", os.Getpid())
3124     fmt.Printf(" EXE=%s", exe)
3125     fmt.Printf("\n")
3126   }
3127 }
3128 
3129 // <a name="history">History</a>
3130 // these should be browsed and edited by HTTP browser
3131 // show the time of command with -t and directory with -ls
3132 // open history by -o or by -m -c
3133 // sort by elapsed time by -t -c
3134 // search by "more" like interface
3135 // edit history
3136 // sort history, and wc or uniq
3137 // CPU and other resource consumptions
3138 // make log output (by time or so)
3139 // export / import history
3140 func (gshCtx *GshContext)xHistory(argv []string){
3141   atWorkDirX := -1
3142   if 1 < len(argv) && strBegins(argv[1],"-") {
3143     atWorkDirX_ = strconv.Atoi(argv[1][1:])
3144   }
3145   //fmt.Println("--D-- showHistory(iv)\n",argv)
3146   for i, v := range gshCtx.CommandHistory {
3147     // exclude commands not to be listed by default
3148     // internal commands may be suppressed by default
3149     if v.CmdLine == "" && !isin("-a",argv) {
3150       continue;
3151     }
3152     if 0 <= atWorkDirX {
3153       if v.WorkDirX != atWorkDirX {
3154         continue
3155       }
3156     }
3157     if !isin("-n",argv){ // like "fc"
3158       fmt.Println("i-2d ",i)
3159     }
3160     if isnin("-y",argv){
3161       fmt.Println(v) // should be with it date
3162     }else{
3163       if isnin("-l",argv) || isnin("-10",argv) {
3164         elps := v.EndAt.Sub(v.StartAt);
3165         start := v.StartAt.Format(time.Stamp)
3166         fmt.Printf("%d ",v.WorkDirX)
3167         fmt.Printf("[v] %1v/%t ",start,elps)
3168       }
3169       if isnin("-t",argv) && isnin("-10",argv){
3170         fmt.Println(v.Rusage,"%t %u// %s",argv,v.Rusage)
3171       }
3172       if isnin("-xt",argv) { // isnin("-ls",argv){
3173         dhi := v.WorkDirX // workdir history index
3174         fmt.Printf("%d %t",dhi,v.WorkDir)
3175         // show the FileInfo of the output command??
3176       }
3177       fmt.Printf("%s",v.CmdLine)
3178       fmt.Println("\n")
3179     }
3180   }
3181 }
3182 // in - history index
3183 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
3184   if gline[0] != '-' {
3185     hix, err := strconv.Atoi(gline[1:])
3186     if err != nil {
3187       fmt.Println("--E-- (%s : range)\n",hix)
3188       return "", false, true
3189     }
3190     if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3191       fmt.Println("--E-- (%d : out of range)\n",hix)
3192       return "", false, true
3193     }
3194     return gshCtx.CommandHistory[hix].CmdLine, false, false
3195   }
3196   // search
3197   //for i, v := range gshCtx.CommandHistory {
3198   //}
3199   return gline, false, false
3200 }
3201 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
3202   if 0 <= hix && hix < len(gsh.CommandHistory) {
3203     return gsh.CommandHistory[hix].CmdLine,true
3204   }
3205   return "",false
3206 }
3207 
3208 // temporary adding to PATH environment
3209 // cd name -lib for LD_LIBRARY_PATH
3210 // chdir with directory history (date + full-path)
3211 // -s for sort option (by visit date or so)
3212 func (gsh*GshContext)xShowChdirHistory(i int,v GChdirHistory, argv []string){
3213   if i <= 0 {
3214     fmt.Println("i-2d ",i,v.CmdIndex) // the first command at this WorkDir
3215     fmt.Println("[%v ]",v.MovedAt.Format(time.Stamp))
3216     showFileInfo(v.Dir,argv)
3217   }
3218   func (gsh*GshContext)xShowChdirHistory(argv []string){
3219     for i, v := range gsh.ChdirHistory {
3220       gsh.ShowChdirHistory(i,v,argv)
3221     }
3222   }
3223   func skipOps(argv[]string)(int{
3224     for i,v := range argv {
3225       if strBegins(v,"-") {
3226         jelse{
3227           return i
3228         }
3229       }
3230     }
3231     return -1
3232   }
3233   func (gshCtx*GshContext)xChdir(argv []string){
3234     chdir := gshCtx.ChdirHistory
3235     if isnin("-",argv) || isnin("-t",argv) || isnin("-a",argv) {
3236       gshCtx.ShowChdirHistory(argv)
3237       return
3238     }
3239     pfd, _ := os.Getwd()
3240     dir := ""
3241 
```

```

3240     if len(argv) <= 1 {
3241         dir = toFullPath("~")
3242     } else {
3243         i := skipopts(argv[1:])
3244         if i < 0 {
3245             dir = toFullPath("~")
3246         } else {
3247             dir = argv[1+i]
3248         }
3249     }
3250     if strBegins(dir, "-") {
3251         if dir == "-0" { // obsolete
3252             dir = gshCtx.StartDir
3253         } else {
3254             if dir == "-i" {
3255                 index := len(cdhist) - 1
3256                 if 0 < index { index -= 1 }
3257                 dir = cdhist[index].dir
3258             } else {
3259                 index, err := strconv.Atoi(dir[1:])
3260                 if err != nil {
3261                     fmt.Printf("--E-- xChdir(%v)\n", err)
3262                     dir = "?"
3263                 } else {
3264                     if len(gshCtx.ChdirHistory) <= index {
3265                         fmt.Printf("--E-- xChdir(history range error)\n")
3266                         dir = "?"
3267                     } else {
3268                         dir = cdhist[index].dir
3269                     }
3270                 }
3271             }
3272             if dir == "?" {
3273                 err := os.Chdir(dir)
3274                 if err != nil {
3275                     fmt.Printf("--E-- xChdir(%s)(%v)\n", argv[1], err)
3276                 } else {
3277                     cwd, _ := os.Getwd()
3278                     if cwd != pid {
3279                         hist1 := GchdirHistory {
3280                             hist1.dir = cwd
3281                             hist1.MovedAt = time.Now()
3282                             hist1.Command = argv[1:(gshCtx.CommandHistory)+1]
3283                             gshCtx.ChdirHistory = append(cdhist, hist1)
3284                             if !isIn("-s", argv) {
3285                                 // cwd, _ := os.Getwd()
3286                                 // fmt.Printf("%s\n", cwd)
3287                                 ix := len(gshCtx.ChdirHistory)-1
3288                                 gshCtx.ShowChdirHistory(ix, hist1, argv)
3289                             }
3290                         }
3291                     }
3292                 if isIn("-ls", argv) {
3293                     cwd, _ := os.Getwd()
3294                     showFileInfo(cwd, argv);
3295                 }
3296             }
3297         }
3298     func TimeValSub(tv1 time.Duration, tv2 time.Duration) {
3299         //tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
3300         tv1 -= tv2;
3301     }
3302     func RusageSub(ru1, ru2 [2]aRusage)([2]aRusage) {
3303         TimeValSub(&ru1[0].Utime, &ru2[0].Utime)
3304         TimeValSub(&ru1[0].Stime, &ru2[0].Stime)
3305         TimeValSub(&ru1[1].Utime, &ru2[1].Utime)
3306         TimeValSub(&ru1[1].Stime, &ru2[1].Stime)
3307         return ru1
3308     }
3309     func TimeValAdd(tv1 time.Duration, tv2 time.Duration) time.Duration {
3310         //tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
3311         tvs := tv1 + tv2;
3312         return tvs;
3313     }
3314     /*
3315     func RusageAdd(ru1, ru2 [2]aRusage)([2]aRusage) {
3316         TimeValAdd(&ru1[0].Utime, &ru2[0].Utime)
3317         TimeValAdd(&ru1[0].Stime, &ru2[0].Stime)
3318         TimeValAdd(&ru1[1].Utime, &ru2[1].Utime)
3319         TimeValAdd(&ru1[1].Stime, &ru2[1].Stime)
3320         return ru1
3321     }
3322     */
3323     // < name="rusage">Resource Usage</a>
3324     func Srusagef(fmtSpec string, argv []string, ru [2]aRusage)(string) {
3325         // ru[0] self, ru[1] children
3326         u := TimeValAdd(ru[0].Utime, ru[1].Utime)
3327         us := TimeValAdd(ru[0].Stime, ru[1].Stime)
3328         /u := int64(st.Usec*1000000000 + int64(rt.Usec) * 1000
3329         /u/u := (int64(st.Seconds)*1000000 + int64(st.Usec)) * 1000
3330         uu := ut; // in nano sec
3331         su := st; // in nano sec
3332         tu := uu + su
3333         ret := fmt.Sprintf("%v/sum", abbtime(tu))
3334         ret += fmt.Sprintf("%v/sec", abbtme(us))
3335         ret += fmt.Sprintf("%v/sys", abbtme(su))
3336         return ret
3337     }
3338     func Rusagef(fmtSpec string, argv []string, ru [2]aRusage)(string) {
3339         at := TimeValAdd(ru[0].Utime, ru[1].Utime)
3340         st := TimeValAdd(ru[0].Stime, ru[1].Stime)
3341         //fmt.Printf("%d.%06ds/u ",st.Seconds,ru[1].Utime)
3342         //fmt.Printf("%d.%06ds/u ",st.Seconds,ru[1].Utime)
3343         fmt.Printf("%d.%06ds/u ",ru[1].Utime.Seconds,ru[1].Utime.Usec)
3344         fmt.Println("%d.%06ds/u ",ru[1].Utime.Seconds,ru[1].Utime.Usec)
3345         fmt.Println("%d.%06ds/u ",st.Seconds,ru[1].Utime.Usec)
3346         return ""
3347     }
3348     func Getrusagev(([2]aRusage) {
3349         var ruv = [2]aRusage{}
3350         aGetrusage(&RUSAGE_SELF, &ruv[0])
3351         aGetrusage(&RUSAGE_CHILDREN, &ruv[1])
3352         return ruv
3353     }
3354     func (gshCtx *GshContext)xTime(argv []string)(bool) {
3355         if 2 < len(argv) {
3356             gshCtx.LastRusage = aRusage{}
3357             rusage := Getrusagev()
3358             fin := gshCtx.gshelv(argv[1:])
3359             rusagev2 := Getrusagev()
3360             showRusage(argv[1], argv, gshCtx.LastRusage)
3361             rusage := RusageSub(rusagev2, rusagev1)
3362             showRusage("self", argv, rusagev1)
3363             showRusage("child", argv, rusagev1)
3364             return false
3365         } else {
3366             rusage := aRusage {}
3367             aGetrusage(&RUSAGE_SELF, &rusage)
3368             aGetrusage("self", argv, &rusage)
3369             aGetrusage(&RUSAGE_CHILDREN, &rusage)
3370             showRusage("child", argv, &rusage)
3371             return false
3372         }
3373     }
3374     func (gshCtx *GshContext)xJobs(argv []string) {
3375         fmt.Println("# Jobs", len(gshCtx.BackGroundJobs))
3376         for ji, pid := range gshCtx.BackGroundJobs {
3377             //wstat := syscall.WaitStatus(0)
3378             rusage := aRusage {}
3379             //wpid, err := syscall.Wait4(pid, &wstat, syscall.WNOHANG, &rusage);
3380             //wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
3381             wpid := pid.Pid();
3382             err := errors.New("stab_NoError");
3383             if err != nil {
3384                 fmt.Printf("--E-- %%d [%d] (%v)\n", ji, wpid, err)
3385             } else {
3386                 fmt.Printf("%d[%d]\n", ji, wpid)
3387                 showRusage("child", argv, &rusage)
3388             }
3389         }
3390     }
3391     func (gsh*GshContext)inBackground(argv[]string)(bool) {
3392         if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n", argv) }
3393         gsh.Background = true // set background option
3394         xfin := false
3395         xfin = !gshelv(argv)
3396         gsh.Background = false
3397         return xfin
3398     }

```

```

3402 // -o file without command means just opening it and refer by #N
3403 // should be listed by "files" command
3404 func (gshCtx *GshContext)fromPipe(argv[]string){
3405     /*var i=1; i<len(argv)-1; i++*/
3406     //err := syscall.Pipe(pv)
3407     //fmt.Printf("--I-- pipe()=[#d,#d](%v)\n",pv[0],pv[1],err)
3408     pin,pout,err := os.Pipe()
3409     fmt.Printf("--I-- pipe()=[#d,#d](%v)\n",pin.Fd(),pout.Fd(),err)
3410 }
3411 func (gshCtx*GshContext)fromPipe(argv[]string){
3412 }
3413 func (gshCtx*GshContext)close(argv[]string){
3414 }
3415
3416 // <a name="redirect">redirect</a>
3417 func (gshCtx*GshContext)redirect(argv[]string)(bool{
3418     if len(argv) < 2 {
3419         return false
3420     }
3421
3422     cmd := argv[0]
3423     fname := argv[1]
3424     var file *os.File = nil
3425
3426     fdix := 0
3427     mode := os.O_RDONLY
3428
3429     switch {
3430     case cmd == "-i" || cmd == "<":
3431         fdix = 0
3432         mode = os.O_RDONLY
3433     case cmd == "-o" || cmd == ">":
3434         fdix = 1
3435         mode = os.O_RDWR | os.O_CREATE
3436     case cmd == "->" || cmd == ">>":
3437         fdix = 1
3438         mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3439     }
3440
3441     if fname[0] == '#' {
3442         fd, err := strconv.Atoi(fname[1:])
3443         if err != nil {
3444             fmt.Printf("--E-- (%v)\n",err)
3445             return false
3446         }
3447         file = os.NewFile(uintptr(fd),"MaybePipe")
3448     }else{
3449         xfile, err := os.Openfile(argv[1], mode, 0600)
3450         if err != nil {
3451             fmt.Printf("--E-- (%s)\n",err)
3452             return false
3453         }
3454         file = xfile
3455     }
3456     gshPA := gshCtx.gshPA
3457     gshPA.Files(fdix) = file.Fd()
3458     gshPA.Files(fdix) = file;
3459     fmt.Printf("--I-- Opened [#d] %s\n",file.Fd(),argv[1])
3460     gshCtx.gshell(argv[2:])
3461     gshPA.Files(fdix) = savfd
3462
3463     return false
3464 }
3465
3466 //fmt.Fprintf(res, "GShell Status: %s", html.EscapeString(req.URL.Path))
3467 func httpHandler(res http.ResponseWriter, req *http.Request){
3468     path := req.URL.Path
3469     fmt.Println("--I-- Got HTTP Request(%s)\n",path)
3470     {
3471         gshctxBuf, _ := setupGshContext()
3472         gshCtx := &gshctxBuf
3473         fmt.Println("--I-- %s\n",path[1:])
3474         gshCtx.tgshell(path[1:])
3475     }
3476     fmt.Fprintf(res, "Hello(~)//%s\n",path)
3477 }
3478 func (gshCtx *GshContext) httpServer(argv []string){
3479     http.HandleFunc("/", httpHandler)
3480     accport := "localhost:9999"
3481     fmt.Println("--I-- HTTP Server Start at [%s]\n",accport)
3482     http.ListenAndServe(accport,nil)
3483 }
3484 func (gshCtx *GshContext)xCo(argv[]string){
3485     go gshCtx.gshell(argv[1]);
3486 }
3487 func (gshCtx *GshContext) xPs(argv[]string)(){
3488 }
3489
3490 // <a name="plugin">plugin</a>
3491 // plugin [-i [name]] to list plugins
3492 // Reference: <a href="https://olano.org/src/plugin/">plugin</a> source code
3493 func (gshCtx *GshContext)whichPlugin(name string,argv[]string)(pi *PluginInfo){
3494     pi = nil
3495     for _,p := range gshctx.PluginFuncs {
3496         if p.Name == name && pi == nil {
3497             pi = &p
3498         }
3499         if isin("-s",argv){
3500             //fmt.Printf("%v %v ",i,p)
3501             if isin("-i",argv){
3502                 showFileInfo(p.Path,argv)
3503             }else{
3504                 fmt.Printf("%s\n",p.Name)
3505             }
3506         }
3507     }
3508     return pi
3509 }
3510 func (gshCtx *GshContext) xPlugin(argv[]string) (error {
3511     if len(argv) == 0 || argv[0] == "-is" {
3512         gshctx.whichPlugin("",argv)
3513         return nil
3514     }
3515     name := argv[0]
3516     pi := gshCtx.whichPlugin(name,[]string{"-s"})
3517     if pi == nil {
3518         os.Args = argv // should be recovered?
3519         pi.Addr.(func())()
3520         return nil
3521     }
3522     sofile := toFullPath(argv[0] + ".so") // or find it by which($PATH)
3523
3524     p, err := plugin.Open(sofile)
3525     if err != nil {
3526         fmt.Println("--E-- plugin.Open(%s)(%v)\n",sofile,err)
3527         return err
3528     }
3529     name := "Main"
3530     f, err := p.Lookup(fname)
3531     if f != nil {
3532         fmt.Println("--E-- plugin.Lookup(%s)(%v)\n",fname,err)
3533         return err
3534     }
3535     pi := PluginInfo{p,f,name,sofile}
3536     gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pi)
3537     fmt.Println("--I-- added (%d)\n",len(gshCtx.PluginFuncs))
3538
3539 //fmt.Println("--I-- first call(%s:%s)%v\n",sofile,fname,argv)
3540 os.Args = argv
3541 f.(func)()
3542 return err
3543 }
3544 func (gshCtx*GshContext)Args(argv[]string){
3545     for i,v := range os.Args {
3546         fmt.Printf("[%v] %v\n",i,v)
3547     }
3548 }
3549 func (gshCtx*GshContext) showVersion(argv[]string){
3550     if isin("-l",argv) {
3551         fmt.Printf("%v/%v (%v)",NAME,VERSION,DATE);
3552     }else{
3553         fmt.Printf("v%v",VERSION);
3554     }
3555     if isin("-a",argv) {
3556         fmt.Printf(" %s",AUTHOR)
3557     }
3558     if isin("-n",argv) {
3559         fmt.Printf("\n");
3560     }
3561 }
3562
3563 // <a name="scant">Scant</a> // string decomposer

```

```

3564 // scanf [format] [input]
3565 func scan(str string)(strv[]string{
3566     strv = strings.Split(str,",")
3567     return strv
3568 }
3569 func scanUntil(src,end string)(rstr string,leng int){
3570     idx := strings.Index(src,end)
3571     if 0 <= idx {
3572         rstr = src[0:idx]
3573         return rstr,idx+leng(end)
3574     }
3575     return src,0
3576 }
3577 // -Bn -- scan base-name part only // can be in some %fmt, for sed rewriting
3578 func (gsh*GshContext)Printf(fmts string,vstr string,optv[]string){
3579     //vint,err := strconv.Atoi(vstr)
3580     var ival int64 = 0
3581     n := 0
3582     err := error(nil)
3583     if strBegins(vstr,"-") {
3584         vx, _ := strconv.Atoi(vstr[1:])
3585         if vx < len(gsh.iValues) {
3586             vstr = gsh.iValues[vx]
3587         }else{
3588         }
3589     } // should use Eval()
3590     if strBegins(vstr,"0x") {
3591         n,err = fmt.Sscanf(vstr[2],"%x",&ival)
3592     }else{
3593         n,err = fmt.Sscanf(vstr,"%d",&ival)
3594     }
3595     //fmt.Printf("--D-- n=%d err=%(v) (%s)-v\n",n,err,vstr, ival)
3596     if n == 1 && err == nil {
3597         //fmt.Printf("--D-- format(%v) ival(%v)\n",fmts,ival)
3598         fmt.Printf("%"+fmts,ival)
3599     }else{
3600         if isin("-bn",optv){
3601             fmt.Printf("%"+fmts,filepath.Base(vstr))
3602         }else{
3603             fmt.Printf("%"+fmts,vstr)
3604         }
3605     }
3606 }
3607 func (gsh*GshContext)Printfv(fmts,div string,argv[]string,optv[]string,list[]string{
3608     //fmt.Printf("(#d)",len(list))
3609     //curfmt := "v"
3610     outlen := 0
3611     curfmt := gsh.iFormat
3612     if 0 < len(fmts) {
3613         for xi := 0; xi < len(fmts); xi++ {
3614             fch := fmts[xi]
3615             if fch == 'v' {
3616                 if xi+1 < len(fmts) {
3617                     curfmt = string(fmts[xi+1])
3618                 }
3619             }
3620             curfmt = curfmt + fch
3621             xi += 1
3622             if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3623                 vals,leng := eval(fmts[xi+2:],")")
3624                 //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
3625                 gsh.printVal(curfmt,vals,optv)
3626                 xi += 2+leng-1
3627                 outlen += 1
3628             }
3629             continue
3630         }
3631         if fch == '-' {
3632             hi,leng := scanInt(fmts[xi+1:])
3633             if 0 < leng {
3634                 if hi < len(gsh.iValues) {
3635                     gsh.printVal(curfmt,gsh.iValues[hi],optv)
3636                     outlen += 1 // should be the real length
3637                 }else{
3638                     fmt.Printf("((out-range))")
3639                 }
3640                 xi += leng
3641                 continue;
3642             }
3643             curfmt = curfmt + fch
3644         }
3645         fmt.Println(curfmt)
3646         outlen += 1
3647     }else{
3648         //fmt.Printf("--D-- print (%s)\n")
3649         for i,v := range list {
3650             if 0 < i {
3651                 fmt.Printf(div)
3652             }
3653             gsh.printVal(curfmt,v,optv)
3654             outlen += 1
3655         }
3656     }
3657     if 0 < outlen {
3658         fmt.Println("\n")
3659     }
3660 }
3661 func (gsh*GshContext)Scanv(argv[]string){
3662     //fmt.Printf("--D-- Scanv(%v)\n",argv)
3663     if len(argv) == 1 {
3664         return
3665     }
3666     argv = argv[1:]
3667     if strBegins(argv[0],"-F") {
3668         fmts = argv[0]
3669         gsh.iDelimiter = fmts
3670         argv = argv[1:]
3671     }
3672     input := strings.Join(argv, " ")
3673     if fmts == "" { // simple decomposition
3674         v := scanv(input)
3675         gsh.iValues = v
3676         //fmt.Println("v\n",strings.Join(v,""))
3677     }else{
3678         v := make([]string,0)
3679         n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3680         fmt.Printf("--D-- Scanf ->(%v) n=%d err=%(v)\n",v,n,err)
3681         gsh.iValues = v
3682     }
3683 }
3684 func (gsh*GshContext)Printv(argv[]string{
3685     if false { //%%U
3686         fmt.Printf("vv\n",strings.Join(argv[1:], " "))
3687     }
3688 }
3689     //fmt.Printf("--D-- Printv(%v)\n",argv)
3690     //fmt.Printf("--D-- Printv(%v)\n",strings.Join(gsh.iValues,""))
3691     div := gsh.iDelimiter
3692     fmts := ""
3693     argv = argv[1:]
3694     if 0 < len(argv) {
3695         if strBegins(argv[0],"-F") {
3696             div = argv[1:2]
3697             argv = argv[1:]
3698         }
3699     }
3700     optv := []string{
3701         for v := range argv {
3702             if strBegins(v,"-"){
3703                 optv = append(optv,v)
3704             }else{
3705                 break;
3706             }
3707         }
3708     }
3709     if 0 < len(argv) {
3710         fmts = strings.Join(argv, " ")
3711     }
3712     gsh.printfv(fmts,div,argv,optv,gsh.iValues)
3713 }
3714 func (gsh*GshContext)Basename(argv[]string){
3715     for i,v := range gsh.iValues {
3716         gsh.iValues[i] = filepath.Base(v)
3717     }
3718 }
3719 func (gsh*GshContext)Sortv(argv[]string){
3720     for i,v := range gsh.iValues {
3721         gsh.iValues[i] = sort.Slice(sv, func(i,j int) bool {
3722             sv := gsh.iValues
3723             sort.Slice(sv , func(i,j int) bool {
3724                 sv := gsh.iValues
3725                 sort.slice(sv , func(i,j int) bool {
3726                     sv := gsh.iValues
3727                 })
3728             })
3729         })
3730     }
3731 }
3732 }
3733 }
3734 }
3735 }
3736 }
3737 }
3738 }
3739 }
3740 }
3741 }
3742 }
3743 }
3744 }
3745 }
3746 }
3747 }
3748 }
3749 }
3750 }
3751 }
3752 }
3753 }
3754 }
3755 }
3756 }
3757 }
3758 }
3759 }
3760 }
3761 }
3762 }
3763 }
3764 }
3765 }
3766 }
3767 }
3768 }
3769 }
3770 }
3771 }
3772 }
3773 }
3774 }
3775 }
3776 }
3777 }
3778 }
3779 }
3780 }
3781 }
3782 }
3783 }
3784 }
3785 }
3786 }
3787 }
3788 }
3789 }
3790 }
3791 }
3792 }
3793 }
3794 }
3795 }
3796 }
3797 }
3798 }
3799 }
3800 }
3801 }
3802 }
3803 }
3804 }
3805 }
3806 }
3807 }
3808 }
3809 }
3810 }
3811 }
3812 }
3813 }
3814 }
3815 }
3816 }
3817 }
3818 }
3819 }
3820 }
3821 }
3822 }
3823 }
3824 }
3825 }
3826 }
3827 }
3828 }
3829 }
3830 }
3831 }
3832 }
3833 }
3834 }
3835 }
3836 }
3837 }
3838 }
3839 }
3840 }
3841 }
3842 }
3843 }
3844 }
3845 }
3846 }
3847 }
3848 }
3849 }
3850 }
3851 }
3852 }
3853 }
3854 }
3855 }
3856 }
3857 }
3858 }
3859 }
3860 }
3861 }
3862 }
3863 }
3864 }
3865 }
3866 }
3867 }
3868 }
3869 }
3870 }
3871 }
3872 }
3873 }
3874 }
3875 }
3876 }
3877 }
3878 }
3879 }
3880 }
3881 }
3882 }
3883 }
3884 }
3885 }
3886 }
3887 }
3888 }
3889 }
3890 }
3891 }
3892 }
3893 }
3894 }
3895 }
3896 }
3897 }
3898 }
3899 }
3900 }
3901 }
3902 }
3903 }
3904 }
3905 }
3906 }
3907 }
3908 }
3909 }
3910 }
3911 }
3912 }
3913 }
3914 }
3915 }
3916 }
3917 }
3918 }
3919 }
3920 }
3921 }
3922 }
3923 }
3924 }
3925 }
3926 }
3927 }
3928 }
3929 }
3930 }
3931 }
3932 }
3933 }
3934 }
3935 }
3936 }
3937 }
3938 }
3939 }
3940 }
3941 }
3942 }
3943 }
3944 }
3945 }
3946 }
3947 }
3948 }
3949 }
3950 }
3951 }
3952 }
3953 }
3954 }
3955 }
3956 }
3957 }
3958 }
3959 }
3960 }
3961 }
3962 }
3963 }
3964 }
3965 }
3966 }
3967 }
3968 }
3969 }
3970 }
3971 }
3972 }
3973 }
3974 }
3975 }
3976 }
3977 }
3978 }
3979 }
3980 }
3981 }
3982 }
3983 }
3984 }
3985 }
3986 }
3987 }
3988 }
3989 }
3990 }
3991 }
3992 }
3993 }
3994 }
3995 }
3996 }
3997 }
3998 }
3999 }
3999 }
```

```

3726     return sv[i] < sv[j]
3727   })
3728 }
func (gsh*GshContext)Shiftv(argv[]string){
3729   vi := len(gsh.iValues)
3730   if 0 < vi {
3731     if isn("-r",argv) {
3732       top := gsh.iValues[0]
3733       gsh.iValues = append(gsh.iValues[1:],top)
3734     }else{
3735       gsh.iValues = gsh.iValues[1:]
3736     }
3737   }
3738 }
3739 func (gsh*GshContext)Enq(argv[]string){
3740 }
3741 func (gsh*GshContext)Deq(argv[]string){
3742 }
3743 func (gsh*GshContext)Push(argv[]string{
3744   gsh.iValStack = append(gsh.iValStack,gsh.argv[1:])
3745   fmt.Printf("depth=%d\n",len(gsh.iValStack))
3746 }
3747 func (gsh*GshContext)Dump(argv[]string{
3748   for i,v := range gsh.iValStack {
3749     fmt.Printf("%d %v\n",i,v)
3750   }
3751 }
3752 func (gsh*GshContext)Pop(argv[]string{
3753   depth := len(gsh.iValStack)
3754   if 0 < depth {
3755     v := gsh.iValStack[depth-1]
3756     if isn("-cat",argv){
3757       gsh.iValues = append(gsh.iValues,v...)
3758     }else{
3759       gsh.iValues = v
3760     }
3761   }
3762   gsh.iValStack = gsh.iValStack[0:depth-1]
3763   fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3764 }else{
3765   fmt.Printf("depth=%d\n",depth)
3766 }
3767 }
3768 }
3769 // <a name="interpreter">Command Interpreter</a>
3770 func (gshctx*GshContext)gshellv(argv []string) (fin bool) {
3771   fin = false
3772 }
3773 if gshctx.Cmdtrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3774 if len(argv) <= 0 {
3775   return false
3776 }
3777 argv := []string{
3778   for ai := 0; ai < len(argv); ai++ {
3779     argv = append(argv,strsubst(gshctx,argv[ai],false))
3780   }
3781 }
3782 argv = argv
3783 if false {
3784   for ai := 0; ai < len(argv); ai++ {
3785     fmt.Printf("[%d] %s [%d]\n",
3786           ai,argv[ai],len(argv[ai]),argv[ai])
3787   }
3788 }
3789 cmd := argv[0]
3790 if gshctx.Cmdtrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)%v\n",len(argv),argv) }
3791 switch // https://tour.golang.org/flowcontrol/11
3792 case cmd == "x":
3793   gshctx.CmdTrace = ! gshctx.CmdTrace
3794 case cmd == "-xt":
3795   gshctx.CmdTime = ! gshctx.CmdTime
3796 case cmd == "-ot":
3797   gshctx.sconnect(true, argv)
3798 case cmd == "-ou":
3799   gshctx.sconnect(false, argv)
3800 case cmd == "-it":
3801   gshctx.Accept(true , argv)
3802 case cmd == "-ac":
3803   gshctx.Accept(true , argv)
3804 case cmd == "-as":
3805   gshctx.Accept(false, argv)
3806 case cmd == "-i" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3807   gshctx.Redirect(argv)
3808 case cmd == "-":
3809   gshctx.FilePipe(argv)
3810 case cmd == "arg":
3811   gshctx.Arg(argv)
3812 case cmd == "bg" || cmd == "-bg":
3813   rfin := gshctx.inBackground(argv[1:])
3814   return rfin
3815 case cmd == "-bn":
3816   gshctx.Basename(argv)
3817 case cmd == "call":
3818   _,_ = gshctx.Excommand(false,argv[1:])
3819 case cmd == "cd" || cmd == "chdir":
3820   gshctx.Xchdir(argv);
3821 case cmd == "cksum":
3822   gshctx.Xfind(argv)
3823 case cmd == "-sum":
3824   gshctx.Xfind(argv)
3825 case cmd == "-suntest":
3826   if 1 < len(argv) { str = argv[1] }
3827   crc := strCRC32(str,uint64(len(str)))
3828   fprintf(stderr,"%v %v\n",crc,len(str))
3829 case cmd == "close":
3830   gshctx.Xclose(argv)
3831 case cmd == "":
3832   gshctx.FileCm(argv)
3833 case cmd == "dec" || cmd == "decode":
3834   gshctx.Dec(argv)
3835 case cmd == "#define":
3836 case cmd == "dic" || cmd == "d":
3837   gshctx.Dic(argv)
3838 case cmd == "dump":
3839   gshctx.Dump(argv)
3840 case cmd == "echo" || cmd == "e":
3841   echo(argv,true)
3842 case cmd == "enc" || cmd == "encode":
3843   gshctx.Enc(argv)
3844 case cmd == "env":
3845   env(argv)
3846 case cmd == "eval":
3847   xEval(argv[1],true)
3848 case cmd == "events":
3849   case cmd == "events":
3850     dumpEvents(argv)
3851 case cmd == "exec":
3852   _,_ = gshctx.Excommand(true,argv[1:])
3853   // should not return here
3854 case cmd == "exit" || cmd == "quit":
3855   // write result code EXIT to 3>
3856   return true
3857 case cmd == "fdle":
3858   // dump the attributes of fds (of other process)
3859 case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3860   gshctx.Xfind(argv[1:])
3861 case cmd == "fu":
3862   gshctx.XFind(argv[1:])
3863 case cmd == "fork":
3864   // mainly for a server
3865 case cmd == "-gen":
3866   gshctx.Gen(argv)
3867 case cmd == "-go":
3868   gshctx.XGo(argv)
3869 case cmd == "-grep":
3870   gshctx.Xfind(argv)
3871 case cmd == "gded":
3872   gshctx.Gded(argv)
3873 case cmd == "genc":
3874   gshctx.Eng(argv)
3875 case cmd == "gpop":
3876   gshctx.Pop(argv)
3877 case cmd == "gput":
3878   gshctx.Put(argv)
3879 case cmd == "history" || cmd == "hi": // hi should be alias
3880   gshctx.XHistory(argv)
3881 case cmd == "jobs":
3882   gshctx.XJobs(argv)
3883 case cmd == "lisp" || cmd == "nlisp":
3884   gshctx.SLisp(argv)
3885 case cmd == "-ls":
3886   gshctx.XFind(argv)
3887 case cmd == "nop":
3888 }
```

```

3888     // do nothing
3889     case cmd == "pipe":
3890         gshCtx.xPipe(argv)
3891     case cmd == "plus" || cmd == "plugin" || cmd == "pin":
3892         gshCtx.xPlugin(argv[1:])
3893     case cmd == "print" || cmd == "-pr":
3894         // output internal slice // also sprintf should be
3895         gshCtx.Println(argv)
3896     case cmd == "xps":
3897         gshCtx.xPs(argv)
3898     case cmd == "pstitle":
3899         // to be gsh.title
3900     case cmd == "rexecd" || cmd == "rexd":
3901         gshCtx.RexecServer(argv)
3902     case cmd == "rex" || cmd == "rex":
3903         gshCtx.RexecClient(argv)
3904     case cmd == "repeat" || cmd == "rep": // repeat cond command
3905         gshCtx.repeat(argv)
3906     case cmd == "replay":
3907         gshCtx.xReplay(argv)
3908     case cmd == "scan":
3909         // scan input (or so in fscanf) to internal slice (like Files or map)
3910         gshCtx.Scanv(argv)
3911     case cmd == "set":
3912         // set name ...
3913     case cmd == "serv":
3914         gshCtx.hndlServer(argv)
3915     case cmd == "shift":
3916         gshCtx.Shiftv(argv)
3917     case cmd == "sleep":
3918         gshCtx.Sleep(argv)
3919     case cmd == "-sort":
3920         gshCtx.Sortv(argv)
3921
3922     case cmd == "j" || cmd == "join":
3923         gshCtx.Rjoin(argv)
3924     case cmd == "a" || cmd == "alpa":
3925         gshCtx.Ralpha(argv)
3926     case cmd == "jcd" || cmd == "jchdir":
3927         gshCtx.Rchdir(argv)
3928     case cmd == "jget":
3929         gshCtx.Rget(argv)
3930     case cmd == "jls":
3931         gshCtx.Rls(argv)
3932     case cmd == "jput":
3933         gshCtx.Rput(argv)
3934     case cmd == "jpwd":
3935         gshCtx.Rpwd(argv)
3936
3937     case cmd == "time":
3938         fin = gshCtx.xTime(argv)
3939     case cmd == "ungets":
3940         if 1 < len(argv) {
3941             ungets(argv[1]+"\n")
3942         } else {
3943
3944     case cmd == "pwd":
3945         gshCtx.xPwd(argv);
3946     case cmd == "ver" || cmd == "-ver" || cmd == "version":
3947         gshCtx.showVersion(argv)
3948     case cmd == "who":
3949         // data file or so?
3950     case cmd == "which":
3951         which("PATH", argv);
3952     case cmd == "g" && 1 < len(argv) && argv[1] == "listen":
3953         go gj.server(argv[1]);
3954     case cmd == "g" && 1 < len(argv) && argv[1] == "serve":
3955         go gj.server(argv[1]);
3956     case cmd == "g" && 1 < len(argv) && argv[1] == "join":
3957         go gj.client(argv[1]);
3958     case cmd == "g":
3959         join(argv);
3960     case cmd == "jsend":
3961         jsend(argv);
3962     default:
3963         if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
3964             gshCtx.xPlugin(argv)
3965         } else {
3966             notfound_ := gshCtx.excommand(false,argv)
3967             if notfound_ {
3968                 fmt.Printf("-E-- command not found (%v)\n",cmd)
3969             }
3970         }
3971     }
3972     return fin
3973 }
3974
3975 func (gsh*GshContext)gshell(gline string)(rfin bool) {
3976     argv := strings.Split(string(gline), " ")
3977     fin := gsh.gshell(argv)
3978     return fin
3979 }
3980 func (gsh*GshContext)tgshell(gline string)(xfin bool){
3981     start := time.Now()
3982     fin := gsh.gshell(gline)
3983     end := time.Now()
3984     elps := end.Sub(start);
3985     if gsh.CmdTime {
3986         fmt.Printf("-E-- %s + time.Now().Format(time.Stamp) + "(%d.%09ds)\n",
3987             elps/1000000000,elps%1000000000)
3988     }
3989     return fin
3990 }
3991 func Ttyid() (int {
3992     fi, err := os.Stdin.Stat()
3993     if err != nil {
3994         return 0;
3995     }
3996     //fmt.Printf("Stdin: %v Dev=%d\n",
3997     // fi.Mode(),fi.Mode()&os.ModeDevice)
3998     if (fi.Mode() & os.ModeDevice) != 0 {
3999         stat := &astat_t{};
4000         err = err.(*stat)
4001         if err != nil {
4002             //fmt.Printf("-I-- Stdin: (%v)\n",err)
4003         } else{
4004             //fmt.Printf("-I-- Stdin: rdev=%d %d\n",
4005             // stat.Rdev&0xFF,stat.Rdev);
4006             //fmt.Printf("-I-- Stdin: tty%d\n",stat.Rdev&0xFF);
4007             return int(stat.Rdev & 0xFF)
4008         }
4009     }
4010     return 0
4011 }
4012 func (gshCtx *GshContext) ttyfile() string {
4013 //fmt.Println("--I-- GSH_HOME=%s",gshCtx.GshHomeDir)
4014     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
4015         fmt.Sprintf("%02d",gshCtx.TerminalId)
4016     //strconv.Itoa(gshCtx.TerminalId)
4017 //fmt.Println("--I-- ttyfile=%s",ttyfile)
4018     return ttyfile
4019 }
4020 func (gshCtx *GshContext) ttyline()(*os.File{
4021     file, err := os.Openfile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4022     if err != nil {
4023         fmt.Printf("--F-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
4024         return file;
4025     }
4026     return file
4027 }
4028 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string {
4029     if skipping{
4030         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4031         line, _ := reader.ReadLine()
4032         return string(line)
4033     }else{
4034         if true {
4035             return xgetline(hix,prevline,gshCtx)
4036         }
4037     }
4038     if with_exgetline && gshCtx.GetLine != "" ){
4039         //var xhix int64=hix // cast
4040         newenv := os.Environ()
4041         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
4042
4043         tty := gshCtx.ttyline()
4044         tty.WriteString(prevline)
4045         Pa := os.ProcessEnv {
4046             -- // start dir
4047             newenv, //os.Environ(),
4048             []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
4049

```

```

4050         nil,
4051     }
4052 //fmt.Println("--I-- getline=&s // *\n",gsh_getlinev[0],gshCtx.GetLine)
4053 proc,err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&PA)
4054     if err != nil {
4055         fmt.Printf("--F-- getline process error (%v)\n",err)
4056         // for ; ; {
4057         return "exit (getline program failed)"
4058     }
4059 //stat, err := proc.Wait()
4060 proc.Wait()
4061 buff := make([]byte,LINESIZE)
4062 count,err := tty.Read(buff)
4063 //_,err = tty.Read(buff)
4064 //fmt.Println("--D-- getline (%d)\n",count)
4065 if err != nil {
4066     if !(count == 0) { // && err.String() == "EOF" ) {
4067         fmt.Printf("--E-- getline error (%s)\n",err)
4068     }
4069 }else{
4070     fmt.Printf("--I-- getline OK \\'s\\n",buff)
4071 }
4072 tty.Close()
4073 gline := string(buff[0:count])
4074 return gline
4075 }
4076 else
4077 {
4078     // if isatty {
4079     fmt.Printf("!\t%d",hix)
4080     fmt.Println(PROMPT)
4081     //}
4082     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4083     line,_ := reader.ReadLine()
4084     return string(line)
4085 }
4086 }
4087 /**
4088 ** begin ===== getline
4089 */
4090 * getline.c
4091 * 2020-0819 extracted from dog.c
4092 * getline.go
4093 * 2020-0822 ported to Go
4094 */
4095 /*
4096 package main // getline main
4097 import (
4098     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
4099     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
4100     "os" // <a href="https://golang.org/pkg/os/">os</a>
4101     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
4102     "/bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
4103     "/os/exec" // <a href="https://golang.org/pkg/os/">os</a>
4104 )
4105 */
4106
4107 // C language compatibility functions
4108 var errno = 0
4109 var stdin *os.File = os.Stdin
4110 var stdout *os.File = os.Stdout
4111 var stderr *os.File = os.Stderr
4112 var EOF = -1
4113 var NULL = 0
4114 type FILE os.File
4115 type Stat struct{[]byte}
4116 var NULL FP os.File = nil
4117 var NULLP = 0
4118 //var LINESIZE = 1024
4119
4120 func system(cmdstr string)(int){
4121     // /proc/self/cAttr {
4122     PA := os.ProcAttr {
4123         "", // the starting directory
4124         os.Environ(),
4125         //|uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
4126         []*os.File(os.Stdin,os.Stdout,os.Stderr),
4127         nil,
4128     }
4129     argv := strings.Split(cmdstr," ")
4130     //pid,err := syscall.ForkExec(argv[0],argv,&PA)
4131     proc,err := os.StartProcess(argv[0],argv,&PA);
4132     if (err != nil) {
4133         //fmt.Println("--E-- system(%v)\n(%v)",cmdstr,err);
4134         return -1;
4135     }
4136     pstat_r := proc.Wait();
4137     Pid := pstat.Pid();
4138     if (err != nil) {
4139         fmt.Println("--E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
4140     }
4141     //syscall.Wait4(pid,nil,0,nil)
4142     //fmt.Println("==== pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
4143
4144     /*
4145     argv := strings.Split(cmdstr," ")
4146     fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
4147     //cmd := exec.Command(argv[0];...)
4148     cmd := exec.Command(argv[0],argv[1],argv[2])
4149     cmd.Stdin = strings.NewReader("output of system")
4150     var serr bytes.Buffer
4151     cmd.Stdout = &serr
4152     var serr bytes.Buffer
4153     cmd.Stderr = &serr
4154     err := cmd.Run()
4155     if (err != nil) {
4156         fmt.Fprintf(os.Stderr,"--E-- system(%v)err(%v)\n",argv,err)
4157         fmt.Println("ERR:%s",serr.String())
4158     }else{
4159         fmt.Println("%s",out.String())
4160     }
4161     */
4162     return 0
4163 }
4164 func atoi(str string)(ret int){
4165     ret,err := fmt.Sscanf(str,"%d",ret)
4166     if err == nil {
4167         return ret
4168     }else{
4169         // should set errno
4170         return 0
4171     }
4172 }
4173 func getenv(name string)(string){
4174     val,got := os.LookupEnv(name)
4175     if got {
4176         return val
4177     }else{
4178         return "?"
4179     }
4180 }
4181 func strcpy(dst StrBuff, src string){
4182     var i int
4183     srcb := []byte(src)
4184     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4185         dst[i] = srcb[i]
4186     }
4187     dst[i] = 0
4188 }
4189 func xstrcpy(dst StrBuff, src StrBuff){
4190     dst = src
4191 }
4192 func strcat(dst StrBuff, src StrBuff){
4193     dst = append(dst,src...)
4194 }
4195 func strdup(str StrBuff)(string){
4196     return string(str[:strlen(str)])
4197 }
4198 func strlen(str string)(int){
4199     return len(str)
4200 }
4201 func strlen(str StrBuff)(int){
4202     var i int
4203     for i = 0; i < len(str) && str[i] != 0; i++ {
4204         }
4205     return i
4206 }
4207 func sizeof(data StrBuff)(int){
4208     return len(data)
4209 }
4210 func isatty(fd int)(ret int){
4211     return 1

```

```

4212 }
4213 func fopen(file string, mode string)(fp*os.File){
4214     if mode == "r" {
4215         fp,err := os.Open(file)
4216         if( err != nil ){
4217             fmt.Printf("-- fopen(%s,%s)=(%v)\n",file,mode,err)
4218             return NULL_FP;
4219         }
4220         return fp;
4221     }else{
4222         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4223         if( err != nil ){
4224             return NULL_FP;
4225         }
4226         return fp;
4227     }
4228 }
4229 func fclose(fp*os.File){
4230     fp.Close()
4231 }
4232 func fflush(fp *os.File)(int){
4233     return 0
4234 }
4235 func fgetc(fp*os.File)(int){
4236     var buf [1]byte
4237     err := fp.Read(buf[0:1])
4238     if( err != nil ){
4239         return EOF;
4240     }else{
4241         return int(buf[0])
4242     }
4243 }
4244 func fgets(str*string, size int, fp*os.File)(int){
4245     buf := make(StrBuff, size)
4246     buf := StrBuff{buf, size}
4247     var ch int
4248     var i int
4249     for i = 0; i < len(buf)-1; i++ {
4250         ch = fgetc(fp)
4251         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4252         if( ch == EOF ) {
4253             break;
4254         }
4255         buf[i] = byte(ch);
4256         if( ch == '\n' ){
4257             break;
4258         }
4259     }
4260     buf[i] = 0
4261     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4262     return i
4263 }
4264 func fgets(buf StrBuff, size int, fp*os.File)(int){
4265     var ch int
4266     var i int
4267     for i = 0; i < len(buf)-1; i++ {
4268         ch = fgetc(fp)
4269         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4270         if( ch == EOF ) {
4271             break;
4272         }
4273         buf[i] = byte(ch);
4274         if( ch == '\n' ){
4275             break;
4276         }
4277     }
4278     buf[i] = 0
4279     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4280     return i
4281 }
4282 func fputs(ch int , fp*os.File)(int){
4283     var buf [1]byte
4284     buf[0] = byte(ch)
4285     fp.WriteString(buf[0:i])
4286     return 0
4287 }
4288 func fputs(buf StrBuff, fp*os.File)(int){
4289     fp.Write(buf)
4290     return 0
4291 }
4292 func xputts(str string, fp*os.File)(int){
4293     return fputs(([]byte(str)),fp)
4294 }
4295 func sscanfStr StrBuff,fmts string, params ...interface()(int){
4296     fmt.Sscanf(string(str[0:len(str)]),fmts,params...)
4297     return 0
4298 }
4299 func fprintf(fp*os.File,fmts string, params ...interface()(int){
4300     fmt.Fprintf(fp,fmts,params...)
4301     return 0
4302 }
4303 }
4304 // <a name="IME">>Command Line IME</a>
4305 //----- MyIME
4306 var MyIMEMap = map[string]MyIME{0.0.2};
4307 type RomKana struct {
4308     dic string // dictionary ID
4309     pat string // input pattern
4310     cut string // output pattern
4311     hit int64 // count of hit and used
4312 }
4313 var dicents = 0
4314 var RomKana [1024]RomKana
4315 var Romkanan []RomKana
4316
4317 func isinDic(str string)(int{
4318     for i,v := range RomKana {
4319         if v.pat == str {
4320             return i
4321     }
4322 }
4323 return -1
4324 }
4325 const {
4326     DIC_COM_LOAD = "im"
4327     DIC_COM_DUMP = "s"
4328     DIC_COM_LIST = "ls"
4329     DIC_COM_ENR = "en"
4330     DIC_COM_DIS = "di"
4331 }
4332 func helpdic(argv []string){
4333     cmd := stder
4334     cmd.Stderr = cmd
4335     if 0 < len(argv) { cmd = argv[0] }
4336     fprintf(out,..., `Usage(%n",cmd)
4337     fprintf(out,..., `Commands(%n")
4338     fprintf(out,..., `  % 3v [dicName] [dicURL] -- Import dictionary(%n",cmd,DIC_COM_LOAD)
4339     fprintf(out,..., `  % 3v [pattern] -- Search in dictionary(%n",cmd,DIC_COM_DUMP)
4340     fprintf(out,..., `  % 3v [dicName] -- List dictionaries(%n",cmd,DIC_COM_LIST)
4341     fprintf(out,..., `  % 3v [dicName] -- Enable dictionary(%n",cmd,DIC_COM_ENR)
4342     fprintf(out,..., `  % 3v [dicName] -- Disable dictionary(%n",cmd,DIC_COM_DIS)
4343     fprintf(out,..., `  Keys ... %v\n",`%s can be used for '\\\'")
4344     fprintf(out,..., `  \\'c -- Reverse the case of the last character(%n",
4345     fprintf(out,..., `  \\'i -- Replace input with translated text(%n",
4346     fprintf(out,..., `  \\'l -- Force Lower Case(%n"),
4347     fprintf(out,..., `  \\'u -- Force Upper Case (software CapsLock)\n",
4348     fprintf(out,..., `  \\'v -- Show translation actions(%n",
4349     fprintf(out,..., `  \\'x -- Replace the last input character with it Hexa-Decimal(%n",
4350     }
4351 }
4352 func xdic(argv[]string){
4353     if len(argv) <= 1 {
4354         helpdic(argv)
4355         return
4356     }
4357     argv = argv[1]
4358     var debug = false
4359     var info = false
4360     var silent = false
4361     var dump = false
4362     var builtin = false
4363     cmd := argv[0]
4364     argv = argv[1:]
4365     Out := os.Stdout
4366     arg := ""
4367
4368     if 0 < len(argv) {
4369         arg := argv[0]
4370         if argv[0] == '-' {
4371             switch arg {
4372                 default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4373             return
4374         }
4375     }
4376 }

```

```

4374         case "-b": builtin = true
4375         case "-d": debug = true
4376         case "-s": silent = true
4377         case "-v": info = true
4378     }
4379     opt = argv[1]
4380     argv = argv[2:]
4381 }
4382
4383 dicName := ""
4384 dicURL := ""
4385 if 0 < len(argv) {
4386     arg0 := argv[0]
4387     dicName = arg0
4388     argv = argv[1:]
4389 }
4390 if 0 < len(argv) {
4391     dicURL = argv[0]
4392     argv = argv[1:]
4393 }
4394 if false {
4395     fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4396 }
4397 if cmd == DIC_COM_LOAD {
4398     //dicType := "load"
4399     dicBody := ""
4400     if !builtin && dicName != "" && dicURL == "" {
4401         f,err := os.Open(dicName)
4402         if err != nil {
4403             dicURL = dicName
4404         } else {
4405             f,err = os.Open(dicName+".html")
4406             if err == nil {
4407                 dicURL = dicName+".html"
4408             } else {
4409                 f,err = os.Open("gshdic-"+dicName+".html")
4410                 if err == nil {
4411                     dicURL = "gshdic-"+dicName+".html"
4412                 }
4413             }
4414         }
4415     }
4416     if err == nil {
4417         var buf := make([]byte,128*1024)
4418         count,err := f.Read(buf)
4419         f.Close()
4420         if info {
4421             fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
4422         }
4423         dicBody = string(buf[0:count])
4424     }
4425 }
4426 if dicBody == "" {
4427     switch arg {
4428     default:
4429         dicName = "WorldDic"
4430         dicURL = WorldDic
4431         if info {
4432             fprintf(stderr,"--Id-- default dictionary \"%v\"\n",
4433                 dicName);
4434         }
4435         case "wnn":
4436             dicName = "WnnDic"
4437             dicURL = WnnDic
4438         case "sURL":
4439             dicName = "SumomoDic"
4440             dicURL = sumomoDic
4441         case "sijimi":
4442             dicName = "SijimiDic"
4443             dicURL = SijimiDic
4444         case "jkl":
4445             dicName = "JKLJaDic"
4446             dicURL = JA_JKLDic
4447     }
4448     if debug {
4449         fprintf(stderr,"--Id-- %v URL=%v\n",dicName,dicURL);
4450     }
4451     dicv := strings.Split(dicURL,",")
4452     if debug {
4453         fprintf(stderr,"--Id-- %v encoded data...\n",dicName)
4454         fprintf(stderr,"%v\n",dicv[0])
4455         fprintf(stderr,"%v\n",dicv[1])
4456         fprintf(stderr,"%v\n")
4457     }
4458     body,_ := base64.StdEncoding.DecodeString(dicv[1])
4459     dicBody = string(body)
4460 }
4461 if info {
4462     fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
4463     fmt.Printf("%s\n",dicBody)
4464 }
4465 if debug {
4466     fprintf(stderr,"--Id-- dicName %v text...\n",dicName)
4467     fprintf(stderr,"%v\n",string(dicBody))
4468 }
4469 envt := strings.Split(dicBody,"\n");
4470 if info {
4471     fprintf(stderr,"--Id-- %v scan...\n",dicName);
4472 }
4473 var added int = 0
4474 var dup int = 0
4475 for i,v := range envt {
4476     var pat string
4477     var out string
4478     fmt.Sscanf(v,"%s %s",&pat,&out)
4479     if len(pat) <= 0 {
4480         if else{
4481             if 0 <= isinDic(pat) {
4482                 dup += 1
4483                 continue
4484             }
4485             romkanan[dicents] = RomKana(dicName,pat,out,0)
4486             dicents += 1
4487             added += 1
4488             Ronkan = append(Ronkan,RomKana(dicName,pat,out,0))
4489             if debug {
4490                 fmt.Printf("[%v]:[%2v]-%v [%2v]\n",
4491                     i,len(pat),pat,len(out),out)
4492             }
4493         }
4494     }
4495     if isilent {
4496         url := dicURL
4497         if strbegins(url,"data:") {
4498             url = "builtin"
4499         }
4500         fprintf(stderr,"--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4501             dicName,added,dup,len(Ronkan),url);
4502     }
4503     // should sort by pattern length for conclete match, for performance
4504     if debug {
4505         arg = "" // search pattern
4506         dump = true
4507     }
4508 }
4509 if cmd == DIC_COM_DUMP || dump {
4510     fprintf(stderr,"--Id-- %v dump... %v entries:\n",dicName,len(Ronkan));
4511     var match = 0
4512     for i := 0; i < len(Ronkan); i++ {
4513         dic := Ronkan[i].dic
4514         pat := Ronkan[i].pat
4515         out := Ronkan[i].out
4516         if arg == "" || 0 <= strings.Index(pat,arg)|| 0 <= strings.Index(out,arg) {
4517             fmt.Printf("\n\\%v[%2v]-%v [%2v]\n",
4518                 i,dic,len(pat),pat,len(out),out)
4519             match += 1
4520         }
4521     }
4522     fprintf(stderr,"--Id-- %v matched %v / %v entries:\n",arg,match,len(Ronkan));
4523 }
4524 }
4525 func loadDefaultDic(dic int){
4526     if 0 < len(Ronkan) {
4527         return
4528     }
4529     //fprintf(stderr,"r\n")
4530     xDic([]string{"dic",DIC_COM_LOAD});
4531
4532     var info = false
4533     if info {
4534         fprintf(stderr,"--Id-- Congururations!! WorldDic is now activated.\r\n")
4535         fprintf(stderr,"--Id-- enter \"dic\" command for help.\r\n")
4536

```

```

4536 }
4537 }
4538 func readDic()(int){
4539 /*
4540 var rk *os.File;
4541 var dic = "MyIME-dic.txt";
4542 /*rk = fopen("romkana.txt","r");
4543 /*rk = fopen("JK-JA-morse-dic.txt","r");
4544 rk = fopen("r","r");
4545 if( rk == NULL_FI ){
4546   if( true ){
4547     fprintf(stderr,"--%s-- Could not load %s\n",MyIMEVER,dic);
4548   }
4549   return -1;
4550 }
4551 if( true ){
4552   var di int;
4553   var line = make(StrBuff,1024);
4554   var pat string;
4555   var out string;
4556   for di = 0; di < 1024; di++ {
4557     if( fgets(line,sizeof(line),rk) == NULLSP ){
4558       break;
4559     }
4560     fmt.Sscanf(string(line[0:strlen(line)]),"s %s",&pat,&out);
4561     //sscanf(line[0:strlen(line)],"%\r\n",&pat,&out);
4562     romkana[di].pat = out;
4563     romkana[di].out = out;
4564     //fprintf(stderr,"--%d-- loaded romkana.txt [%d]\n",MyIMEVER,di);
4565   }
4566   dicens += di;
4567   if( false ){
4568     fprintf(stderr,"--%s-- loaded romkana.txt [%d]\n",MyIMEVER,di);
4569     for di = 0; di < dicens; di++ {
4570       fprintf(stderr,
4571         "%s %s\n",romkana[di].pat,romkana[di].out);
4572     }
4573   }
4574 }
4575 fclose(rk);
4576 //romkana[dicens].pat = "//ddump"
4577 //romkana[dicens].pat = "//ddump" // dump the dic. and clean the command input
4578 */
4579 return 0;
4580 }
4581 func matchlen(stri string, pati string)(int){
4582   if strBegins(stri,pati) {
4583     return len(pati)
4584   }else{
4585     return 0
4586   }
4587 }
4588 }
4589 func convs(src string)(string){
4590 /*
4591 var sx = len(src);
4592 var di int;
4593 var mi int;
4594 var dstb []byte;
4595
4596 for si = 0; si < sx; { // search max. match from the position
4597   if strBegins(src[si:],"/x/") {
4598     // %x/integer// s/a/b/
4599     ix := strings.Index(src[si+3:],"/")
4600     if 0 < ix {
4601       var iv int = 0
4602       //fmt.Sscanf(src[si+3:si+3+ix],"#d",&iv)
4603       fmt.Sscanf(src[si+3:si+3+ix],"#v",&iv)
4604       eval := fmt.Sprintf("%x",iv)
4605       bval := []byte(eval)
4606       dstb = append(dstb,bval...)
4607       si = si+3+ix+1
4608       continue
4609     }
4610     if strBegins(src[si:],"/d/") {
4611       // #d/integer// s/a/b/
4612       ix := strings.Index(src[si+3:],"/")
4613       if 0 < ix {
4614         var iv int = 0
4615         //fmt.Sscanf(src[si+3:si+3+ix],"#v",&iv)
4616         eval := fmt.Sprintf("#d",iv)
4617         bval := []byte(eval)
4618         dstb = append(dstb,bval...)
4619         si = si+3+ix+1
4620         continue
4621       }
4622     }
4623     if strBegins(src[si:],"/t") {
4624       now := time.Now()
4625       if true {
4626         date := now.Format(time.Stamp)
4627         dstb = append(dstb,[byte(date)])
4628         si = si+3
4629       }
4630     }
4631     continue
4632   }
4633   var maxlen int = 0;
4634   var len int;
4635   mi = -1;
4636   for di = 0; di < dicens; di++ {
4637     len = matchlen(src[si:],romkana[di].pat);
4638     if( maxlen < len){
4639       maxlen = len;
4640       mi = di;
4641     }
4642   }
4643   if( 0 < maxlen ) {
4644     out := romkana[mi].out;
4645     dstb = append(dstb,[byte(out)]);
4646     si += maxlen;
4647   }else{
4648     dstb = append(dstb,src[si])
4649     si += 1;
4650   }
4651 }
4652 return string(dstb)
4653 }
4654 func trans(src string)(int){
4655   dst := convs(src);
4656   xputs(dst,stderr);
4657   return 0;
4658 }
4659 */
4660 //----- LINEEDIT
4661 // "7" at the top of the line means searching history
4662
4663 // should be compatible with Telnet
4664 const {
4665   EV_MODE      = 255
4666   EV_IDLE      = 254
4667   EV_TIMEOUT   = 253
4668
4669   GO_UP        = 252 // k
4670   GO_DOWN      = 253 // j
4671   GO_RIGHT     = 250 // l
4672   GO_LEFT      = 249 // h
4673   DEL_RIGHT    = 248 // x
4674   GO_TOPL     = 'A'-0x40 // 0
4675   GO_ENDL     = 'E'-0x40 // $ 
4676
4677   GO_TOWM     = 239 // b
4678   GO_ENDW     = 238 // e
4679   GO_NEXTW    = 237 // w
4680
4681   GO_FORWCH   = 229 // f
4682   GO_PAIRCH   = 228 // t
4683
4684   GO_DEL       = 219 // d
4685
4686   HI_SRCH_FW  = 209 // /
4687   HI_SRCH_BK  = 208 // ?
4688   HI_SRCH_RFW = 207 // n
4689   HI_SRCH_RBK = 206 // R
4690 }
4691
4692 // should return number of octets ready to be read immediately
4693 //fprintf(stderr,"In--Select(%v %v)\n",err,r.Bits[0])
4694
4695 var EventRecvFd = -1 // file descriptor
4696 var EventSendFd = -1

```

```

4698 const EventFdOffset = 1000000
4699 const NormalFdOffset = 100
4700
4701 /* 2020-10-21 replaced poll() with channel/select
4702 func putKeyInEvent(event int, evarg int){
4703     if true {
4704         if EventRecvFd < 0 {
4705             var sv [1]int = [-1,-1]
4706             syscall.Write(fd,sv)
4707             EventRecvFd = pv[0]
4708             EventSendFd = pv[1]
4709             //fmt.Printf("--> EventPipe created[%v,%v]\n",EventRecvFd,EventSendFd)
4710         }
4711     }else{
4712         if EventRecvFd < 0 {
4713             // the document differs from this spec
4714             // https://golang.org/src/syscall/syscall_unix.go?g=8096:8158#L340
4715             sv,err := syscall.Socketpair(syscall.AF_UNIX,syscall.SOCK_STREAM,0)
4716             EventRecvFd = sv[0]
4717             EventSendFd = sv[1]
4718             if err != nil {
4719                 fmt.Printf("--> EventSock created[%v,%v](%v)\n",EventSendFd,event,n,err)
4720             }
4721         }
4722     }
4723     var buf = [1]byte{byte(event)}
4724     n,err := syscall.Write(EventSendFd,buf)
4725     if err != nil {
4726         fmt.Printf("--> putEvent[%v](%v)(%v)\n",EventSendFd,event,n,err)
4727     }
4728 }
4729 */
4730 func ungets(str string){
4731     for _,ch := range str {
4732         putKeyInEvent(int(ch),0)
4733     }
4734 }
4735
4736 func (gsh*GshContext)xReplay(argv[]string){
4737     hix := 0
4738     tempo := 1.0
4739     xtempo := 1.0
4740     repeat := 1
4741     for _,a := range argv { // tempo
4742         if strBegins(a,"x") {
4743             fmt.Sscanf(a[1:],"%f",&xtempo)
4744             tempo = 1 / xtempo
4745             //fprint(stderr,"%-Dr-- tempo=[%v]@%v\n",a[2:],tempo);
4746         }else{
4747             if strBegins(a,"r") {
4748                 fmt.Sscanf(a[1:],"%v",&repeat)
4749             }else
4750                 if strBegins(a,"!") {
4751                     fmt.Sscanf(a[1:],"%d",&hix)
4752                 }else{
4753                     fmt.Sscanf(a,"%d",&hix)
4754                 }
4755         }
4756     }
4757     if hix == 0 || len(argv) <= 1 {
4758         hix = len(gsh.CommandHistory)-1
4759     }
4760     fmt.Printf("--Ir-- Replay(%v %v %v)\n",hix,xtempo,repeat)
4761     //dumpEvents(hix)
4762     //gsh.xScanReplay(hix,false,repeat,tempo,argv)
4763     gsh.XScanReplay(hix,true,repeat,tempo,argv)
4764
4765     runtime.Gosched(); // wait xScanReplay is launched
4766     //fmt.Printf("--Ir-- Replay set\n");
4767 }
4768
4769 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4770 // 2020-08-27 GShell-0.7.3
4771 /*
4772 func Fpollini(fp *os.File,usec int)(uintptr){
4773     fd := 1
4774
4775     rdv := syscall.FdSet {}
4776     fd1 := fp.Fd()
4777     bank1 := fd1/32
4778     mask1 := int32(1 << fd1)
4779     rdv.Bits[bank1] = mask1
4780
4781     fd2 := -1
4782     bank2 := -1
4783     var mask2 int32 = 0
4784
4785     if 0 <= EventRecvFd {
4786         fd2 = EventRecvFd
4787         fd := fd2
4788         bank2 = fd2/32
4789         mask2 = int32(1 << fd2)
4790         rdv.Bits[bank2] |= mask2
4791         //fmt.Printf("--> EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
4792     }
4793
4794     tout := syscall.NsecToTimeval(int64(usec*1000))
4795     />n,err := syscall.Select(fd,&rdv,nil,nil,&tout) // spec. mismatch
4796     err := syscall.Select(fd,&rdv,nil,nil,&tout)
4797     if err != nil {
4798         //fmt.Printf("--> select() err(%v)\n",err)
4799     }
4800     if err == nil {
4801         if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4802             if false {
4803                 fmt.Printf("--> got Event\n")
4804             }
4805             return uintptr(EventFdOffset + fd2)
4806         }else
4807             if (rdv.Bits[bank1] & mask1) != 0 {
4808                 return uintptr(NormalFdOffset + fd1)
4809             }else
4810                 return 1
4811         }else
4812         return 0
4813     }
4814 }
4815 */
4816
4817 /*
4818 func fgetctimeout(fp *os.File,usec int)(int{
4819     READ1:
4820     />readyFd := Fpollini(fp,usec)
4821     readyFd := CPollini(fp,usec)
4822     if readyFd < 100
4823         return EV_TIMEOUT
4824     }
4825
4826     var buf [1]byte
4827
4828     if EventFdOffset <= readyFd {
4829         fd := int(readyFd-EventFdOffset)
4830         _,err := syscall.Read(fd,buf[0:1])
4831         if (err != nil) {
4832             return EOF;
4833         }else{
4834             if buf[0] == EV_MODE {
4835                 recvKeyEvent(fd)
4836                 goto READ1
4837             }
4838             return int(buf[0])
4839         }
4840     }
4841
4842     if err := fp.Read(buf[0:1])
4843     if err != nil {
4844         return EOF;
4845     }else{
4846         return int(buf[0])
4847     }
4848 */
4849 */
4850
4851 func visibleChar(ch int)(string){
4852     switch {
4853         case '!': <= ch && ch <= '-':
4854             return string(ch)
4855     }
4856     switch ch {
4857         case ' ': return "\n"
4858         case '\n': return "\r"
4859         case '\r': return "\r"
4860     }
4861 }

```

```

4860     case '\t': return "\\\t"
4861 }
4862 switch ch {
4863     case 0x00: return "NUL"
4864     case 0x07: return "BEL"
4865     case 0x08: return "BS"
4866     case 0x0E: return "SO"
4867     case 0x0F: return "SI"
4868     case 0x1B: return "ESC"
4869     case 0x7F: return "DEL"
4870 }
4871 switch ch {
4872     case EV_IDLE: return fmt.Sprintf("IDLE")
4873     case EV_MODE: return fmt.Sprintf("MODE")
4874 }
4875 return fmt.Sprintf("%x",ch)
4876 */
4877 func recvKeyEvent(fd int){
4878     var buf [1]byte
4879     if _, err := syscall.Read(fd,buf[0:1]); err != nil {
4880         log.Panic(err)
4881     }
4882     if buf[0] != 0 {
4883         romkammode = true
4884     } else{
4885         romkammode = false
4886     }
4887 }
4888 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
4889     var Start time.Time
4890     var events = []Event{}
4891     for e := range Events {
4892         if hix == 0 || e.CmdIndex == hix {
4893             events = append(events,e)
4894         }
4895     }
4896     elen := len(events)
4897     if 0 < elen {
4898         if events[elen-1].event == EV_IDLE {
4899             events = events[0:elen-1]
4900         }
4901     }
4902     for r := 0; r < repeat; r++ {
4903         for i,e := range events {
4904             nano := e.when.Nanosecond()
4905             micro := nano / 1000
4906             if Start.Second() == 0 {
4907                 Start = time.Now()
4908             }
4909             diff := time.Now().Sub(Start)
4910             if replay {
4911                 if e.event != EV_IDLE {
4912                     //fmt.Println("--replay tv / tv event=%X\n",i,len(events),e.event);
4913                     putKeyinEvent(e.event,0)
4914                     if e.event == ev_MODE { // event with arg
4915                         putKeyinEvent(int(e.evarg),0)
4916                     }
4917                 } else{
4918                     //fmt.Println("--replay tv / tv idle=%X\n",i,len(events),e.event);
4919                 }
4920             } else{
4921                 fmt.Printf("%.7.3fms #%-3v %v [%v.%06d] %3v %02X %-4v %10.3fms\n",
4922                     float64(diff)/1000000.0,
4923                     i,
4924                     e.CmdIndex,
4925                     e.when.Format(time.Stamp),micro,
4926                     e.event,e.event,visibleChar(e.event),
4927                     float64(e.evarg)/1000000.0)
4928             }
4929             if e.event == EV_IDLE {
4930                 //fmt.Println("--replay tv / v delay\n",i,len(events));
4931                 d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
4932                 //nsleep(time.Duration(e.evarg))
4933                 nsleep(d)
4934             }
4935         }
4936     }
4937 }
4938 func dumpEvents(argv[]string){
4939     hix := 0
4940     if 1 < len(argv) {
4941         fmt.Sscanf(argv[1],"%d",&hix)
4942     }
4943     for i,e := range Events {
4944         nano := e.when.Nanosecond()
4945         micro := nano / 1000
4946         //if e.event != EV_TIMEOUT {
4947         if hix == 0 || e.CmdIndex == hix {
4948             fmt.Printf("%3v %v [%v.%06d] %3v %02X %-4v %10.3fms\n",i,
4949                     e.CmdIndex,
4950                     e.when.Format(time.Stamp),micro,
4951                     e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
4952         }
4953     }
4954 }
4955 */
4956 func fgetcTimeout(fp *os.File,usec int)(int){
4957     ch := fgetcTimeout(fp,usec)
4958     if ch != EV_TIMEOUT {
4959         now := time.Now()
4960         if 0 < len(Events) {
4961             last := Events[len(Events)-1]
4962             dura := int64(now.Sub(last.when))
4963             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
4964         }
4965         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4966     }
4967     return ch
4968 }
4969 */
4970 /*
4971 // 2020-10-21 replaced poll() with channel/select
4972 var Kbd = make(chan int)
4973 var KbInit = false;
4974 var evQ = make(chan int);
4975 /*
4976 */
4977 func keyInput(kbd chan int, fp *os.File){
4978     for {
4979         ch := C.gettc(C.stdin);
4980         if( ch == C.EOF){
4981             break;
4982         }
4983         kbd <- int(ch);
4984     }
4985 }
4986 */
4987 // https://godoc.org/golang.org/x/crypto/ssh/terminal
4988 // https://github.com/golang/go/issues/14094190/function-similar-to-getchar
4989 func keyInput(kbd chan int, tty *os.File){
4990     tmode := C.setTermMode(tmode);
4991     defer func(){ C.setTermMode(tmode); }();
4992     if !OnWindows(){
4993         system("/bin/stty -echo -icanon");
4994         defer func(){ system("/bin/stty sane"); }();
4995     }
4996     for {
4997         var rbuf []byte = make([]byte,1);
4998         if( OnWindows ){
4999             C.setTermRaw();
5000         }
5001         rerr := tty.Read(rbuf);
5002         if( rerr != nil){
5003             break;
5004         }
5005         //fmt.Printf("++KBD[%X]\n",rbuf[0]);
5006         kbd <- int(rbuf[0]);
5007     }
5008     if !OnWindows { system("/bin/stty echo sane"); }
5009 }
5010 func fgetcTimeout(fp *os.File,usec int)(int){
5011     if( !KbInit ){
5012         KbInit = true;
5013         go keyInput(Kbd,fp);
5014     }
5015     for {
5016         select {
5017             case <- time.After(time.Duration(usec*1000)):
5018                 //fmt.Println("--timeout %v us\n",usec);
5019                 return EV_TIMEOUT;
5020             case ch := <- Kbd:
5021                 //fmt.Println("--KBD[%X]\n",ch);
5022         }
5023     }
5024 }

```

```

5022 // record a keyin(ch) Event
5023 {
5024     now := time.Now()
5025     if 0 < len(Events) {
5026         last := Events[len(Events)-1]
5027         dura := int64(now.Sub(last.when))
5028         Events = append(Events,Event{last.when,EN_IDLE,dura,last.CmdIndex})
5029     }
5030     Events = append(Events,Event(time.Now(),ch,0,CmdIndex))
5031 }
5032 return ch;
5033 case ch == evQ:
5034     if( ch == EV_MODE){
5035         recvKeyEvent()
5036     }else{
5037         return ch;
5038     }
5039 }
5040 }
5041 func putKeyEvent(event int, evarg int) {
5042     evQ <- event;
5043 }
5044 func recvKeyEvent(){
5045     ch := <- evQ;
5046     if( ch != 0 ){
5047         romkannmode = true
5048     }else{
5049         romkannmode = false
5050     }
5051 }
5052 }
5053 var AtConsoleLineTop = true
5054 var TtyMaxCol = 72 // to be obtained by ioctl?
5055 var EscTimeout = (100*1000)
5056 var (
5057     MODE_ViMode bool // vi compatible command mode
5058     MODE_ShowMode bool // viw compatible command mode
5059     romkannmode bool // shown translation mode, the mode to be retained
5060     MODE_CapsLock bool // software CapsLock
5061     MODE_LockLock bool // force lower-case character lock
5062     MODE_VisPrint int // visible insert mode, should be like "I" icon in X Window
5063     MODE_ViTrace bool // output newline before translation
5064 )
5065 type IInnput struct {
5066     lno int
5067     lastline int
5068     pch []int // input queue
5069     ptemp string
5070     line string
5071     right string
5072     inEditMode bool
5073     pinMode bool
5074     waitingMeta string // waiting meta character
5075     LastCmd string
5076 }
5077
5078 func (iin*IInput)Getc(timeoutUs int)(int){
5079     ch1 := EOF
5080     ch2 := EOF
5081     ch3 := EOF
5082     if( 0 < len(iin.pch) ) { // deq
5083         iin.pch = iin.pch[1:]
5084     }else{
5085         ch1 = fgetcTimeout(stdin,timeoutUs);
5086     }
5087     if( ch1 == 033 ){ // escape sequence
5088         ch2 = fgetcTimeout(stdin,EscTimeout);
5089         if( ch2 == EV_TIMEOUT ){
5090             ch3 = fgetcTimeout(stdin,EscTimeout);
5091             if( ch3 == EV_TIMEOUT ){
5092                 lin.pch = append(iin.pch,ch2) // enQ
5093             }else{
5094                 switch( ch2 ){
5095                     default:
5096                         iin.pch = append(iin.pch,ch3) // enQ
5097                     case '[':
5098                         switch( ch3 ){
5099                             case 'A': ch1 = GO_UP; // ^
5100                             case 'B': ch1 = GO_DOWN; // v
5101                             case 'C': ch1 = GO_RIGHT; // >
5102                             case 'D': ch1 = GO_LEFT; // <
5103                             case '3':
5104                                 ch4 := fgetcTimeout(stdin,EscTimeout);
5105                                 if( ch4 == '-' ){
5106                                     //ffprintf(stderr,"%#02X %#02X %#02X\n",ch1,ch2,ch3,ch4);
5107                                     ch1 = DEL_RIGHT
5108                                 }
5109                             case '\'':
5110                                 ch4 := fgetcTimeout(stdin,EscTimeout);
5111                                 //ffprintf(stderr,"%#02X %#02X %#02X\n",ch1,ch2,ch3,ch4);
5112                                 ch1 = DEL_RIGHT
5113                         }
5114                     case 'H':
5115                         ch4 := fgetcTimeout(stdin,EscTimeout);
5116                         //ffprintf(stderr,"%#02X %#02X %#02X\n",ch1,ch2,ch3,ch4);
5117                         switch( ch3 ){
5118                             case '^': ch1 = DEL_RIGHT
5119                         }
5120                     }
5121                 }
5122             }
5123         }
5124     }
5125     return ch1
5126 }
5127 func (iin*IInput)clearline(){
5128     var i int
5129     fprintf(stderr,"%r");
5130     // should be ANSI ESC sequence
5131     for 1 <= i < (TtyMaxCol); i++ { // to the max. position in this input action
5132         fputc(' ',os.Stderr);
5133     }
5134     fprintf(stderr,"%r");
5135 }
5136 func (iin*IInput)Redraw(){
5137     redraw(iin.lno,iin.line,iin.right)
5138 }
5139 func redraw(iin *Input,lno int,line string,right string){
5140     inMeta := false
5141     showMode := "" // visible Meta mode on the cursor position
5142     showLine := fmt.Sprintf("%d! ",lno)
5143     insertMark := "" // in invisible insert mode
5144
5145     if MODE_ViMode {
5146     }else{
5147         if 0 < len(iin.right) {
5148             InsertMark = ""
5149         }
5150
5151         if( 0 < len(iin.waitingMeta) ){
5152             inMeta = true
5153             if in.waitingMeta[0] != 033 {
5154                 showMeta = iin.waitingMeta
5155             }
5156         }
5157         if( romkannmode ){
5158             //romkannmark = "*";
5159         }else{
5160             //romkannmark = "";
5161         }
5162         if MODE_ShowMode {
5163             romkan := "--";
5164             inveri := "-";
5165             inveri := "";
5166             if MODE_CapsLock {
5167                 inmeta = "A";
5168             }
5169             if MODE_LowerLock {
5170                 inmeta = "a";
5171             }
5172             if MODE_ViTrace {
5173                 inveri = "v";
5174             }
5175             if MODE_ViMode {
5176                 inveri = "-";
5177             }
5178             if romkannmode {
5179                 romkan = "%3d\\201\\202";
5180                 if MODE_CapsLock {
5181                     inmeta = "R";
5182                 }else{
5183                     inmeta = "r";
5184                 }
5185             }
5186         }
5187     }
5188 }

```

```

5184     }
5185     if inMeta {
5186         inmeta = "\\"+
5187     }
5188     showMode = "["+romkan+inmeta+inveri+"]";
5189 }
5190 Pre := "\r" + showMode + showLino
5191 Output += Pre
5192 Left := ""
5193 Right := ""
5194 if romkanmode {
5195     Left = convs(line)
5196     Right = InsertMark+convs(right)
5197     Yellow := true
5198     Left = line
5199     Right = InsertMark+right
5200 }
5201 Output = Pre+left
5202 if MODE_ViTrace {
5203     Output += in.LastCmd
5204 }
5205 Output += showMetaRight
5206 for len(Output) < TtyMaxCol { // to the max. position that may be dirty
5207     Output += " "
5208     // should be ANSI ESC sequence
5209     // not necessary just after newline
5210 }
5211 Output += PreLeftshowMeta // to set the cursor to the current input position
5212 fprinf(stderr,"$s",Output)
5213 if MODE_ViTrace {
5214     if 0 < len(in.LastCmd) {
5215         in.LastCmd = ""
5216         fprinf(stderr,"\r\n")
5217     }
5218     AtconsoleLinePop = false
5219 }
5220 //fmt.Printf("Redraw(%v)\n",len(line),len(right));
5221 } // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5222 func delHeadChar(str string)(line string,head string){
5223     clen := len(str)
5224     head = string(str[:clen])
5225     return str[clen:],head
5226 }
5227 func delta1Char(str string)(rline string, last string){
5228     var i = 0
5229     var clen = 0
5230     for {
5231         ,siz := utf8.DecodeRune([]byte(str)[i:])
5232         if siz <= 0 { break }
5233         clen = siz
5234         i += siz
5235     }
5236     last = str[len(str)-clen:]
5237     return str[0:len(str)-clen],last
5238 }
5239 // 3* for output and history
5240 // 4* for keylog?
5241 // <a name="getline">Command Line Editor</a>
5242 func xgetline(iin int, prevline string, gsh*GshContext)(string){
5243     var iin Input
5244     iin.lastline = iin.line
5245     iin.lno = iin.line.lno
5246     CmdIndex = len(gsh.CommandHistory)
5247     if (isatty(0) == 0) {
5248         if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
5249             iin.line = "exit\n";
5250         }else{
5251         }
5252     }
5253     return iin.line
5254 }
5255 if( true ){
5256     //var pts string;
5257     //pts = ptsname(0);
5258     //if pts != ttynname(0);
5259     //fprinf(stderr,"-pts[0] = %s\n",pts?pts:"?");
5260 }
5261 if( false ){
5262     fprinf(stderr,"! ");
5263     fprinf(stderr,LINESIZE,stdin);
5264     sfgets(&iin.line,LINESIZE,stdin);
5265     return iin.line
5266 }
5267 if( !OnWindows ) { system("/bin/stty -echo -icanon"); }
5268 xline := iin.xgetline(prevline,gsh)
5269 if( !OnWindows ) {system("/bin/stty echo sane"); }
5270 return xline
5271 }
5272 func (iin*Input)Translate(cmdch int){
5273     romkanmode = !romkanmode;
5274     if MODE_ViTrace {
5275         fprinf(stderr,"%v\r\n",string(cmdch));
5276     }else{
5277         if cmdch == 'J' ){
5278             fprinf(stderr,"J\r\n");
5279             iin.inJMode = true
5280         }
5281         iin.Redraw();
5282         loadDefaultDic(cmdch);
5283         iin.Redraw();
5284     }
5285 }
5286 func (iin*Input)Replace(cmdch int){
5287     iin.LastCmd = fmt.Sprintf("\%v",string(cmdch))
5288     iin.Redraw();
5289     loadDefaultDic(cmdch);
5290     dist := convs(iin.line+iin.right);
5291     iin.line = dist
5292     iin.right = ""
5293     if( iin.line[iin.line-1] == 'I' ){
5294         fprinf(stderr,"I\r\n");
5295         iin.inJMode = true
5296     }
5297     iin.Redraw();
5298 }
5299 }
5300 // as 12 ala1
5301 func isAlpha(ch rune)(bool){
5302     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5303         return true
5304     }
5305     return false
5306 }
5307 }
5308 }
5309 // 0.2.8 2020-09-01 created
5310 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5311 func (iin*Input)GetTOFW(){
5312     str := iin.line
5313     i := len(str)
5314     if i <= 0 {
5315         return
5316     }
5317     /i0 := i
5318     i -= 1
5319     lastSize := 0
5320     var lastRune rune
5321     var found = -1
5322     for 0 < i { // skip preamble spaces
5323         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5324         if !isAlnum(lastRune) { // character, type, or string to be searched
5325             i -= lastSize
5326             continue
5327         }
5328         break
5329     }
5330     for 0 < i {
5331         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5332         if lastsize <= 0 { continue } // not the character top
5333         if !isAlnum(lastRune) { // character, type, or string to be searched
5334             found = i
5335             break
5336         }
5337     }
5338 }
5339 }
5340 
```

```

5346     }
5347     i -= lastSize
5348   } if found < 0 && i == 0 {
5349     found = 0
5350   }
5351 }
5352 if 0 <= found {
5353   if isalnum(lastRune) { // or non-kana character
5354     } else{ // when positioning to the top o the word
5355       i += lastSize
5356     }
5357     iin.right = str[i:] + iin.right
5358   if 0 < i {
5359     iin.line = str[0:i]
5360   } else{
5361     iin.line = ""
5362   }
5363 }
5364 //fmt.Printf("\n%d,%d,%d)[%s][%s]\n",i,found,iin.line,iin.right)
5365 //fmt.Printf("") // set debug messae at the end of line
5366 }
5367 // 0.2.8 2020-0901 created
5368 func (iin*Input)GotoENDW(){
5369   str := iin.right
5370   if len(str) <= 0 {
5371     return
5372   }
5373   lastSize := 0
5374   var lastRune rune
5375   var lastW = 0
5376   i := 0
5377   inWord := false
5378
5379   lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
5380   if isalnum(lastRune) {
5381     r,z := utf8.DecodeRuneInString(str[lastSize:])
5382     if 0 < z && isalnum(r) {
5383       inWord = true
5384     }
5385   }
5386   for i < len(str) {
5387     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5388     if lastSize <= 0 { break } // broken data?
5389     if !isalnum(lastRune) { // character, type, or string to be searched
5390       break
5391     }
5392     lastW = i // the last alnum if in alnum word
5393     i += lastSize
5394   }
5395   if inWord {
5396     goto DISP
5397   }
5398   for i < len(str) {
5399     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5400     if lastSize <= 0 { break } // broken data?
5401     if !isalnum(lastRune) { // character, type, or string to be searched
5402       break
5403     }
5404     i += lastSize
5405   }
5406   for i < len(str) {
5407     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5408     if lastSize <= 0 { break } // broken data?
5409     if !isalnum(lastRune) { // character, type, or string to be searched
5410       break
5411     }
5412     lastW = i
5413     i += lastSize
5414   }
5415 DISP:
5416   if 0 < lastW {
5417     iin.line = iin.line + str[0:lastW]
5418     iin.right = str[lastW:]
5419   }
5420   //fmt.Printf("\n%d,%d)[%s][%s]\n",i,in.line,iin.right)
5421   //fmt.Printf("") // set debug messae at the end of line
5422 }
5423 // 0.2.8 2020-0901 created
5424 func (iin*Input)GotoNEXTW(){
5425   str := iin.right
5426   if len(str) <= 0 {
5427     return
5428   }
5429   lastSize := 0
5430   var lastRune rune
5431   var found = -1
5432   i := 1
5433   for i < len(str) {
5434     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5435     if lastSize <= 0 { break } // broken data?
5436     if !isalnum(lastRune) { // character, type, or string to be searched
5437       found = i
5438     }
5439   }
5440   i += lastSize
5441
5442   if 0 < found {
5443     if isalnum(lastRune) { // or non-kana character
5444       } else{ // when positioning to the top o the word
5445         found += lastSize
5446       }
5447     iin.line = iin.line + str[0:found]
5448   if 0 < found {
5449     iin.right = str[found:]
5450   } else{
5451     iin.right = ""
5452   }
5453 }
5454 //fmt.Printf("\n%d,%d)[%s][%s]\n",i,in.line,iin.right)
5455 //fmt.Printf("") // set debug messae at the end of line
5456 }
5457 // 0.2.8 2020-0902 created
5458 func (iin*Input)GotoPAIRCH(){
5459   str := iin.right
5460   if len(str) <= 0 {
5461     return
5462   }
5463   lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5464   if lastSize <= 0 {
5465     return
5466   }
5467   forw := false
5468   back := false
5469   pair := ""
5470   switch string(lastRune){
5471     case "{}": pair = "{}"; forw = true
5472     case ")": pair = "("; back = true
5473     case "(" : pair = ")"; forw = true
5474     case ")": pair = "("; back = true
5475     case "]": pair = "["; forw = true
5476     case "]": pair = "["; back = true
5477     case "<": pair = ">"; forw = true
5478     case ">": pair = "<"; back = true
5479     case "\"": pair = "\""; // context depednet, can be f' or back-double quote
5480     case "'": pair = "'"; // context depednet, can be f' or back-quote
5481     // case Japanese Kakkos
5482   }
5483   if forw {
5484     iin.SearchForward(pair)
5485   }
5486   if back {
5487     iin.SearchBackward(pair)
5488   }
5489 }
5490 // 0.2.8 2020-0902 created
5491 func (iin*Input)SearchForward(pat string)(bool){
5492   right := iin.right
5493   found := -1
5494   i := 0
5495   if strBegins(right,pat) {
5496     _,z := utf8.DecodeRuneInString(right[i:])
5497     if 0 < z {
5498       i += z
5499     }
5500   }
5501   for i < len(right) {
5502     if strBegins(right[i:],pat) {
5503       found = i
5504       break
5505     }
5506     _,z := utf8.DecodeRuneInString(right[i:])
5507     if z <= 0 { break }
5508   }
5509 }
```

```

5508     i += z
5509   }
5510   if 0 <= found {
5511     iin.line = iin.line + right[0:found]
5512     iin.right = iin.right[found:]
5513     return true
5514   }else{
5515     return false
5516   }
5517 }
5518 // 0.2.8 2020-0902 created
5519 func (lin*Input)searchBackward(pat string)(bool){
5520   line := iin.line
5521   found := -1
5522   i := len(line)-1
5523   for i := i; 0 <= i; i-- {
5524     _,z := util.DecodeRuneInString(line[i:])
5525     if z == 0 {
5526       continue
5527     }
5528     //printf(stderr,"-- %v\n",pat,line[i:])
5529     if strBegins(line[i:],pat) {
5530       found = i
5531       break
5532     }
5533   }
5534   //printf(stderr,"--%d\n",found)
5535   if 0 <= found {
5536     iin.line = line[0:found] + iin.right
5537     iin.right = line[found:]
5538     return true
5539   }else{
5540     return false
5541   }
5542 }
5543 // 0.2.8 2020-0902 created
5544 // search from top, end, or current position
5545 func (gsh*GshContext)searchHistory(pat string, forw bool)(bool,string){
5546   if forw {
5547     for _v := range gsh.CommandHistory {
5548       if 0 <= strings.Index(_v.CmdLine,pat) {
5549         //printf(stderr,"<--- found %s [%v]\n",i,pat,v.CmdLine)
5550         return true,_v.CmdLine
5551     }
5552   }else{
5553     hlen := len(gsh.CommandHistory)
5554     for i := hlen-1; 0 <= i; i-- {
5555       v := gsh.CommandHistory[i]
5556       if 0 <= strings.Index(v.CmdLine,pat) {
5557         //printf(stderr,">--- found %s [%v]\n",i,pat,v.CmdLine)
5558         return true,_v.CmdLine
5559     }
5560   }
5561 }
5562 //printf(stderr,"n--De-- not-found(%v)\n",pat)
5563 return false,"(Not Found in History)"
5564 // 0.2.8 2020-0902 created
5565 func (lin*Input)gotFORWSTR(pat string,gsh*GshContext){
5566   found := false
5567   if 0 < len(iin.right) {
5568     found = iin.SearchForward(pat)
5569   }
5570   if !found {
5571     found,line := gsh.SearchHistory(pat,true)
5572     if found {
5573       iin.line = line
5574       iin.right = ""
5575     }
5576   }
5577 }
5578 func (lin*Input)gotBACKSTR(pat string,gsh*GshContext){
5579   found := false
5580   if 0 < len(iin.line) {
5581     found = iin.SearchBackward(pat)
5582   }
5583   if !found {
5584     found,line := gsh.SearchHistory(pat,false)
5585     if found {
5586       iin.line = line
5587       iin.right = ""
5588     }
5589   }
5590 }
5591 }
5592 func (lin*Input)getStringI(prompt string)(string){ // should be editable
5593   iin.clearline();
5594   printf(stderr,"%v",prompt)
5595   str := ""
5596   for {
5597     ch := iin.Getc(10*1000*1000)
5598     if ch == '\n' || ch == '\r' {
5599       break
5600     }
5601     sch := string(ch)
5602     str += sch
5603     printf(stderr,"%s",sch)
5604   }
5605   return str
5606 }
5607 }
5608 // search pattern must be an array and selectable with '^N^P'
5609 var SearchHist = ""
5610 var SearchForw = true
5611 613 func (lin*Input)getlineI(prevline string,gsh*GshContext)(string){
5612   var ch int;
5613   MODE_ShowMode = false
5614   MODE_VicMode = false
5615   iin.Redraw();
5616   first := true
5617   for cix := 0; ; cix++ {
5618     iin.pinMode = iin.inMode
5619     iin.inMode = false
5620
5621     ch := iin.Getc(1000*1000)
5622
5623     if ch == EV_TIMEOUT && first {
5624       first = false
5625       mode := 0
5626       if romkanemode {
5627         mode = 1
5628       }
5629       now := time.Now()
5630       Events = append(Events,Event(now,EV_MODE,int64(mode),CmdIndex))
5631     }
5632     if ch == 033 {
5633       MODE_ShowMode = true
5634       MODE_VicMode = !MODE_VicMode
5635       iin.Redraw();
5636       continue
5637     }
5638   if MODE_VicMode {
5639     switch ch {
5640     case '0': ch = GO_TOPL
5641     case '0': ch = GO_ENDL
5642     case 'b': ch = GO_TOPW
5643     case 'e': ch = GO_ENDW
5644     case 'w': ch = GO_NEXTW
5645     case '%': ch = GO_PAIRCH
5646
5647     case 'j': ch = GO_DOWN
5648     case 'k': ch = GO_UP
5649     case 'h': ch = GO_LEFT
5650     case 'l': ch = GO_RIGHT
5651     case 'x': ch = DEL_RIGHT
5652     case 'a': MODE_VicMode = !NODE_VicMode
5653     ch = GO_RIGHT
5654     case 'i': MODE_VicMode = !MODE_VicMode
5655     iin.Redraw();
5656     continue
5657     case 'z':
5658       if iin.head == delHeadChar(iin.right)
5659       if len([]byte(head)) == 1 {
5660         ch = int(head[0])
5661         if 'a' <= ch && ch <= 'z' {
5662           ch = ch + 'A'-'a'
5663         }else{
5664           if 'A' <= ch && ch <= 'Z' {
5665             ch = ch + 'a'-'A'
5666           }
5667         }
5668       }
5669     }
5670   }
5671 }
```

```

5670         }
5671         iin.right = string(ch) + right
5672     } iin.Redraw();
5673     continue
5674 case 'f': // GO_FOWCH
5675     iin.Redraw();
5676     ch.iin.Getc(3*1000*1000)
5677     if ch == EV_TIMEOUT {
5678         iin.Redraw();
5679     continue
5680     }
5681     SearchPat = string(ch)
5682     SearchFor = true
5683     iin.GotoFORWSTR(SearchPat,gsh)
5684     iin.Redraw();
5685     continue
5686 case '/':
5687     SearchPat = iin.getstring1("//") // should be editable
5688     SearchFor = false
5689     iin.GotoBACKSTR(SearchPat,gsh)
5690     iin.Redraw();
5691     continue
5692 case '?':
5693     SearchPat = iin.getstring1("?) // should be editable
5694     SearchFor = false
5695     iin.GotoFORWSTR(SearchPat,gsh)
5696     iin.Redraw();
5697     continue
5698 case 'n':
5699     if SearchFor {
5700         iin.GotoFORWSTR(SearchPat,gsh)
5701     } else{
5702         iin.GotoBACKSTR(SearchPat,gsh)
5703     }
5704     iin.Redraw();
5705     continue
5706 case 'k':
5707     if iSearchFor {
5708         iin.GotoFORWSTR(SearchPat,gsh)
5709     } else{
5710         iin.GotoBACKSTR(SearchPat,gsh)
5711     }
5712     iin.Redraw();
5713     continue
5714 }
5715 }
5716 switch ch {
5717     case GO_UP:
5718         iin.GotoUPW()
5719         iin.Redraw();
5720     continue
5721     case GO_ENDW:
5722         iin.GotoENDW()
5723         iin.Redraw();
5724     continue
5725     case GO_NEXTW:
5726         // to next space then
5727         iin.GotoNEXTW()
5728         iin.Redraw();
5729     continue
5730     case GO_PARCH:
5731         iin.GotoPAIRCH()
5732         iin.Redraw();
5733     continue
5734 }
5735
5736 //fprintf(stderr,"A[%02X]\n",ch);
5737 if( ch == '\\' || ch == 033 ){
5738     MODE_ShowMode = true
5739     metach := ch
5740     iin.waitingMeta = string(ch)
5741     iin.Redraw();
5742     // set cursor //fprintf(stderr,"???\b\b\b")
5743     ch = fgetctimeout(stdin,2000*1000)
5744     // reset cursor
5745     iin.waitingMeta = ""
5746
5747     cmdch := ch
5748     if( ch == EV_TIMEOUT ){
5749         if metach == 033 {
5750             continue
5751         }
5752         ch = metach
5753     } else
5754     /*
5755     if( ch == 'm' || ch == 'M' ){
5756         mch := fgetctimeout(stdin,1000*1000)
5757         if mch == 'r' {
5758             romkanmode = true
5759         } else{
5760             romkanmode = false
5761         }
5762     } continue
5763     */
5764     if( ch == 'k' || ch == 'K' ){
5765         MODE_Recursive = !MODE_Recursive
5766         iin.Translate(cmdch);
5767     continue
5768     } else
5769     if( ch == 'j' || ch == 'J' ){
5770         iin.Translate(cmdch);
5771     continue
5772     } else
5773     if( ch == 'i' || ch == 'I' ){
5774         iin.Replace(cmdch);
5775     continue
5776     } else
5777     if( ch == 'l' || ch == 'L' ){
5778         MODE_LowerLock = !MODE_LowerLock
5779         MODE_CapsLock = false
5780         if MODE_VITrace {
5781             fprintf(stderr,"%v\r\n",string(cmdch));
5782         }
5783         iin.Redraw();
5784     continue
5785     } else
5786     if( ch == 'u' || ch == 'U' ){
5787         MODE_CapsLock = !MODE_CapsLock
5788         MODE_LowerLock = false
5789         if MODE_VITrace {
5790             fprintf(stderr,"%v\r\n",string(cmdch));
5791         }
5792         iin.Redraw();
5793     continue
5794     } else
5795     if( ch == 'v' || ch == 'V' ){
5796         MODE_VITrace = !MODE_VITrace
5797         if MODE_VITrace {
5798             fprintf(stderr,"%v\r\n",string(cmdch));
5799         }
5800         iin.Redraw();
5801     continue
5802     } else
5803     if( ch == 'c' || ch == 'C' ){
5804         if 0 < len(iin.line) {
5805             xline,tail := deltailChar(iin.line)
5806             if len([byte(tail)]) == 1 {
5807                 ch = int(tail[0])
5808                 if('a' <= ch && ch <= 'z' ){
5809                     ch = ch + 'A'-'a'
5810                 } else{
5811                     ch = ch + 'a'-'A'
5812                 }
5813                 if('A' <= ch && ch <= 'Z' ){
5814                     ch = ch + 'a'-'A'
5815                 }
5816             }
5817             iin.line = xline + string(ch)
5818         }
5819         if MODE_VITrace {
5820             fprintf(stderr,"%v\r\n",string(cmdch));
5821         }
5822         iin.Redraw();
5823     continue
5824     } else{
5825         iin.pch = append(iin.pch,ch) // push
5826         ch = '\\'
5827     }
5828 }
5829 switch( ch ){
5830     case 'P'-0x40: ch = GO_UP
5831     case 'N'-0x40: ch = GO_DOWN

```

```

5832     case 'B'-0x40: ch = GO_LEFT
5833     case 'F'-0x40: ch = GO_RIGHT
5834   }
5835   //fprintf(stderr,"B[%02X]\n",ch);
5836   switch( ch ){
5837     case 0:
5838       continue;
5839
5840     case 't':
5841       iin.Replace('j');
5842       continue;
5843     case 'X'-0x40:
5844       iin.Replace('j');
5845       continue;
5846
5847     case EV_TIMEOUT:
5848       iin.Redraw();
5849       if iin.pinJMode {
5850         fprintf(stderr,"\\J\r\n");
5851         iin.inJMode = true
5852       }
5853       continue;
5854     case GO_UP:
5855       if iin.lno == 1 {
5856         continue;
5857       }
5858       cmd_ok := gsh.cmdStringInHistory(iin.lno-1)
5859       if ok {
5860         iin.line = cmd
5861         iin.right = ""
5862         iin.lno = iin.lno - 1
5863       }
5864       iin.Redraw();
5865       continue;
5866     case GO_DOWN:
5867       cmd_ok := gsh.cmdStringInHistory(iin.lno+1)
5868       if ok {
5869         iin.line = cmd
5870         iin.right = ""
5871         iin.lno = iin.lno + 1
5872       }else{
5873         iin.line = "";
5874         iin.right = "";
5875         if iin.lno == iin.lastlno-1 {
5876           iin.lno = iin.lno + 1
5877         }
5878       }
5879       iin.Redraw();
5880       continue;
5881     case GO_LEFT:
5882       if 0 < len(iin.line) {
5883         xline,tail := delTailChar(iin.line)
5884         iin.line = xline
5885         iin.right = tail + iin.right
5886       }
5887       iin.Redraw();
5888       continue;
5889     case GO_RIGHT:
5890       if( 0 < len(iin.right) && iin.right[0] != 0 ){
5891         xright,head := delHeadChar(iin.right)
5892         iin.right = xright
5893         iin.line += head
5894       }
5895       iin.Redraw();
5896       continue;
5897     case EOF:
5898       goto EXIT;
5899     case 'R'-0x40: // replace
5900       dst := convA(iin.line+iin.right);
5901       iin.line = dst
5902       iin.right = "";
5903       iin.Redraw();
5904       continue;
5905     case 'T'-0x40: // just show the result
5906       readDict();
5907       romkanmode = !romkanmode;
5908       iin.Redraw();
5909       continue;
5910     case 'L'-0x40:
5911       iin.Redraw();
5912       continue;
5913     case 'K'-0x40:
5914       iin.right = "";
5915       iin.Redraw();
5916       continue;
5917     case 'E'-0x40:
5918       iin.line += iin.right
5919       iin.right = "";
5920       iin.Redraw();
5921       continue;
5922     case 'A'-0x40:
5923       iin.right = iin.line + iin.right
5924       iin.line = ""
5925       iin.Redraw();
5926       continue;
5927     case 'U'-0x40:
5928       iin.line = ""
5929       iin.right = ""
5930       iin.clearline();
5931       iin.Redraw();
5932       continue;
5933     case DEL_RIGHT:
5934       if( 0 < len(iin.right) ){
5935         iin.right,_ = delHeadChar(iin.right)
5936         iin.Redraw();
5937       }
5938       continue;
5939     case 0x7F: // BS? not DEL
5940       if( 0 < len(iin.line) ){
5941         iin.line,_ = delTailChar(iin.line)
5942         iin.Redraw();
5943       }
5944     else
5945       if( 0 < len(iin.right) ){
5946         iin.right,_ = delHeadChar(iin.right)
5947         iin.Redraw();
5948       }
5949     else
5950       continue;
5951     case 'H'-0x40:
5952       if( 0 < len(iin.line) ){
5953         iin.line,_ = delTailChar(iin.line)
5954         iin.Redraw();
5955       }
5956       continue;
5957     continue;
5958   }
5959   if( OnWindows && ch == '\n' ){
5960     continue;
5961   }
5962   if( ch == '\n' || ch == 'r' ){
5963     iin.line += iin.right;
5964     iin.right = "";
5965     iin.Redraw();
5966     //fprintf(stderr,"%c",ch);
5967     //fprintf(stderr,"\\r\\n"); // NL on Unix, CR on Windows
5968     AtConsoleLineTop = true
5969   }
5970   break;
5971   if MODE_CapsLock {
5972     if "A" <= ch && ch <= 'z' {
5973       ch = ch+'A'-'a'
5974     }
5975   }
5976   if MODE_LowerLock {
5977     if "A" <= ch && ch <= 'Z' {
5978       ch = ch+'a'-'A'
5979     }
5980   }
5981   iin.line += string(ch);
5982   iin.Redraw();
5983 }
5984 EXIT:
5985   return iin.line + iin.right;
5986 }
5987
5988 func getline_main(){
5989   line := xgetline(0,"",nil)
5990   fprintf(stderr,"$s\n",line);
5991   /* dp = strpbrk(line,`\r\n`);
5992   if( dp != NULL ){
```

```

5994     *dp = 0;
5995   }
5996
5997   if( 0 ){
5998     fprintf(stderr,"%n%d\n",int(strlen(line)));
5999   }
6000   if( lseek(3,0,0) == 0 ){
6001     if( romkamode() ){
6002       var buf [8*1024]byte;
6003       convert(line,buf);
6004       strcpy(line,buf);
6005     }
6006     write(3,line,strlen(line));
6007     ftruncate(3,lseek(3,0,SEEK_CUR));
6008     //fseek(3,0,SEEK_SET,offset+fd);
6009     lseek(3,0,SEEK_SET);
6010     close(3);
6011   }else{
6012     fprintf(stderr,"%r\ngetline: ");
6013     trans(line);
6014     //printf("%s\n",line);
6015     printf("\n");
6016   }
6017 */
6018 }
6019 //== end ===== getline
6020
6021 //
6022 // $USERHOME/.gsh/
6023 //   gsh-rc.txt, or gsh-configure.txt
6024 //   gsh-history.txt
6025 //   gsh-aliases.txt // should be conditional?
6026 //
6027 func (gshctx *GshContext)gshSetupPhomedir()(bool) {
6028   homedir,found := userHomeDir()
6029   if !found {
6030     fmt.Println("--E-- You have no UserHomeDir\n")
6031     return true
6032   }
6033   gshhome := homedir + "/" + GSH_HOME
6034   err2 := os.Stat(gshhome)
6035   if err2 != nil {
6036     err1 := os.Mkdir(gshhome,0700)
6037     if err3 != nil {
6038       fmt.Printf("--E-- Could not Create %s (%s)\n",
6039         gshhome,err3)
6040       return true
6041     }
6042     fmt.Println("--I-- Created %s\n",gshhome)
6043   }
6044   gshctx.GshHomeDir = gshhome
6045   return false
6046 }
6047 func setupGshContext()(GshContext,bool){
6048   //gshctx := setupGsh();
6049   gshPA := os.Pwd();
6050   cwd, // the starting directory
6051   os.Environ(), // environ[]
6052   //[]uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
6053   []*os.File(os.Stdin,os.Stdout,os.Stderr),
6054   nil, // OS specific
6055   cwd, := os.Getwd()
6056   gshctx := GshContext {
6057     cwd, // StartDir
6058     "", // GetLine
6059     []GchDirHistory { { cwd,time.Now(),0} }, // ChdirHistory
6060     gshPA,
6061     []GCommandHistory{}, //Something for invocation?
6062     GCommandHistory{}, // CmdCurrent
6063     false,
6064     []os.ProcessState{}, //[]int{},
6065     abusage(),
6066     "", // GshHomeDir
6067     TtyId(),
6068     false,
6069     false,
6070     []PluginInfo{},
6071     []string{},
6072     "-",
6073     "v",
6074     ValueStack{},
6075     GServer{"/"}, // LastServer
6076     "tcp4:5555", // LSPN
6077     cwd, // RWD
6078     CheckSum{},
6079   }
6080   err := gshctx.gshSetupHomedir()
6081   return gshctx, err
6082 }
6083 }
6084 func (gsh*GshContext)gshshellh(gline string)(bool){
6085   ghist := gsh.CmdCurrent
6086   ghist.WorkDir_ = os.Getwd()
6087   ghist.WorkDir_ = len(gsh.ChdirHistory)-1
6088   //"/"+os.Getenv("PWD")+"\n"+gsh.ChdirHistory[fd]\n",len(gsh.ChdirHistory))
6089   ghist.StartAt = time.Now()
6090   rusagev1 := Getrusagev()
6091   gsh.CmdCurrent.Foundfile = []string{}
6092   fin := gsh.tgshtshell(gline)
6093   rusagev2 := Getrusagev()
6094   ghist.Rusagev = append(gsh.Rusagev,rusagev1)
6095   ghist.Endat = time.Now()
6096   ghist.Cmdline = gline
6097   ghist.Foundfile = gsh.CmdCurrent.Foundfile
6098   /* record it but not show in list by default
6099   if len(gline) == 0 {
6100     continue
6101   }
6102   if gline == "hi" || gline == "history" { // don't record it
6103     continue
6104   }
6105   */
6106   gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6107   return fin
6108 }
6109 //}
6110 // <a name="main">Main loop</a>
6111 func scroll(gshctxGiven *GshContext) (_ GshContext) {
6112   gshctxBuf,err0 := setupGshContext()
6113   if err0 {
6114     return gshctxBuf;
6115   }
6116   gshctx := *gshctxBuf
6117
6118   //fmt.Printf("--I-- GSH_HOME=%s\n",gshctx.GshHomeDir)
6119   //resmap()
6120
6121 /*
6122   if false {
6123     gsh_getlinev, with_execline := "
6124       which(\"PATH\",[]string{\"which\", \"gsh-getline\", \"-s\"})
6125     if with_execline {
6126       gsh_getlinev[0] = toFullPath(gsh_getlinev[0])
6127       gshctx.Getline = toFullPath(gsh_getlinev[0])
6128     }else{
6129       fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6130     }
6131   */
6132 }
6133
6134 ghist0 := gshctx.CmdCurrent // something special, or gshrc script, or permanent history
6135 gshctx.CommandHistory = append(gshctx.CommandHistory,ghist0)
6136
6137 prevline := ""
6138 skipping := false
6139 for hix := len(gshctx.CommandHistory); ; {
6140   gline := gshctx.getline(hix,skipping,prevline)
6141   if skipping {
6142     if strings.Index(gline,"fi") == 0 {
6143       fmt.Printf("fi\n");
6144       skipping = false;
6145     }else{
6146       //fmt.Printf("%s\n",gline);
6147     }
6148     continue
6149   }
6150   if strings.Index(gline,"if") == 0 {
6151     //fmt.Printf("--D-- if start: %s\n",gline);
6152     skipping = true;
6153     continue
6154   }
6155   if false {

```



```

6318 //</span>
6319 var sijiminic = "<span id="gsh-sijim-dic">
6320 <data:text/dic;base64,>
6321 "PG1ldGgt2hcnNldD0iVVRGLTgiPg08dGV4dGFyZWRgY29scz04MCByb3dzPTQwPgovL3z1"+
6322 "cglnU2hb1lbKcc0lNRVxzG1jdglvbmyExz9yXHNTAglqaWipXHMVLzxMjAyMCwODMw"+
6323 "CNpCeOB1pzaGkJ44GXCMppCe0mAptaQnjb8khmJ44qCmp1CeOBmOCHq4eXU44KF"+
6324 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6325 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6326 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6327 "5QphCmtvewahlglsKa29rceWA1wp1b3gJ5YCLCm5hbmFgjXVhuxXgJNjIKhmFuYe1d5pef9g"+
6328 "7y7hcmFxCeOeoB1+OClp0awthcmE5J5YqbcjwvdGv4dGfPyZWE+Cg"+
6329 "awthcmFxCeOeoB1+OClp0awthcmE5J5YqbcjwvdGv4dGfPyZWE+Cg"+
6330 //</span>
6331
6332 var JA_JKDdic = "<span id="gsh-ja-jk1-dic">
6333 <data:text/dic;base64,>
6334 "ly92ZXJscCU158U1FanRpYZptb3jZwRKOwKSxWmjAyKGowDE5KsheLV4pL1nhhG94SVRT"+
6335 "CmtqamprGtqsa2p2sIOs1ueV1ApqamtjamW44GCCmtqanjYQRa2tgdnjhYYKamtg"+
6336 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6337 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6338 "amw44GICmtqat2rbAnjgOk2praw2J44GICmpramtrAnjgYOKa2trawJ44GCPmramps"+
6339 "az-te21sCeBgApramtrAcBqppq2prabjans2aztraw2J44GCSmpaq2psCeObPrqar2p2p"+
6340 "banjga4Kamtra2vJ44GCMppq2tgbAnjgOk2praw2J44GICmtqceObuPqa2sCeObuPqa2p+"
6341 "44KCMppra2p2AnjgOk2pramtrAcBqppq2pCeObC1Apaz2tsCeO1qamsCcOeC1qppaq2w44KLCmpg"+
6342 "amw44GICmtqap2pceoCjppq2psCeOjCyprramtrAnjgOk2praw2J44GCPmtqamtrAnjgpbEka2pqamwJ"+
6343 "44KCMppra2p2rbAnjgpkRa2pq2psCeObvAptra2w44KDCmtramprbAnjgpRa2pramtbAnj+"
6345 "gIEK";
6346 //</span>
6347
6348 //</span>
6349
6350 <style id="gsh-references-style">
6351 #references details { font-family:Georgia; }
6352 #references a { font-family:Georgia; }
6353 .wrap { white-space:normal; }
6354 </style>
6355 <details id="references"><summary>References</summary><div class="gsh-src">
6356 Web technology
6357 <a href="https://html.spec.whatwg.org">HTML: The Living Standard</a> (September 2020)
6358 <a href="https://html.spec.whatwg.org/dev">Developer Version</a>
6359
6360 <a href="https://drafts.csswg.org">CSS Working Group Editor Drafts</a> (September 2020)
6361 <a href="https://developer.mozilla.org/ja/docs/Web">MDN: web docs</a>
6362 <a href="https://developer.mozilla.org/ja/docs/Web/HTML/Element">HTML</a>
6363 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/CSS</a> : <span class="wrap">
6364 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors">selectors</a>
6365 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/Background_repeating">repeat</a></span>
6366 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/JavaScript">JavaScript</a>
6367 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
6368 <a href="https://mdn-web-dna.s3-us-west-2.amazonaws.com/MDN-Browser-Compatibility_Report_2020.pdf">MDN Browser Compatibility Report (2020)</a>
6370
6371 Go language (August 2020 / Go 1.15)
6372 <a href="https://golang.org/doc/effective_go#Programming_Language">Effective Go Programming Language</a>
6373 <a href="https://golang.org/pkg/>Packages</a>
6374 <a href="https://godoc.org/x/net/websocket">WebSocket</a>
6375
6376 <a href="https://stackoverflow.com">Stackoverflow</a>
6377 <!--
6378 <iframe src="https://golang.org" width="100%" height="300"></iframe>
6379 </div></details>
6380 </>
6381 </>
6382 <!--
6383 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>
6384 <!-- h2>the full of this HTML including the Go code is here.</h2 -->
6385 <details id="gsh-whole-view"><summary>Whole file</summary>
6386 <a name="whole-src-view"></a>
6387 <span id="src-frame"></span><!-- a window to show source code -->
6388 </details>
6389 <!--
6390 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
6391 <a name="style-src-view"></a>
6392 <span id="gsh-style-view"></span>
6393 </details>
6394 <!--
6395 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
6396 <a name="script-src-view"></a>
6397 <span id="gsh-script-view"></span>
6398 </details>
6399 <!--
6400 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Built-in data part</summary>
6401 <a name="gsh-data-frame"></a>
6402 <span id="gsh-data-view"></span>
6403 </details>
6404 <!--
6405 </div></details>
6406 </>
6407 </>
6408 </>
6409 <!--
6410 <div id="GshFooter0"></div>
6411 <!-- 2020-09-17 SatorixS, visible script ( -- >
6412 <script>document.write('visible script');</script>
6413 <style>.gjscript { font-family:Georgia; }</style>
6414 <pre id="gjscript_1" class="gjscript">function gitest1(){ alert('Hello GJSscript!'); }
6415 <pre>
6416 <pre>
6417 <script>
6418 <gjs> document.getElementById('gjscript_1');
6419 //eval(gjs.innerHTML);
6420 //gjs.innerHTML = "";
6421 </script>
6422 </details><!-- ----- END-OF-VISIBLE-PART ----- >
6423 </>
6424 <!--
6425 // 2020-09-06 added,
6426 https://developer.mozilla.org/en-US/docs/Web/CSS/:index
6427 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6428 -->
6429 <span id="GshGrid">(&lt;/&gt;)/<small>(Hit j k l h)</small></span>
6430 <span id="GStat"><br>
6431 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6432 <span id="GTop" onclick="showGshellPlane();"></span>
6433 <div id="RawTextViewer" onclick="hideRawTextViewer()></div>
6434 <div id="RawTextViewerClose" onclick="hideRawTextViewer()> CLOSE </div>
6435 <style id="GshStyleDef">
6440 #line-numbered table,tr,td {
6441 margin:0;
6442 padding:4px;
6443 spacing:0;
6444 border:1px;
6445 }
6446 textarea.LineNumber {
6447 font-size:12px;
6448 font-family:monospace,Courier New;
6449 color:#282;
6450 padding:4px;
6451 text-align:right;
6452 }
6453 textarea.LineNumbered {
6454 font-size:12px;
6455 font-family:monospace,Courier New;
6456 padding:4px;
6457 wrap:off;
6458 }
6459 #RawTextViewer{
6460 z-index:0;
6461 position:fixed; top:0px; left:0px;
6462 width:100%; xxheight:50px; xheight:0px;
6463 overflow:auto;
6464 color:#fff; background-color:rgba(128,128,256,0.2);
6465 font-size:10px; font-family:Georgia;
6466 spellcheck:false;
6467 }
6468 #RawTextViewerClose{
6469 z-index:0;
6470 position:fixed; top:-100px; left:-100px;
6471 color:#fff; background-color:rgba(128,128,256,0.2);
6472 font-size:10px; font-family:Georgia;
6473 white-space:pre;
6474 }
6475 #xxGshellPlane{
6476 z-index:0;
6477 position:fixed; top:0px; left:0px;
6478 width:100%; height:0px;
6479 overflow:auto;

```



```

7128     color:#ffff; background-color:rgba(0,64,0.7);
7129     text-align:left;
7130     vertical-align:middle;
7131   }
7132 </style>
7133
7134 <script id="gsh-script">
7135 // 2013_0909 added, permanent local storage
7136 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7137 var MyHistory = null;
7138 Permanent.localstorage;
7139 MyHistory = Permanent.getItem('MyHistory');
7140 if( MyHistory == null ){ MyHistory = "" }
7141 d = new Date();
7142 MyHistory += gettime() / 1000 + " " + document.URL + "\n" + MyHistory
7143 Permanent.setItem('MyHistory', MyHistory);
7144 //Permanent.setitem('MyHistory', MyHistory)
7145
7146 var GJLog_Win = null;
7147 var GJLog_Tab = null;
7148 var GJLog_Stat = null;
7149 var GJLog_Text = null;
7150 var GJWin_Mode = null;
7151 var FProductInterval = 0;
7152
7153 var GJ_FactoryID = -1;
7154 var GJFactory = null;
7155 if( e = document.getElementById('GJFactory_0') ){
7156   GJfactory_1.height = 0;
7157   GJfactory = e;
7158   e.setAttribute('class','GJFactory');
7159   var GJ_FactoryID = 0;
7160 }else{
7161   GJfactory = GJfactory_1;
7162   var GJ_FactoryID = 1;
7163 }
7164
7165 function GJFactory_Destroy(){
7166   gif = GJfactory;
7167   //gif = document.getElementById('GJFactory');
7168   //alert('gif'+gif);
7169   if( gif != null ){
7170     if( gif.childNodes != null ){
7171       for( i=0; i < gif.childNodes.length; i++ ){
7172         gif.removeChild(gif.childNodes[i]);
7173       }
7174     }
7175     gif.innerHTML = '';
7176     gif.style.width = 0;
7177     gif.style.height = 0;
7178     gif.removeAttribute('style');
7179     GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null;
7180     window.clearInterval(FProductInterval);
7181     return '-- Destroy: work product destroyed';
7182   }else{
7183     return '-- Destroy: work product not exist';
7184   }
7185 }
7186
7187 var TransMode = false;
7188 var OnkeyControl = false;
7189 var OnKeyShift = false;
7190 var OnKeyAlt = false;
7191 var OnKeyF = false;
7192 var OnKeyK = false;
7193 var OnKeyL = false;
7194
7195 function GJWin_OnkeyUp(ev){
7196   keycode = ev.code;
7197   if( keycode == 'ShiftLeft' ){
7198     OnKeyShift = false;
7199   }else{
7200     if( keycode == 'ControlLeft' ){
7201       OnKeyControl = false;
7202     }else{
7203       if( keycode == 'AltLeft' ){
7204         OnKeyAlt = false;
7205       }else{
7206         if( keycode == 'KeyJ' )( OnKeyJ = false )else
7207           if( keycode == 'KeyK' )( OnKeyK = false )else
7208             if( keycode == 'KeyL' )( OnKeyL = false );
7209       }
7210     }
7211   }
7212   ev.preventDefault();
7213 }
7214 function and(a,b){ if(a){ if(b){ return true; } return false; } }
7215 function GJWin_OnkeyDown(ev){
7216   keycode = ev.code;
7217   mode = '';
7218   key = '';
7219   if( keycode == 'ControlLeft' ){
7220     onKeyControl = true;
7221     ev.preventDefault();
7222   }else{
7223     if( keycode == 'ShiftLeft' ){
7224       OnKeyShift = true;
7225       ev.preventDefault();
7226     }else{
7227       if( keycode == 'AltLeft' ){
7228         ev.preventDefault();
7229         OnKeyAlt = true;
7230       }else{
7231         if( keycode == 'Backquote' ){
7232           TransMode = !TransMode;
7233           ev.preventDefault();
7234         }else{
7235           if( and(keycode == 'Space', OnKeyShift) ){
7236             TransMode = !TransMode;
7237             ev.preventDefault();
7238           }else{
7239             if( keycode == 'ShiftRight' ){
7240               TransMode = !TransMode;
7241             }else{
7242               if( keycode == 'Escape' ){
7243                 TransMode = true;
7244                 ev.preventDefault();
7245               }else{
7246                 if( keycode == 'Enter' ){
7247                   TransMode = false;
7248                 }else{
7249                   //ev.preventDefault();
7250                 }
7251               }
7252             }
7253             if( ev.altKey )( key += 'Alt+' );
7254             if( onKeyControl )( key += 'Ctrl+' );
7255             if( OnKeyShift )( key += 'Shift+' );
7256             if( and(keycode != 'KeyJ', OnKeyJ) )( key += 'J+' );
7257             if( and(keycode != 'KeyK', OnKeyK) )( key += 'K+' );
7258             if( and(keycode != 'KeyL', OnKeyL) )( key += 'L+' );
7259             key += keycode;
7260           }
7261           if( TransMode ){
7262             //mode = "[\u0343_201_0202]";
7263             JAAutf8 = JAAlineArray[0343_0201_0202];
7264             utf8dec = new TextDecoder('utf-8');
7265             JAa = utf8dec.decode(JAAutf8);
7266             mode = "[" + JAa + "]";
7267           }
7268         }
7269       }
7270     }
7271   }
7272   else{
7273     mode = '[---]';
7274   }
7275   //--- /gjmode.innerHTML = "[---]";
7276   GJWin_Mode.innerHTML = mode + ' ' + key;
7277   //alert('key'+keycode)
7278   ev.stopPropagation();
7279   //ev.preventDefault();
7280 }
7281
7282 function GJWin_OnScroll(ev){
7283   x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0);
7284   y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0);
7285   GJLog_append('OnScroll: x=' + x + ',y=' + y);
7286 }
7287
7288 document.addEventListener('scroll',GJWin_OnScroll)
7289 function GJWin_ONResize(ev){
7290   w = window.innerWidth

```

```

7290     h = window.innerHeight
7291     GJLog_append('OnResize: w=' + w + ',h=' + h)
7292   }
7293   window.addEventListener('resize', GJWin_OnResize)
7294
7295 var DragStartX = 0
7296 var DragStartY = 0
7297 function GJWin_DragStart(ev){
7298   // set the dragging position
7299   this.style.position = 'fixed'
7300   x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
7301   y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
7302   GJLog_Stat.value = 'DragStart: x=' + x + ',y=' + y
7303
7304   function GJWin_Drag(ev){
7305     x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
7306     this.style.left = x - DragStartX
7307     this.style.top = y - DragStartY
7308     this.style.zIndex = '30000'
7309     this.style.position = 'fixed'
7310     x = this.getBoundingClientRect().left.toFixed(0)
7311     y = this.getBoundingClientRect().top.toFixed(0)
7312     GJLog_Stat.value = 'x=' + x + ',y=' + y
7313     ev.preventDefault()
7314     ev.stopPropagation()
7315   }
7316   function GJWin_DragEnd(ev){
7317     x = ev.clientX; y = ev.clientY
7318     //x = ev.pageX; y = ev.pageY
7319     this.style.left = x - DragStartX
7320     this.style.top = y - DragStartY
7321     this.style.zIndex = '30000'
7322     this.style.position = 'fixed'
7323     if( true ){
7324       console.log('Dropped: '+this.nodeName+'#'+this.id+' x=' + x + ' y=' + y
7325         + ' parent=' + this.parentNode.id)
7326     }
7327     x = this.getBoundingClientRect().left.toFixed(0)
7328     y = this.getBoundingClientRect().top.toFixed(0)
7329     GJLog_Stat.value = 'x=' + x + ',y=' + y
7330     ev.preventDefault()
7331     ev.stopPropagation()
7332   }
7333   function GJWin_DragIgnore(ev){
7334     ev.preventDefault()
7335     ev.stopPropagation()
7336   }
7337   // 2020-09-15 let every object have console view!
7338   var txap0 = new Date();
7339   PrevReport = new Date()
7340   function GJLog_StatUpdate(){
7341     txa = GJLog_Stat;
7342     if( txa == null ){
7343       return;
7344     }
7345     tmlap0 = new Date();
7346     p = txa.parentNode;
7347     pw = txa.getBoundingClientRect().width;
7348     ph = txa.getBoundingClientRect().height;
7349     //txa.value += '#'+p.id+ 'pw=' + pw + ', ph=' + ph + '\n';
7350     txi = '#'+p.id+ 'pw=' + pw + ', ph=' + ph + '\n';
7351
7352     w = txa.getBoundingClientRect().width;
7353     h = txa.getBoundingClientRect().height;
7354     txi += 'w=' + w + ', h=' + h + '\n';
7355     txi += 'w=' + w + ', h=' + h + '\n';
7356
7357     //txa.value += '\n';
7358     //txa.value += DateShort() + '\n';
7359     txi += '\n';
7360     txi += DateShort() + '\n';
7361     tmlap1 = new Date();
7362
7363     txa.value += txi;
7364     tmlap2 = new Date();
7365
7366     // vertical centering of the last line
7367     sHeight = txa.scrollHeight - 30; // depends on the font-size
7368     tmlap3 = new Date();
7369
7370     txa.scrollTop = sHeight; // depends on the font-size
7371     tmlap4 = new Date();
7372
7373     now = tmlap0.getTime();
7374     if( PrevReport == 0 || 10000 <= now-PrevReport ){
7375       PrevReport = now;
7376       console.log('StatusBarUpdate:'
7377         + 'leng=' + txa.value.length + ' byte, '
7378         + 'tms=' + (tmlap4-tmlap0) + 'ms {'
7379         + 'tadd=' + (tmlap1-tmlap0) + ','
7380         + 'tcal=' + (tmlap3-tmlap2) + ','
7381         + 'scrl=' + (tmlap4-tmlap3) + '}'
7382       );
7383     }
7384   }
7385   GJWin_StatUpdate = GJLog_StatUpdate;
7386   function GJ_ShowTime(wid){
7387     //e = document.getElementById(wid);
7388     //console.log(wid.id+'.value.length=' + wid.value.length)
7389     if( e != null ){
7390       e.value = DateShort();
7391     }else{
7392       // should remove the Listener
7393     }
7394   }
7395   function GJWin_OnResizeTextarea(ev){
7396     this.value += 'resized: ' + '\n'
7397   }
7398   function GJ_NewConsole(wname){
7399     wid = wname + '_' + GJ_ConsoleID
7400     GJ_ConsoleID += 1
7401
7402     GJFactory.style.setProperty('width', '360px'); //GJFsize
7403     GJFactory.style.setProperty('height', '320px')
7404     e = GJFactory;
7405     console.log('GJF #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7406
7407     if( GJFactory.innerHTML == "" ){
7408       GJFactory.innerHTML = '<' + 'H3>' + GJ_Factory_ + GJ_FactoryID + '<' + '/' + 'H3' + '<' + 'hr>\n'
7409     }else{
7410       GJFactory.innerHTML += '<' + 'hr>\n'
7411     }
7412
7413     gjwin = GJLog_Win = document.createElement('span')
7414     gjwin.id = wid
7415     gjwin.setAttribute('class', 'GJWin')
7416     gjwin.setAttribute('draggable', 'true')
7417     gjwin.addEventListener('dragstart', GJWin_DragStart)
7418     gjwin.addEventListener('drag', GJWin_Drag)
7419     gjwin.addEventListener('dropend', GJWin_Drag)
7420     gjwin.addEventListener('dragover', GJWin_DragIgnore)
7421     gjwin.addEventListener('dragenter', GJWin_DragIgnore)
7422     gjwin.addEventListener('dragleave', GJWin_DragIgnore)
7423     gjwin.addEventListener('dragexit', GJWin_DragIgnore)
7424     gjwin.addEventListener('drop', GJWin_DragIgnore)
7425     gjwin.addEventListener('keydown', GJWin_OnKeyDown)
7426
7427     gjtab = GJLog_Tab = document.createElement('textarea')
7428     gjtab.addEventListener('keydown', GJWin_OnKeyDown)
7429     gjtab.style.readonly = true
7430     gjtab.contentEditable = false
7431     gjtab.value = ''
7432     gjtab.id = wid + '_Tab'
7433     gjtab.setAttribute('class', 'GJTab')
7434     gjtab.setAttribute('spellcheck', 'false')
7435     gjwin.appendChild(gjtab)
7436
7437     gjstat = GJLog_Stat = document.createElement('textarea')
7438     gjstat.addEventListener('keydown', GJWin_OnKeyDown)
7439     gjstat.id = wid + '_Stat'
7440     gjstat.value = DateShort()
7441     gjstat.setAttribute('class', 'GJStat')
7442     gjstat.setAttribute('spellcheck', 'false')
7443     gjwin.appendChild(gjstat)
7444
7445     gjicon = document.createElement('span')
7446     gjicon.addEventListener('keydown', GJWin_OnKeyDown)
7447     gjicon.id = wid + '_Icon'
7448     gjicon.innerHTML = '<font color="#ff44">J</font>'
7449     gjicon.setAttribute('class', 'GJIcon')
7450     gjicon.setAttribute('spellcheck', 'false')
7451     gjwin.appendChild(gjicon)

```

```

7452 gjtext = GJLog_Text = document.createElement('textareा')
7453 gjtext.addEventListener('keydown',GJWin_OnKeydown)
7454 gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7455 gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7456 gjtext.id = wid + '_Text'
7457 gjtext.setAttribute('class','GJText')
7458 gjtext.setAttribute('spellcheck','false')
7459 gjwin.appendChild(gjtext)
7460
7461
7462
7463 // user's mode as of IME
7464 gjmode = GJWin_Mode = document.createElement('textareा')
7465 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7466 gjmode.addEventListener('keyup',GJWin_OnKeyUp)
7467 gjmode.id = wid + '_Mode'
7468 gjmode.setAttribute('class','GJMode')
7469 gjmode.setAttribute('spellcheck','false')
7470 gjmode.innerHTML = '---'
7471 gjwin.appendChild(gjmode)
7472
7473 gjwin.zIndex = 30000
7474 GJfactory.appendChild(gjwin)
7475
7476 gjtab.scrollTop = 0
7477 gjstat.scrollTop = 0
7478
7479 //x = gwin.getBoundingClientRect().left.toFixed(0)
7480 //y = gwin.getBoundingClientRect().top.toFixed(0)
7481 //gwin.style.position = 'static'
7482 //gwin.style.left = 0
7483 //gwin.style.top = 0
7484
7485 //update = {'wid':value=rateShort()});
7486 update = '(GJ_showTime1("wid"))';
7487 //2020-09-19 this causes memory leaks
7488 //FProductInterval = window.setInterval(update,200)
7489 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
7490 //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
7491 FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7492 return update
7493 }
function xxxCF_StartClass(){
7494 GJLog_Min.style.removeProperty('width')
7495 GJLog_Tab.style.removeProperty('width')
7496 GJLog_Stat.style.removeProperty('width')
7497 GJLog_Text.style.removeProperty('width')
7498 return "Stripped classes"
7499 }
function islem(id){
7500 return document.getElementById(id) != null
7501 }
function GJLog_append(...args){
7502 txt = GJLog_Text;
7503 if (txt == null){
7504     return; // maybe GJLog element is removed
7505 }
7506 logs = args.join(' ');
7507 txt.value += logs + '\n';
7508 txt.scrollTop = txt.scrollHeight
7509 //GJLog_Status = DateShort()
7510 //window.addEventListener('time',GJLog_StatusUpdate)
7511 function test_GJ_Console(){
7512     window.setInterval(GJLog_StatusUpdate,1000);
7513     e = GJfactory();
7514     console.log('GJFO #' + e.id' from w=' + e.style.width + ', h=' + e.style.height)
7515     e.style.width = 360; //GJSize
7516     e.style.height = 320;
7517     console.log('GJFO #' + e.id' to w=' + e.style.width + ', h=' + e.style.height)
7518 }
/// test_GJ_Console();
7519
7520 var StopConsoleLog = true
7521 // 2020-09-15 added,
7522 // log should be saved to permanent memory
7523 //console.info = new Proxy(console.log,{ alert() })
7524 _console_log = console.log
7525 _console_info = console.info
7526 _console_warn = console.warn
7527 _console_error = console.error
7528 _console_exception = console.exception
7529 // above functions are inlined
7530 console.exception = function(...args){
7531     _console_exception(...args)
7532     alert('-- got console.exception("'+args+'")')
7533 }
7534 console.error = function(...args){
7535     _console_error(...args)
7536     alert('-- got console.error("'+args+'")')
7537 }
7538 console.warn = function(...args){
7539     _console_warn(...args)
7540     alert('-- got console.warn("'+args+'")')
7541 }
7542 console.info = function(...args){
7543     _console_info(...args)
7544     alert('-- got console.info("'+args+'")')
7545 }
7546 _console_info(...args)
7547
7548 console.xxlog = function(...args){ // rewrite xxlog to log to enable it
7549     _console_log(...args)
7550     if(StopConsoleLog ){
7551         return;
7552     }
7553     if( 0 <= args[0].indexOf('!') ){
7554         //alert('-- got console.log("'+args+'")')
7555     }
7556 }
7557 GJLog_append(...args)
7558
7559 //document.getElementById('GshFaviconIcon').href = GShellFavicon
7560 //document.getElementById('GshFaviconIcon'),href = GshellInsideIcon
7561 //document.getElementById('GshFaviconURL').href = ITSMoreQR
7562 //document.getElementById('GshFaviconURL').href = GSellLogo
7563
7564 // id of GShell HTML elements
7565 var E_BANNER = "GshBanner" // banner element in HTML
7566 var E_FOOTER = "GshFooter" // footer element in HTML
7567 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7568 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7569 var E_TODO = "gsh-todo" // TODO of GShell
7570 var E_DICT = "gsh-dict" // Dictionary of GShell
7571
7572 function bannerElem(){ return document.getElementById(E_BANNER); }
7573 function bannerStyleFunc(){ return bannerElem().style; }
7574 var bannerStyle = bannerStyleFunc()
7575 function GshSetImages(){
7576     document.getElementById('GshFaviconURL').href = GShellInsideIcon
7577     bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7578     //bannerStyle.backgroundImage = "url("+GShellFavicon+"")";
7579     //bannerStyle.backgroundImage = "url("+GShellFavicon+"")";
7580     //Menu.style.backgroundImage = "url("+GShellIcon+"")";
7581     //showFooter();
7582 }
7583 function GshInsideIconSetup(){
7584     GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7585     GMenu.style.zIndex = 1000000;
7586     //GMenu.style.left = window.innerWidth - 10
7587     GMENU.style.left = 0;
7588     GMENU.style.top = window.innerHeight - 90; // - 200
7589     window.addEventListener('resize',GshInsideIconSetup);
7590 }
7591 function footerElem(){ return document.getElementById(E_FOOTER); }
7592 function footerStyle(){ return footerElem().style; }
7593 //footerElem().style.backgroundImage="url("+ITSMoreQR+"")";
7594 //footerStyle().backgroundImage = "url("+ITSMoreQR+"")";
7595
7596 function html_fold(e){
7597     if( e.innerHTML == "Fold" ){
7598         e.innerHTML = "Unfold"
7599         document.getElementById('gsh-menu-exit').innerHTML=""
7600         GshFeatures.open = false
7601         document.getElementById('GshStatement').open=false
7602         document.getElementById('html-src').open=false
7603         document.getElementById('GINDEX').open=false
7604         document.getElementById('GOCODE').open=false
7605         document.getElementById('TODO').open=false
7606         document.getElementById('references').open=false
7607     }else{
7608         e.innerHTML = "Fold"
7609     }
7610 }

```

```

7614     document.getElementById('GshStatement').open=true
7615     GshFeatureA.style.display = true
7616     document.getElementById(E_GINDEX).open=true
7617     document.getElementById(E_GCODE).open=true
7618     document.getElementById(E_TODO).open=true
7619     document.getElementById('references').open=true
7620   }
7621   function html_pure(s){
7622     if( e.innerHTML == "Pure" ){
7623       document.getElementById('gsh').style.display=true
7624       //document.style.display = false
7625       e.innerHTML = "Unpure"
7626     }else{
7627       document.getElementById('gsh').style.display=false
7628       //document.style.display = true
7629       e.innerHTML = "Pure"
7630     }
7631   }
7632 }
7633
7634 var bannerIsStopping = false
7635 //NOTE: .com/ISREF/prop_style_backgroundposition.asp
7636 function shiftBG(){
7637   bannerIsStopping = !bannerIsStopping
7638   bannerStyle.backgroundPosition = "0 0";
7639 }
7640 // status should be inherited on Window Fork(), so use the status in DOM
7641 function html_stop(e,toggle){
7642   if( toggle ){
7643     if( e.innerHTML == "Stop" ){
7644       bannerIsStopping = true
7645       e.innerHTML = "Start"
7646     }else{
7647       bannerIsStopping = false
7648       e.innerHTML = "Stop"
7649     }
7650   }else{
7651     // update JavaScript variable from DOM status
7652     if( e.innerHTML == "Stop" ){ // shown if it's running
7653       bannerIsStopping = false
7654     }else{
7655       bannerIsStopping = true
7656     }
7657   }
7658 }
7659 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7660 //html_stop(bannerElem),false) // onInit.
7661
7662 https://www.w3schools.com/jsref/met_win_setinterval.asp
7663 var banNshift = 0;
7664 function conslog(str){
7665   //console.log(str);
7666 }
7667 function shiftBanner(){
7668   var now = new Date().getTime();
7669   bpos = ((now/10)1000).toFixed(0)+"px" + " 0px";
7670   if( ibannerIsStopping ){
7671     bannerStyle.backgroundPosition = bpos;
7672     //GshBanner.style.setProperty('background-position',bpos,'important');
7673     banNshift += 1;
7674     conslog("shiftBanner <"+GshBanner.nodeName+"> 'banNshift"
7675     + " "+now+");\n"
7676     //+
7677     //+ "stop"+bannerIsStopping
7678     + " pos="+bpos
7679     + " -> 'bannerStyle.backgroundPosition");
7680   }
7681 }
7682 function Banner_init(){
7683   console.log("-- Banner Shift init.");
7684   window.setInterval(shiftBanner,10); // onInit.
7685 }
7686
7687 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7688 var Mychildren = 0
7689 function html_fork(){
7690   ResetterPerf();
7691   Resetter();
7692   Reset_ShadingCanvas();
7693   GJfactory_Destroy();
7694   Mychildren += 1
7695   WinId = document.getElementById('gsh-WinId').innerHTML + "." + Mychildren;
7696   newwin = window.open("",WinId,"");
7697   newwin.document.getElementById('gsh');
7698   srchtml = src.outerHTML;
7699   newwin.document.write("<*<html>\n");
7700   newwin.document.write(srchtml);
7701   newwin.document.write("<*<html>\n");
7702   newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7703   newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7704   newwin.document.close();
7705   newwin.focus();
7706 }
7707 function html_close(){
7708   window.close()
7709 }
7710 function win_jump(win){
7711   //win = window.top;
7712   win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7713   if( win == null ){
7714     console.log("jump to window.opener(\"+win+\")Error)\n")
7715   }else{
7716     console.log("jump to window.opener(\"+win+\")\n")
7717     win.focus();
7718   }
7719 }
7720
7721 // 0.2.9 2020-0902 created cheksum of HTML
7722 CRC32UNIX = 0x04C11DB7 // Unix cksum
7723 function byteCRC32add(bigcrc,octstr,octlen){
7724   var crc = new Int32Array(1)
7725   crc[0] = bigcrc
7726
7727   let oit = 0
7728   for( ; oit < octlen; oit++ ){
7729     var oct = new Int8Array(1)
7730     oct[0] = octstr[oit]
7731     for( let bit = 0; bit < 8; bit++ ){
7732       //console.log("~~~CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oit+"."+bit+"]\n")
7733       ovf1 = crc[0] < 0 ? 1 : 0
7734       ovf2 = oct[0] < 0 ? 1 : 0
7735       ovf = ovf1 ^ ovf2
7736       oct[0] <<= 1
7737       crc[0] <<= 1
7738       if( ovf ){ crc[0] ^= CRC32UNIX }
7739     }
7740   }
7741   //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oit+"/"+octlen+"\n")
7742   return crc[0];
7743 }
7744 function strCRC32add(bigcrc,str1,strlen){
7745   var crc = new Uint32Array(1)
7746   crc[0] = bigcrc
7747   var code = new Uint8Array(strlen);
7748   for( i = 0; i < strlen; i++){
7749     code[i] = str1.charCodeAt(i) // not charAt() !!!!\n
7750     //console.log("~~~ "+code[i].toString(16)+" <== "+str1[i]+\n")
7751   }
7752   crc[0] = byteCRC32add(crc,code,strlen)
7753   //console.log("--CRC32 strAdd return crc="+crc[0]+\n")
7754   return crc[0];
7755 }
7756 function byteCRC32end(bigcrc,len){
7757   var crc = new Uint32Array(1)
7758   crc[0] = bigcrc
7759   var slen = new Uint32Array(4)
7760   let li = 0
7761   for( ; li < 4; ){
7762     slen[li] = len
7763     li += 1
7764     len >>= 8
7765     if( len == 0 ){
7766       break
7767     }
7768   }
7769   crc[0] = byteCRC32add(crc[0],slen,li)
7770   crc[0] ^= 0xFFFFFFFF
7771   return crc[0];
7772 }
7773 function strCRC32(str1,len){
7774   var crc = new Uint32Array(1)
7775   crc[0] = 0

```

```

7776     crc[0] = strCRC32add(0,str1,len)
7777     crc[0] = byteCRC32end(crc[0],len)
7778     //console.log(~"~CRC32 "+crc[0]+~"~len~\n")
7779     return crc[0]
7780 }
7781
7782 DestroyGJLink = null; // to be replaced
7783 DestroyFooter = null; // to be defined
7784 DestroyEventSharingCodeview = function dummy(){}
7785 Destroy_WirtualDesktop = function(){}
7786 DestroyNavButtons = function(){}
7787
7788 function getSourceText(){
7789     if(!DestroyFooter || null ) DestroyFooter();
7790     var sdoc = document.getElementById('GshVersion').innerHTML;
7791     sfavico = document.getElementById('GshFaviconURL').href;
7792     sbanner = document.getElementById('GshBanner').style.backgroundImage;
7793     spositi = document.getElementById('GshBanner').style.backgroundPosition;
7794
7795     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7796     if( DestroyGJLink != null ) DestroyGJLink();
7797     DestroyEventSharingCodeview();
7798     Destroy_WirtualDesktop();
7799     GshTopbar.innerHTML = "";
7800     DestroyIndexBar();
7801     DestroyNavButtons();
7802     ResetFooter();
7803     ResetAffView();
7804     Reset_ShadingCanvas();
7805
7806     // these should be removed by CSS selector or class, after sevaed to non-printed attribute
7807     GshBanner.removeAttribute('style');
7808     document.getElementById('GshMenuSign').removeAttribute('style');
7809     styleGMenu = GMenu.getAttribute("style")
7810     GMENU.removeAttribute("style");
7811     styleGStat = GStat.getAttribute("style")
7812     GStat.removeAttribute("style");
7813     styleGGrid = GGrid.getAttribute("style")
7814     GGrid.removeAttribute("style");
7815     styleGPos = GPos.getAttribute("style");
7816     GPos.removeAttribute("style");
7817     //styleRawTextViewer = RawTextViewer.getAttribute("style");
7818     //styleRawTextViewerClose = RawTextViewerClose.getAttribute("style");
7819     //styleRawTextViewerClose = RawTextViewerClose.getAttribute("style");
7820
7821     //styleLog = GLog.getAttribute("style");
7822     //GLog.innerHTML = "";
7823     styleGShellPlane = GShellPlane.getAttribute("style")
7824     GShellPlane.removeAttribute("style");
7825     styleRawTextViewer = RawTextViewer.getAttribute("style");
7826     RawTextViewer.removeAttribute("style");
7827     styleRawTextViewerClose = RawTextViewerClose.getAttribute("style");
7828     RawTextViewerClose.removeAttribute("style");
7829
7830     GshFaviconURL.href = "";
7831     if( iesem('ConfigIcon') ) ConfigIcon.src = "";
7832
7833 // it seems that interHTML and outerHTML generate style="" for these (?)
7834 //GshBanner.removeAttribute('style');
7835 //GshFooter.removeAttribute('style');
7836 //GshMenuSign.removeAttribute('style');
7837 //GshBanner.style="";
7838
7839
7840     textarea = document.createElement("textarea");
7841     srtchmt = document.getElementById('GshVersion').innerHTML;
7842     //textarea = document.createElement("textarea");
7843     // 2020-0910 ??? ... this causes inserting style="" to Banner and Footer,
7844     // with Chromium?/ after reloading from file:///
7845     textarea.innerHTML = srtchmt
7846     // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7847     var tlen = textarea.value.length;
7848     //rawtext = gsh.getTextContent() // this removes finclude <FILENAME> too
7849     var orgtext = ""
7850     + /*<+*/"\n" // lost preamble text
7851     + rawtext
7852     + /*<+*/"\n" // lost trail text
7853
7854
7855 tlen = orgtext.length
7856 //console.log("getSourceText: length="+tlen+"\n")
7857 document.getElementById('GshFaviconURL').href = sfavico;
7858
7859 document.getElementById('GshBanner').style.backgroundImage = sbanner;
7860 document.getElementById('GshBanner').style.backgroundPosition = spositi;
7861
7862 GStat.setAttribute("style",styleGStat)
7863 GMENU.setAttribute("style",styleGMENU)
7864 GGrid.setAttribute("style",styleGGrid)
7865 GPos.setAttribute("style",styleGPos)
7866 GLog.setAttribute("style",styleGLog)
7867
7868 GshGrid.setAttribute("style",styleGshGrid)
7869 GshellPlane.setAttribute("style",styleGShellPlane)
7870 RawTextViewer.setAttribute("style",styleRawTextViewer)
7871 RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
7872 canontext = orgtext.replace(' style=""','');
7873 // open="" too
7874 return canontext
7875 }
7876
7877 function getDigest(){
7878     var text = ""
7879     text = getSourceText()
7880     var digest = ""
7881     tlen = text.length
7882     digest = strCRC32(text,tlen) + " " + tlen
7883     return { text, digest }
7884 }
7885 function html_digest(){
7886     version = document.getElementById('GshVersion').innerHTML
7887     let {text, digest} = getDigest()
7888     alert("cksum: " + digest + " " + version)
7889 }
7890
7891 function charsin(str1,char){
7892     ln = 0;
7893     for( i = 0; i < str1.length; i++ ){
7894         if( str1.charCodeAt(i) == char.charCodeAt(0) )
7895             ln++;
7896     }
7897     return ln;
7898 }
7899
7900 //class digestElement extends HTMLElement {}
7901 //<script>customElements.define("digest",digestElement)</script>
7902
7903 function digest(e){
7904     result = "version" + GshVersion.innerHTML + '\n'
7905     result += "lines" + e.dataset.lines + '\n'
7906     + 'length' + e.dataset.length + '\n'
7907     + 'crc32u' + e.dataset.crc32u + '\n'
7908     + 'time' + e.dataset.time + '\n';
7909
7910     alert(result);
7911 }
7912
7913 function html_sign(e){
7914     if( RawTextViewer.style.zIndex == 1000 ){
7915         GshTopbar.innerHTML = "";
7916         ResetFooter();
7917         ResetShadingCanvas();
7918         DestroyIndexBar();
7919         DestroyNavButtons();
7920         DestroyEventSharingCodeview();
7921         Destroy_WirtualDesktop();
7922         Gfaction.Destroy();
7923         if( DestroyGJLink != null ) DestroyGJLink();
7924         if( DestroyGJLink != null ) DestroyGJLink();
7925         gsh.digest_.innerHTML = digest + " " + tlen
7926         text = getSourceText() // the original text
7927         tlen = text.length
7928         digest = strCRC32(text,tlen)
7929         //gsh.digest_.innerHTML = digest + " " + tlen
7930         //text = getSourceText() // the text with its digest
7931         Lines = charsin(text,'`')
7932
7933         name = "gph"
7934         sid = name + "-digest"
7935         d = new Date()
7936         signedAt = d.getTime()
7937

```

```

7938 sign = '/'+*+'<'+span+'\n'
7939   + '  id=' + id + '\n'
7940   + '  class=' + digest + '\n'
7941   + '  data-target-id=' + name + '\n'
7942   + '  data-crc32u=' + digest + '''\n'
7943   + '  data-length=' + tlen + '''\n'
7944   + '  data-lines=' + Lines + '''\n'
7945   + '  data-time=' + signedat + '''\n'
7946   + '>' + '/span>\n*' + '/\n'
7947
7948 text = sign + text
7949
7950 txhtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>
7951   + '<!--extarea cols=5 rows=' + Lines + ' class="LineNumber">
7952   for( i = 1; i < Lines; i++ ) {
7953     txhtml += i.toString() + '\n'
7954   }
7955 txhtml += ''
7956   + '<' + '/td><' + 'td>
7957   + '<!--extarea cols=150 rows=' + Lines + ' spellcheck="false" '
7958   + ' class="LineNumbered">
7959   + text + '<!--extarea>
7960   + '<' + '/td><' + 'tr></table>
7961
7962 for( i = 1; i <= 30; i++ ) {
7963   txhtml += '<br>\n'
7964 }
7965 RawTextViewer.innerHTML = txhtml
7966 RawTextViewer.spellcheck = false // ( spelcheck above seems ineffective)
7967
7968 btn = e
7969 e.style.color = "rgba(128,128,255,0.9)";
7970 y = e.getBoundingClientRect().top.toFixed(0)
7971 //y = e.getBoundingClientRect().height.toFixed(0)
7972 RawTextViewer.style.top = Number(y) + 30
7973 RawTextViewer.style.zIndex = 1000;
7974 RawTextViewer.style.height = window.innerHeight - 20;
7975 //RawTextViewer.style.opacity = 1.0;
7976 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7977 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
7978 RawTextViewer.style.zIndex = 1000;
7979 RawTextViewer.style.display = true;
7980
7981 if( RawTextViewerClose.style == null ){
7982   RawTextViewerClose.style = "";
7983 }
7984 RawTextViewerClose.style.top = Number(y) + 10
7985 RawTextViewerClose.style.left = 100;
7986 RawTextViewerClose.style.zIndex = 1001;
7987
7988 ScrollToElement(CurElement,RawTextViewerClose)
7990
7991 function hideRawTextViewer(){
7992   RawTextViewer.style.zIndex = 1000;
7993   RawTextViewer.style.zIndex = -100;
7994   RawTextViewer.style.opacity = 0.0;
7995   RawTextViewer.style = null;
7996   RawTextViewer.innerHTML = "";
7997
7998 GshMenusign.style.color = "rgba(255,128,128,1.0)";
7999 RawTextViewerClose.style.top = 0;
8000 RawTextViewerClose.style = null
8001 }
8002
8003 // source code view
8004 function frame_close(){
8005   srcframe = document.getElementById("src-frame");
8006   srcframe.innerHTML = "";
8007   //srcframe.style.cols = 1;
8008   srcframe.style.rows = 1;
8009   srcframe.style.height = 0;
8010   srcframe.style.display = false;
8011   src = document.getElementById("SrcTextarea");
8012   src.innerHTML = "";
8013   //src.cols = 0
8014   src.rows = 0
8015   src.style = false
8016   //alert("--closed--")
8017 }
8018 //<-- | <span onclick="html_view();">Source</span> -->
8019 //<-- | <span onclick="frame_close();">SourceClose</span> -->
8020 //<-- | <span>Download</span> -->
8021 function GshTopbar_init(){
8022   GshTopbar.innerHTML = "";
8023   ResetPerfMon();
8024   ResetAffView();
8025   Reset_ShadingCanvas();
8026   DestroyIconbinds();
8027   DestroyInitions();
8028   if( DestroyFooter != null ) DestroyFooter();
8029   document.getElementById('GshFaviconURL').href = "";
8030   oldsrc = document.getElementById("GENSRC");
8031   if( oldsrc != null ){
8032     alert("-I--(erasing old text)")
8033     oldsrc.innerHTML = "";
8034     return
8035   }else{
8036     //alert("--I--(no old text)")
8037   }
8038   styleBanner = GshBanner.getAttribute("style")
8039   GshBanner.removeAttribute("style")
8040   if( document.getElementById('GJC_1') ) GJC_1.remove()
8041
8042 GshFaviconURL.href = "";
8043 if( ConfigIcon ) ConfigIcon.src = "";
8044 GStat.removeAttribute("style")
8045 GshGrid.removeAttribute("style")
8046 GshMenusign.removeAttribute("style")
8047 //GPos.removeAttribute("style")
8048 //GPos.innerHTML = "";
8049 //GLog.removeAttribute("style")
8050 //GMenu.removeAttribute("style")
8051 GMenu.removeAttribute("style")
8052 GTop.removeAttribute("style")
8053 GShellPlane.removeAttribute("style")
8054 RawTextViewer.removeAttribute("style")
8055 RawTextViewerClose.removeAttribute("style")
8056
8057 if( DestroyGJLink != null ) DestroyGJLink();
8058 GJfactory_Destroy();
8059 Destroy_WirtualDesktop();
8060 DestroyEventSharingCodeview();
8061
8062 src = document.getElementById("gsh");
8063 srchtml = src.outerHTML;
8064 srcframe = document.getElementById("src-frame");
8065 srcframe.innerHTML = ""
8066   + '<!--cite><!--GENSRC\'' + '\n'
8067   + '<!--tab-size:4;\'' + '\n'
8068   + '#GENSRC textarea(tab-size:4);\'' + '\n'
8069   + '#GENSRC textarea(moz-tab-size:4);\'' + '\n'
8070   + '#GENSRC textarea(spellcheck:false);\'' + '\n'
8071   + '<!--\'' + '\n'
8072   + '#GENSRC textarea id="SrcTextarea" cols=100 rows=20 class="gsh-code" spellcheck="false">\'' + '\n'
8073   + '/<!--html\'' + '\n' // lost preamble text
8074   + srchtml
8075
8076   + '<!--/html\'' + '\n' // lost tab text
8077   + '<!--textarea\'' + '\n'
8078   + '<!--cite><!-- GENSRC -->\'' + '\n';
8079
8080 //srcframe.style.cols = 80;
8081 //srcframe.style.rows = 80;
8082
8083 GshBanner.setAttribute('style',styleBanner)
8084
8085 function fill_CSSView(){
8086   part = document.getElementById('GshStyleDef')
8087   view = document.getElementById('gsh-style-view')
8088   view.innerHTML = ''
8089   + '<!--textarea cols=100 rows=20 class="gsh-code">\'' + '\n'
8090   + part.innerHTML
8091   + '<!--/textarea\'' + '\n';
8092 }
8093 function fill_JavaScriptView(){
8094   jpart = document.getElementById('gsh-script')
8095   view = document.getElementById('gsh-script-view')
8096   view.innerHTML = ''
8097   + '<!--textarea cols=100 rows=20 class="gsh-code">\'' + '\n'
8098   + jpart.innerHTML
8099   + '<!--/textarea\'' + '\n';
8100

```

```

8100 }
8101 function fill_DataView(){
8102     part = document.getElementById('gsh-data')
8103     view = document.getElementById('gsh-data-view')
8104     view.innerHTML =
8105         + "<"+'textarea cols=100 rows=20 class="gsh-code">" +
8106         + part.innerHTML +
8107         + "<"/'+textarea>"
8108 }
8109 function jump_to_StyleView(){
8110     jsview = document.getElementById('html-src')
8111     jsview.open = true
8112     jsview = document.getElementById('gsh-style-frame')
8113     jsview.open = true
8114     fill_CSSView()
8115 }
8116 function jump_to_JavaScriptView(){
8117     jsview = document.getElementById('html-src')
8118     jsview.open = true
8119     jsview = document.getElementById('gsh-script-frame')
8120     jsview.open = true
8121     fill_JavaScriptView()
8122 }
8123 function jump_to_DataView(){
8124     jsview = document.getElementById('html-src')
8125     jsview.open = true
8126     jsview = document.getElementById('gsh-data-frame')
8127     jsview.open = true
8128     fill_DataView()
8129 }
8130 function jump_to_WholeView(){
8131     jsview = document.getElementById('html-src')
8132     jsview.open = true
8133     jsview = document.getElementById('gsh-whole-view')
8134     jsview.open = true
8135     frame_open()
8136 }
8137 function html_view(){
8138     html_stop();
8139 }
8140 banner = document.getElementById('GshBanner').style.backgroundImage;
8141 footer = document.getElementById('GshFooter').style.backgroundImage;
8142 document.getElementById('GshBanner').style.backgroundPosition = "center";
8143 document.getElementById('GshBanner').style.backgroundPositionY = "";
8144 document.getElementById('GshFooter').style.backgroundImage = "";
8145
8146 //srcwin = window.open("", "CodeView2");
8147 //srcwin = window.open("", "", "");
8148 srcwin.document.write("<span id='gsh'\n");
8149
8150 src = document.getElementById("gsh");
8151 srcwin.document.write("<"+style+"\n");
8152 srcwin.document.write("textarea{tab-size:4;}\n");
8153 srcwin.document.write("textarea{~o-tab-size:4;}\n");
8154 srcwin.document.write("textarea{border:none;outline:none;}\n");
8155 srcwin.document.write("textarea{border:none;outline:none;}\n");
8156 srcwin.document.write("textarea{border:none;outline:none;}\n");
8157 srcwin.document.write("textarea{border:none;outline:none;}\n");
8158 //srcwin.document.write("<span onclick=\"window.close();\">Close</span> | \n");
8159 //srcwin.document.write("<span onclick=\"html_stop()\">Run</span>\n");
8160 srcwin.document.write("<"+'input type="text" id="gsh-src-src" cols=100 rows=60'+"");
8161 srcwin.document.write("<"+'html'+">gsh");
8162 srcwin.document.write("<"+'span id="gsh'"+">");
8163 srcwin.document.write(src.innerHTML);
8164 srcwin.document.write("<"+'span></html>\n");
8165 srcwin.document.write("<"+'textarea'+">\n");
8166
8167 document.getElementById('GshBanner').style.backgroundImage = banner;
8168 document.getElementById('GshFooter').style.backgroundImage = footer;
8169
8170 sty = document.getElementById("gshStyleDef");
8171 srcwin.document.write("<"+style+"\n");
8172 srcwin.document.write(sty.innerHTML);
8173 srcwin.document.write("<"+style+">\n");
8174
8175 run = document.getElementById("gsh-script");
8176 srcwin.document.write("<"+'script>\n");
8177 srcwin.document.write(run.innerHTML);
8178 srcwin.document.write("<"+'script>\n");
8179
8180 srcwin.document.write("<"+'/span><"+'/html>\n"); // gsh span
8181 srcwin.document.close(); // gsh span
8182 srcwin.focus();
8183
8184 GSH = document.getElementById("gsh")
8185 //GSH.onclick = "alert('Ouch!')";
8186 //GSH.css = "#background-color:#eef;";
8187 //GSH.style = "background-color:#eef;";
8188 //GSH.style.display = false;
8189 //GSH.style.display = true;
8190 //alert('Ouch!');
8191 //GSH.style.display = true;
8192
8193 // 2020-0904 created, tentative
8194 //document.addEventListener('keydown',jgshCommand);
8195 //CurElement = GshStatement
8196 CurElement = GshMenu
8197 MemElement = GshMenu
8198
8199 function nextSib(e){
8200     n = e.nextSibling;
8201     for( i = 0; i < 100; i++ ){
8202         if( n == null ) {
8203             break;
8204         }
8205         if( n.nodeName == "DETAILS" ){
8206             return n;
8207         }
8208         n = n.nextSibling;
8209     }
8210     return null;
8211 }
8212 function previousSib(e){
8213     n = e.previousSibling;
8214     for( i = 0; i < 100; i++ ){
8215         if( n == null ) {
8216             break;
8217         }
8218         if( n.nodeName == "DETAILS" ){
8219             return n;
8220         }
8221         n = n.previousSibling;
8222     }
8223     return null;
8224 }
8225 function setColor(e,eName,eColor){
8226     if( e.hasChildNodes() ){
8227         s = e.childNodes;
8228         if( s != null ){
8229             for( ci = 0; ci < s.length; ci++ ){
8230                 if( s[ci].nodeName == eName ){
8231                     s[ci].style.color = eColor;
8232                     //s[ci].style.backgroundColor = eColor;
8233                     break;
8234                 }
8235             }
8236         }
8237     }
8238 }
8239 // https://docs.microsoft.com/en-us/previous-versions/hh781509(v=vs.85)
8240 function showCurElementPosition(ev){
8241 //    if( document.getElementById("GPos") == null ){
8242 //        return;
8243 //    }
8244 //    if( GPos == null ){
8245 //        return;
8246 //    }
8247 //    e = CurElement
8248 e = CurElement
8249 y = e.getBoundingClientRect().top.toFixed(0)
8250 x = e.getBoundingClientRect().left.toFixed(0)
8251
8252 h = ev + " "
8253 h += "y='"+y+"", "x='"+x+"', -- "
8254 h += "w='"+window.innerWidth+"", h='"+window.innerHeight+"', -- "
8255 //GPos.test = h
8256 //GPos.innerHTML = h
8257 // GPos.innerHTML = h
8258 }
8259
8260 function zero2(n){
8261     if( n < 10 ){

```

```

8262     return '0' + n;
8263   }else{
8264     return n;
8265   }
8266 }
8267 function DateHourMin(){
8268   d = new Date();
8269   //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes());
8270   return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
8271 }
8272 function DateShort0(d){
8273   return d.getFullYear()+
8274     '/' + zero2(d.getMonth())
8275     + '/' + zero2(d.getDate())
8276     + ':' + zero2(d.getHours())
8277     + ':' + zero2(d.getMinutes())
8278     + ':' + zero2(d.getSeconds())
8279 }
8280 function DateShort(){
8281   return DateShort0(new Date());
8282 }
8283 function DateLong0(ms){
8284   d = new Date();
8285   d.setTime(ms);
8286   return DateShort0(d)
8287     + '.' + d.getMilliseconds()
8288     + '.' + d.getTimezoneOffset()/60
8289     + '.' + d.getTime() + '.' + d.getMilliseconds()
8290 }
8291 function DateLong(){
8292   return DateLong0(new Date());
8293 }
8294 function GShellMenu(e){
8295   //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8296   //showGShellPlane()
8297   ConfigClick();
8298 }
8299 // placements of planes
8300 function GShellResize(ev){
8301   //if( document.getElementById("GMenu") != null ){
8302   //  GshInsideIconSetup();
8303   //  GMENU.style.left = window.innerWidth - 100
8304   //  GMENU.style.top = window.innerHeight - 90 - 200
8305   //  //console.log('place GMENU '+GMENU.style.left+' '+GMENU.style.top)
8306   //}
8307   GStat.style.width = window.innerWidth
8308   //if( document.getElementById("GPos") != null ){
8309   //  GPos.style.width = window.innerWidth
8310   //  GPos.style.top = window.innerHeight - 30; //GPos.style.height
8311   //}
8312   //if( document.getElementById("GLog") != null ){
8313   //  GLog.style.width = window.innerWidth
8314   //  GLog.innerHTML = ""
8315   //}
8316   //if( document.getElementById("GLog") != null ){
8317   //  GLog.innerHTML = "Resize: w=" + window.innerWidth +
8318   //  ", h=" + window.innerHeight
8319   //}
8320   //showCurElementPosition(ev)
8321 }
8322 function GShellResize(){
8323   GShellResizeX("[RESIZE]")
8324 }
8325 window.onresize = GShellResize
8326 var prevNode = null;
8327 var mouseMoveOverElement = false;
8328 function GJSH_OnMouseMove(ev){
8329   if( LogMouseMoveOverElement == false ){
8330     return;
8331   }
8332   x = ev.clientX
8333   y = ev.clientY
8334   d = new Date()
8335   t = d.getTime() / 1000
8336   if( document.elementFromPoint ){
8337     e = document.elementFromPoint(x,y)
8338     if( e != null ){
8339       if( e == prevNode ){
8340         console.log('Mo-' + t + '(' + x + ',' + y + ') ' +
8341           + e.nodeType + '+' + e.tagName + '#' + e.id)
8342         prevNode = e
8343       }
8344     }else{
8345       console.log(t + '(' + x + ',' + y + ') no element')
8346     }
8347   }else{
8348     console.log(t + '(' + x + ',' + y + ') no elementFromPoint')
8349   }
8350 }
8351 window.addEventListener('mousemove',GJSH_OnMouseMove);
8352 function GJSH_OnMouseMoveScreen(ev){
8353   x = ev.clientX
8354   y = ev.clientY
8355   d = new Date()
8356   t = d.getTime() / 1000
8357   console.log(t + '(' + x + ',' + y + ') no elementFromPoint')
8358 }
8359 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
8360
8361 function ScrollToElement(oe,ne){
8362   ne.scrollIntoView()
8363   ny = ne.getBoundingClientRect().top.toFixed(0)
8364   nx = ne.getBoundingClientRect().left.toFixed(0)
8365   //GLog.innerHTML = "["+ny+","+nx+"]"
8366   //window.scrollTo(nx,ny)
8367
8368   GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
8369   GshGrid.style.left = "250px";
8370   GshGrid.style.zIndex = 0
8371   if( fail == true ){
8372     oy = oe.getBoundingClientRect().top.toFixed(0)
8373     ox = oe.getBoundingClientRect().left.toFixed(0)
8374     y = e.getBoundingClientRect().top.toFixed(0)
8375     x = e.getBoundingClientRect().left.toFixed(0)
8376     window.scrollTo(ox,y)
8377     ny = ne.getBoundingClientRect().top.toFixed(0)
8378     nx = ne.getBoundingClientRect().left.toFixed(0)
8379     //GLog.innerHTML = "["+oy+","+ox+"]->["+y+","+x+"]->["+ny+","+nx+"]"
8380   }
8381 }
8382 function showGShellPlane(){
8383   if( GShellPlane.style.zIndex == 0 ){
8384     GShellPlane.style.zIndex = 1000;
8385     GShellPlane.style.left = 30;
8386     GShellPlane.style.height = 320;
8387     GShellPlane.innerHTML = DateLong() + "<br>" +
8388     "- History --> or " + MyHistory;
8389   }else{
8390     GShellPlane.style.zIndex = 0;
8391     GShellPlane.style.left = 0;
8392     GShellPlane.style.height = 50;
8393     GShellPlane.innerHTML = "";
8394   }
8395 }
8396 var SuppressGJShell = false
8397 function jgshCommand(keyevent){
8398   if( SuppressGJShell ){
8399     return
8400   }
8401   key = keyevent
8402   keyCode = key.keyCode
8403   //GStat.style.width = window.innerWidth
8404   GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8405   console.log("JGSh-Key:" +keycode+"(^~)//")
8406   if( keyCode == "Slash" ){
8407     console.log('(' + x + ',' + y + ') ')
8408     e = document.elementFromPoint(x,y)
8409     console.log('('+x+','+y+') ' + e.nodeType + '+' + e.tagName + '#' + e.id)
8410   }
8411   else if( keyCode == "Digit0" ){ // fold side-bar
8412     // "zero page"
8413     showGShellPlane();
8414   }
8415   else if( keyCode == "Digit1" ){ // fold side-bar
8416     primary.style.width = "44%"
8417     secondary.style.width = "0%"
8418     secondary.style.opacity = 0
8419   }
8420   GStat.innerHTML = "[Single Column View]"

```

```

8424     else
8425       if( keycode == "Digit2" ){ // unfold side-bar
8426         primary.style.width = "38"
8427         secondary.style.width = "38"
8428         secondary.style.opacity = 1
8429         GStat.innerHTML = "[Double Column view]"
8430       }
8431     if( keycode == "KeyU" ){ // fold/unfold all
8432       html_fold(GshMenuFold);
8433       location.href = "#"+CurElement.id;
8434     }
8435     if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8436       Curlement.open = !CurElement.open;
8437     }
8438     if( keycode == "ArrowLeft" ){ // unfold the element
8439       Curlement.open = true
8440     }
8441     if( keycode == "ArrowLeft" ){ // unfold the element
8442       Curlement.open = false
8443     }
8444     if( keycode == "KeyI" ){ // inspect the element
8445       e = CurElement
8446       //GLog.innerHTML =
8447       GJLog.append("Current Element: " + e + "<br>" +
8448         + "name=" + e.nodeName + ", " +
8449         + "children=" + e.childNodes.length + ", " +
8450         + "parent=" + e.parentNode.id + "<br>" +
8451         + "text=" + e.textContent)
8452       GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8453       return
8454     }
8455     if( keycode == "KeyM" ){ // memory the position
8456       MemElement = CurElement
8457     }
8458     if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8459       e = nextElement(Curlement)
8460       if( e != null ){
8461         setColor(e,"SUMMARY","#fff")
8462       oe = Curlement
8463       Curlement = e
8464       //location.href = "#"+e.id;
8465       ScrollToElement(oe,e)
8466     }
8467     if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8468       oe = Curlement
8469       e = previousCurElement()
8470       if( e != null ){
8471         setColor(e,"SUMMARY","#fff")
8472       oe = Curlement
8473       Curlement = e
8474       //location.href = "#"+e.id;
8475       ScrollToElement(oe,e)
8476     }
8477     else
8478       e = document.getElementById("GshBanner")
8479       if( e != null ){
8480         setColor(CurElement,"SUMMARY","#fff")
8481       CurElement = e
8482       ScrollToElement(oe,e)
8483     }
8484   }
8485   if( keycode == "KeyR" ){
8486     location.reload()
8487   }
8488   if( keycode == "KeyJ" ){
8489     GshGrid.style.top = '120px';
8490     GshGrid.innerHTML = '<>{Down}';
8491   }
8492   if( keycode == "KeyK" ){
8493     GshGrid.style.top = '0px';
8494     GshGrid.innerHTML = '{^}{Up}';
8495   }
8496   if( keycode == "KeyH" ){
8497     GshGrid.style.left = '0px';
8498     GshGrid.innerHTML = '{_}{Left}';
8499   }
8500   else
8501     if( keycode == "KeyL" ){
8502       //GLog.innerHTML +
8503       GJLog.append(
8504         'screen=' + screen.width + 'px' + '<br>' +
8505         'window=' + window.innerWidth + 'px' + '<br>' +
8506       )
8507     GshGrid.style.left = (document.documentElement.clientWidth - 160).toString(10) + 'px';
8508     GshGrid.innerHTML = '(@_){Right}';
8509   }
8510   if( keycode == "KeyS" ){
8511     html_stop(GshMenuStop,true)
8512   }
8513   else
8514     if( keycode == "KeyF" ){
8515       html_fork()
8516     }
8517   else
8518     if( keycode == "KeyC" ){
8519       window.close()
8520     }
8521   else
8522     if( keycode == "KeyD" ){
8523       html_digest()
8524     }
8525   else
8526     if( keycode == "KeyV" ){
8527       e = document.getElementById('gsh-digest')
8528       if( e != null ){
8529         showDigest(e)
8530       }
8531     }
8532   showCurElementPosition("[+key.code+]" --);
8533   //if( document.getElementById("GPos") != null ){
8534   //  GPos.innerHTML += "[+key.code+]" --
8535   //}
8536   //GShellResizeX("[+key.code+] --");
8537 }
8538 var initGSKC = false;
8539 function GShell_initKeyCommands(){
8540   if( initGSKC ) return; initGSKC = true;
8541   GShellResizeX("[INT]");
8542   DisplaySize = '-- Display: ' +
8543     'screen=' + screen.width + 'px, ' + 'window=' + window.innerWidth + 'px';
8544   GJLog.append(
8545     '-- Gshell: ' + GshVersion.innerHTML + '\n' +
8546     '-- Digest: ' + digest + '\n' +
8547     DisplaySize +
8548     '/+ "cbrs" + -- LastVisit:<br> ' + MyHistory
8549   )
8550   GShellResizeX(null);
8551 }
8552 //GShell_initKeyCommands();
8553 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8554 //Convert a string into an ArrayBuffer
8555 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8556 function str2ab(str) {
8557   const buf = new ArrayBuffer(str.length);
8558   const bufView = new Uint8Array(buf);
8559   for (let i = 0, strLen = str.length; i < strLen; i++) {
8560     bufView[i] = str.charCodeAt(i);
8561   }
8562   return buf;
8563 }
8564 function importPrivateKey(pem) {
8565   const binaryDerString = window.atob(pemContents);
8566   const binaryDerBuf = str2ab(binaryDerString);
8567   return window.crypto.subtle.importKey(
8568     "pkcs8",
8569     binaryDer,
8570     {
8571       name: "RSA-PSS",
8572       modulusLength: 2048,
8573       publicExponent: new Uint8Array([1, 0, 1]),
8574       hash: "SHA-256",
8575     }
8576 }

```

```

8586     },
8587     true,
8588     ["sign"]
8589   );
8590 }
8591 //importPrivateKey(ppem)
8592 //Key = {
8593 //buf = "abc"
8594 //enc = "yxxxxxxxx"; //crypto.publicEncrypt(key,buf)
8595 //b64 = btoa(enc)
8596 //dec = atob(b64)
8597 //Glog.innerHTML = "enc:" + b64 + ", dec:" + dec
8598 //SightGlass.innerHTML = "enc:" + enc + ", dec:" + dec
8599 </script>
8600 </>
8601
8602 //<!-- ===== Work { ===== -->
8603 <span id="SightGlass_WorkCodeSpan">
8604   /*
8605   <details id="SightGlass" open=""><summary>SightGlass</summary>
8606     <span id="SightGlass_Summary">SightGlass // 2020-10-23 SatoxITS </span>
8607   </details>
8608   <details><summary>meter</summary>
8609     <div id="SightGlass_1_Position" class="Sightglass_TextData">(0000, 0000)(0000 x 0000) </div>
8610     <div id="SightGlass_1_BGScroffset" class="Sightglass_TextData">(0000, 0000)(0000 x 0000) </div>
8611     <div id="SightGlass_1_BPosition" class="Sightglass_TextData">(0000, 0000)(0000 x 0000) </div>
8612     <div id="SightGlass_1_Wheel" class="Sightglass_TextData">Wheel</div>
8613   </details>
8614   <div id="SightGlass_1_Window" class="SightGlass_Window" draggable="true"></div>
8615   <style>
8616     .SightGlass_TextData {
8617       color:#000;
8618       font-size:10pt;
8619     }
8620     .SightGlass_Window {
8621       xxzoom:2;
8622       resize:both;
8623       width:600px;
8624       height:320px;
8625       border-color:rgba(200,200,200,0.5);
8626       background-size:200px;
8627       background-size:1280px 720px;
8628       background-image:url(WD-WallPaper03.png);
8629     }
8630     xbody {
8631       scroll-behavior:smooth;
8632     }
8633   </style>
8634   <script>
8635   // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
8636   var SGscrinX = 0;
8637   var SGinitX = 0;
8638   var SGinitY = 0;
8639   function SG_adjust(){
8640     xy = window.screenX + ' ' + window.screenY;
8641     xyw = window.innerWidth+'x'+window.innerHeight;
8642     xywh = 'xy(' + xy + ') wh' + xyw + ')';
8643     if( SightGlass.open ) SightGlass_1_BPosition.innerHTML = 'Browser: ' + xywh;
8644
8645     sg = SightGlass_1_Window;
8646     sg = sg.getBoundingClientRect();
8647     xy = sg.left.toFixed(0) + ' ' + sg.top.toFixed(0);
8648     wh = sg.width + 'x' + sg.height;
8649     xywh = 'xy(' + xy + ') wh' + wh + ')';
8650     if( SightGlass.open ) SightGlass_1_BPosition.innerHTML = 'SightGlass: ' + xywh;
8651
8652     //SightGlass_1_Window.style.backgroundPosition = sgr.left+'px ' + sgr.top+'px';
8653     if( SG_initX == 0 ){
8654       SGscrinX = window.screenX;
8655       SGscrinY = window.screenY;
8656       //SightGlass_1_Window.style.backgroundSize = '200%';
8657       //SightGlass_1_Window.style.backgroundImage = 'url("WD-WallPaper03.png")';
8658       SG_initX = sgr.left;
8659       SG_initY = sgr.top;
8660     }
8661     dx = SGscrinX - window.screenX;
8662     dy = SGscrinY - window.screenY;
8663     scroll = 'Screen: '+sx+'dx'+sy+'dy';
8664     if( SightGlass.open ) SightGlass_1_BGScroffset.innerHTML = scroll;
8665     dx += dx;
8666     dy += dy;
8667
8668     bgpos = dx+'px ' + dy+'px';
8669     if( SightGlass.open ) SightGlass_1_Window.style.backgroundPosition = bgpos;
8670     if( SightGlass.open ) SightGlass_1_BPosition.innerHTML = 'BGround: ' + SightGlass_1_Window.style.backgroundPosition;
8671
8672     wheels = 0;
8673     function SG_wheel(e){
8674       wheels += 1;
8675       SightGlass_1_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + '# ' + e.target.id;
8676       if( e.target.id == 'SightGlass_1_Window' ){
8677         e.preventDefault();
8678       }
8679     }
8680   }
8681   function SG_adjust1(){
8682     SG_adjust();
8683     //sampling 0;
8684     sampling = 20;
8685     //sampling = 200;
8686     window.setTimeout(SG_adjust1,sampling);
8687   }
8688   function SightGlass_Setup(){
8689     SG_adjust();
8690     window.addEventListener('resize',SG_adjust);
8691     window.addEventListener('mousemove',SG_adjust);
8692     window.addEventListener('scroll',SG_adjust);
8693     window.addEventListener('wheel',SG_wheel,{passive:false});
8694     //window.moveTo(0,0);
8695
8696     // if focused
8697     //window.setInterval(SG_adjust,1);
8698     window.setTimeout(SG_adjust1,1);
8699   }
8700
8701   // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
8702   function onWindowMove(e){
8703     const { BrowserWindow } = require('electron');
8704     const currentWindow = BrowserWindow.getFocusedWindow();
8705     //currentWindow.on('move',function(){ SG_adjust(); });
8706     currentWindow.addEventListener('move',SG_adjust);
8707   }
8708   </script>
8709   <input id="SightGlass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8710   <input id="SightGlass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8711   <input id="SightGlass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8712   <span id="SightGlass_WorkCodeView"></span>
8713   <script id="SightGlass_WorkCode"></script>
8714   <script>
8715     function SightGlass_openWorkCodeView(){
8716       showHtmlCode(SightGlass_WorkCodeView,SightGlass_WorkCodeSpan);
8717     }
8718     SightGlass_WorkCodeViewOpen.addEventListener('click',SightGlass_showWorkCode);
8719   </script>
8720   SightGlass_OpenWorkCodeView(); // should be invoked by an event
8721
8722 </script>
8723 </details>
8724 <!-- SightGlass_WorkCodeSpan -->
8725 <!-- //</span>
8726 <!-- ===== Work -->
8727
8728
8729 <!-- ----- GJConsole BEGIN ( ----- -->
8730 <span id="GjC" data-title="GJConsole" data-author="sato@its-more.jp">
8731   <details><summary>Gj Console</summary>
8732     <p>
8733       <span id="GjE_RootNode"></span>
8734       <span id="GjC1_Container"></span>
8735     </p>
8736   </details>
8737 <!-- ----- GJConsole END ( ----- -->
8738 <span id="GjConsoleStyle">
8739   .GjConsole {
8740     z-index:1000;

```

```

8748     width:400; height:200px;
8749     margin:2px;
8750     color:#fff; background-color:#66a;
8751     font-size:12px; font-family:monospace,Courier New;
8752   }
8753 </style>
8754 
8755 <script id="GJConsoleScript" class="GJConsole">
8756   var PS1 = " "
8757   function GJC_KeyDown(keyevent){
8758     key = keyevent.code;
8759     if( key == "Enter" ){
8760       GJC_Command(this)
8761       this.value += "\n" + PS1 // prompt
8762     } else if( key == "Escape" ){
8763       SuppressGJShell = false
8764       GahMenu.focus() // should be previous focus
8765     }
8766   }
8767   var GJC_SessionId
8768   function GJC_SetSessionId(){
8769     var xd = new Date();
8770     GJC_SessionId = xd.getTime() / 1000
8771   }
8772 
8773 GJC_SetSessionId()
8774 function GJC_Memory(mem,args,text){
8775   argv = args.split(" ")
8776   cmd = argv[0]
8777   argv.shift()
8778   args = argv.join(' ')
8779   ret = ""
8780 
8781   if( cmd == 'clear' ){
8782     Permanent.setItem(mem,'')
8783   }else{
8784     if( cmd == 'read' ){
8785       ret = Permanent.getItem(mem)
8786     }else{
8787       if( cmd == 'save' ){
8788         val = Permanent.getItem(mem)
8789         if( val == null ){ val = "" }
8790         d = new Date();
8791         val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
8792         val += text.value
8793         Permanent.setItem(mem,val)
8794       }else{
8795         if( cmd == 'write' ){
8796           val = Permanent.getItem(mem)
8797           if( val == null ){ val = "" }
8798           d = new Date();
8799           val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
8800           Permanent.setItem(mem,val)
8801       }else{
8802         ret = "Commands: write | read | save | clear"
8803       }
8804     }
8805   }
8806   // -- 2020-09-14 added TableEditor
8807   var GJE_Commandment = null; //GJE_RootNode
8808   var GJE_NodeSelected = null
8809   GJE_TableNo = 1
8810   function GJE_StyleKeyCommand(kev){
8811     keycode = kev.code
8812     console.log('GJE-Key: '+keycode)
8813     if( keycode == 'Escape' ){
8814       GJE_Setstyle(this);
8815     }
8816     kev.stopPropagation()
8817     // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
8818   }
8819   var GJE_CommandMode = false
8820   function GJE_TableKeyCommand(kev,tab){
8821     wasCmdMode = GJE_CommandMode
8822     key = kev.code
8823     if( key == 'Escape' ){
8824       console.log("To command mode: "+tab.nodeName+"#"+tab.id)
8825       //tab.setAttribute('contenteditable','false')
8826       tab.style.caretColor = "blue"
8827       GJE_CommandMode = true
8828     }else{
8829       if( key == "KeyA" ){
8830         tab.style.caretColor = "red"
8831         GJE_CommandMode = false
8832       }else{
8833         if( key == "KeyY" ){
8834           tab.style.caretColor = "red"
8835           GJE_CommandMode = false
8836         }else{
8837           if( key == "KeyO" ){
8838             tab.style.caretColor = "red"
8839             GJE_CommandMode = false
8840           }else{
8841             if( key == "KeyJ" ){
8842               console.log("ROW-DOWN")
8843             }else{
8844               if( key == "KeyK" ){
8845                 console.log("ROW-UP")
8846               }else{
8847                 if( key == "KeyF" ){
8848                   console.log("COL-FORW")
8849                 }else{
8850                   if( key == "KeyB" ){
8851                     console.log("COL-BACK")
8852                   }
8853                 }
8854               }
8855             }
8856           }
8857         }
8858       }
8859     }
8860     kev.stopPropagation()
8861     if( wasCmdMode ){
8862       kev.preventDefault()
8863     }
8864   }
8865   function GJE_DragEvent(ev,elem){
8866     x = ev.clientX
8867     y = ev.clientY
8868     console.log("Dragged: "+this.nodeName+"#"+this.id+' x="'+x+' y='+y)
8869   }
8870   // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
8871   // https://www.w3.org/TR/uievents/#events-mouseevents
8872   function GJE_DropEvent(ev,elem){
8873     x = ev.clientX
8874     y = ev.clientY
8875     this.style.x = x
8876     this.style.y = y
8877     this.style.position = 'absolute' // 'fixed'
8878     this.eventMode = gsh // just for test
8879     console.log("Dropped: "+this.nodeName+"#"+this.id+' x="'+x+' y='+y)
8880     + parent#+this.parentNode.id)
8881   }
8882   function GJP_SettableStyle(ev){
8883     this.innerHTML = this.value; // sync. for external representation?
8884     if(false){
8885       std = this.parentNode.id+this.id
8886       // and remove _span" at the end
8887       e = document.getElementById(std)
8888       //alert('SetTableStyle #' +e.id+'\n'+this.value)
8889       if( e != null ){
8890         e.innerHTML = this.value
8891       }else{
8892         console.log('Style Not found: '+std)
8893       }
8894     }
8895     //alert('event StopPropagation: '+ev)
8896   }
8897   function setCSSofClass(cclass,cstyle){
8898     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
8899     rlen = ss.cssRules.length;
8900     let tabrule = null;
8901     rulex = -1;
8902     // should skip white space at the top of cstyle
8903     sel = cstyle.charAt(0);
8904     selector = sel+cclass;
8905     console.log("- search style rule for "+selector)
8906     for(let i = 0; i < rlen; i++){
8907       cr = ss.cssRules[i];
8908       console.log('CSS rule ['+i+'/'+rlen+'] '+cr.selectorText);
8909       if( cr.selectorText == selector ){ // css class selector
8910         tabrule = ss.cssRules[i];
8911         console.log('CSS rule found for:['+i+'/'+rlen+'] '+selector);
8912         ss.deleteRule(i);
8913       }
8914     }
8915     rlen = ss.cssRules.length;

```



```

9072
9073 if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
9074   if( GJE_NodeSaved == null ){
9075     xn = GJE_RootNode
9076     GJE_RootNode = GJE_NodeSaved
9077     GJE_NodeSaved = xn
9078     ret = "- did undo"
9079   }else{
9080     ret = "-- could not undo"
9081   }
9082   return ret
9083 }
9084 GJE_NodeSaved = GJE_RootNode.cloneNode()
9085 if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
9086   if( argv[0] == null ){
9087     ne = GJE_RootNode
9088   }else{
9089     if( argv[0] == '..' ){
9090       ne = cwe.parentNode
9091     }else{
9092       ne = document.getElementById(argv[0])
9093     }
9094     if( ne != null ){
9095       GJE_CurElement = ne
9096       ret = "- current node: " + ne.id
9097     }else{
9098       ret = "-- not found: " + argv[0]
9099     }
9100   }
9101   if( cmd == '.mkt' || cmd == '.mktable' ){
9102     makeTable(argv)
9103   }
9104   else{
9105     if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
9106       ne = document.createElement(argv[0])
9107       //ne.id = argv[0]
9108       ret = "- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9109       cwe.appendChild(ne)
9110       if( cmd == '.m' || cmd == '.mk' ){
9111         GJE_CurElement = ne
9112       }
9113     }else{
9114       if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
9115         cwe.id = argv[0]
9116       }
9117     }
9118     if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
9119     }else{
9120       if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
9121         s = argv.join('')
9122         cwe.innerHTML = s
9123       }
9124     }
9125     if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
9126       cwe.setAttribute(argv[0],argv[1])
9127     }else{
9128       if( cmd == '.l' ){
9129         ret = cwe.innerHTML
9130       }else{
9131         if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
9132           ret = cwe.innerHTML
9133         }else{
9134           if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
9135             ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
9136             for( we = cwe.parentNode; we != null; ){
9137               ret += " " + we.nodeType + " " + we.tagName + " " + we.id
9138               we = we.parentNode
9139             }
9140           }else{
9141             ret = "Command: mk | rm \n"
9142             ret += " pw -- print current node\n"
9143             ret += " mx type -- make node with name and type\n"
9144             ret += " nn name -- set the id #name of current node\n"
9145             ret += " rm name -- remove named node\n"
9146             ret += " cd name -- change current node\n"
9147           }
9148         }
9149       }
9150     }
9151     /alert(ret)
9152     return ret
9153   }
9154   function GJC_Command(text){
9155     lines = text.value.split('\n')
9156     line = lines[lines.length-1]
9157     argv = line.split(' ')
9158     text.value += '\n'
9159     if( argv[0] == '$' ) argv.shift()
9160     args0 = argv.join(' ')
9161     argv.shift()
9162     argv.shift()
9163     args = argv.join(' ')
9164     if( cmd == 'nolog' ){
9165       StopConsoleLog = true
9166     }else{
9167       if( cmd == 'new' ){
9168         if( argv[0] == 'table' ){
9169           argv.shift()
9170           console.log(argv.join(' '))
9171           text.value += makeTable(argv)
9172         }else{
9173           if( argv[0] == 'console' ){
9174             text.value += GJ_NewConsole('GJ_Console')
9175           }else{
9176             text.value += '-- new ( console | table )'
9177           }
9178         }
9179       }else{
9180         if( cmd == 'strip' ){
9181           //text.value += GJF_StripClass()
9182         }else{
9183           if( cmd == 'css' ){
9184             sel = '#table_1'
9185             if(argv[0]==''0')
9186               rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
9187             else
9188               rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}';
9189             document.styleSheets[3].deleteRule(0);
9190             document.styleSheets[3].insertRule(rule1,0);
9191             text.value += 'CSS rule added: '+rule1
9192           }else{
9193             if( cmd == 'print' ){
9194               e = null;
9195               if( e == null ){
9196                 e = document.getElementById('GJFactory_0')
9197               }else{
9198                 e = document.getElementById('GJFactory_1')
9199               }
9200               if( e == null ){
9201                 e = document.getElementById('GJFactory_0')
9202               }else{
9203                 if( e != null ){
9204                   text.value += e.outerHTML
9205                 }else{
9206                   text.value += "Not found: " + id
9207                 }
9208               }
9209             }else{
9210               text.value += GJE_RootNode.outerHTML
9211             //text.value += e.innerHTML
9212           }
9213         }
9214       }
9215     }
9216     if( cmd == 'save' ){
9217       e = document.getElementById('GJFactory')
9218       Permanent.setItem('GJFactory-1',e.innerHTML)
9219       text.value += " Saved GJFactory"
9220     }else{
9221       if( cmd == 'load' ){
9222         gif = Permanent.getItem('GJFactory-1')
9223         e = document.getElementById('GJFactory')
9224         e.innerHTML = gif
9225         // must restore EventListener
9226         text.value += " EventListener was not restored"
9227       }
9228       if( cmd.charAt(0) == '.' ){
9229         argv0 = args0.split(' ')
9230         text.value += GJE_NodeEdit(argv0)
9231       }
9232       if( cmd == 'cont' ){
9233         bannerIsStopping = false
9234         GshMenuStop.innerHTML = "Stop"
9235     }

```

```

9234 }else
9235 if( cmd == 'date' ){
9236   text.value += DateLong()
9237 }else
9238 if( cmd == 'echo' ){
9239   text.value += args
9240 }else
9241 if( cmd == 'fork' ){
9242   XML_Fork()
9243 }else
9244 if( cmd == 'last' ){
9245   text.value += MyHistory
9246   //h = document.createElement("span")
9247   //h.innerHTML = MyHistory
9248   //tx.value += h.innerHTML
9249   //tx = MyHistory.replace("\n","");
9250   //text.value += tx.replace("<"+br>","\n") + "xxxx<"+br>yyyy"
9251 }else
9252 if( cmd == 'ne' ){
9253   text.value += GJE_NodeEdit(argv)
9254 }else
9255 if( cmd == 'reload' ){
9256   location.reload()
9257 }else
9258 if( cmd == 'mem' ){
9259   text.value += GJC_Memory('GJC_Storage',args,text)
9260 }else
9261 if( cmd == 'stop' ){
9262   bannerIsStopping = true
9263   GshMenuStop.innerHTML = "Start"
9264 }else
9265 if( cmd == 'who' ){
9266   text.value += "SessionId=" + GJC_SessionId + " " + document.URL
9267 }else
9268 if( cmd == 'wall' ){
9269   text.value += GJC_Memory('GJC_Wall','write',text)
9270 }else
9271 {
9272   text.value += "Commands: help | echo | date | last \n"
9273   + '          new | save | load | mem \n'
9274   + '          who | wall | fork | nife'
9275 }
9276 }

9277 function GJC_Input(){
9278   if( this.value.endsWith("\n") ){ // remove NL added by textarea
9279     this.value = this.value.slice(0,this.value.length-1)
9280   }
9281 }
9282 }

9283 var GJC_Id = null
9284 function GJC_Resize(){
9285   GJC_Id.style.zIndex = 2000
9286   //GJC_Id.style.width = window.innerWidth - 16
9287   GJC_Id.style.height = window.innerHeight - 30;
9288   GJC_Id.style.height = 300;
9289   GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9290   GJC_Id.style.color = "rgba(255,255,255,1.0)"
9291 }
9292 }

9293 function GJC_FocusIn(){
9294   this.spellcheck = false
9295   SuppressGShell = true
9296   this.onkeydown = GJC_Keydown
9297   GJC_Resize()
9298 }
9299 function GJC_FocusOut(){
9300   SuppressGShell = false
9301   this.removeEventlistener('keydown',GJC_Keydown);
9302   window.addEventListener('resize',GJC_Resize);
9303 }

9304 function GJC_Onstorage(e){
9305   //alert('Got Message')
9306   //GJC.value += "\n"((ReceivedMessage))\n"
9307 }
9308 window.addEventListener('storage',GJC_Onstorage);
9309 //window.addEventListener('storage',()=>(alert('GotMessage')))

9310 function GJC_Setup(gjcid){
9311   /gjcid.style.width = gsh.getBoundingClientRect().width
9312   gjcid.style.width = '100%';
9313   gjcid.value = "GShell Console // " + GshVersion.innerHTML + "\n"
9314   /gjid.value = "Date: " + DateLong() + "\n"
9315   gjcid.style.position = 'relative';
9316   gjcid.onfocus = GJC_FocusIn
9317   gjcid.addEventListener('input',GJC_Input);
9318   gjcid.addEventListener('focusout',GJC_FocusOut);
9319   GJC_Id = gjcid
9320 }

9321 function GJC_Clear(id){
9322 }
9323 function GJConsole_initConsole(){
9324   if( document.getElementById("GJC_0") != null ){
9325     GJC_Setup(GJC_0)
9326   }else{
9327     GJC_Container.innerHTML = '<'+
9328       'textarea id="GJC_1" class="GJConsole"><'+/textarea>';
9329     GJC_Setup(GJC_1)
9330     factory = document.createElement('span');
9331     gsh.appendChild(factory)
9332     GJE_RootNode = factory;
9333     GJE_CurElement = GJE_RootNode;
9334   }
9335 }

9336 var initGJCF = false;
9337 function GJConsole_initFactory(){
9338   if( initGJCF ) { return; } initGJCF = true;
9339   Gshell_initKeyCommands();
9340   GJConsole_initConsole();
9341 }

9342 // TODO! focus handling
9343 </details>
9344 <style>
9345   GJ_StyleEditor {
9346     font-size:9pt !important;
9347     font-family:Courier New, monospace !important;
9348   }
9349   </style>
9350 <details>
9351   <span>
9352     <!-- ----- GJConsole END -->
9353   </span>
9354   <br>
9355   <span id="BlinderText">
9356     <!-- ----- GJConsole END -->
9357   </span>
9358   <span id="BlinderTextStyle">
9359     #GJC_0 {
9360       position: absolute; z-index:5000;
9361       position:relative;
9362       display:block;
9363       left:9px;
9364       color:#fff;
9365       width:800px; height:300px; resize:both;
9366       margin:10px; padding:4px;
9367       background-color:rgba(200,200,200,0.5) !important;
9368     }
9369     .MsgText {
9370       width:578px !important;
9371       resize:both !important;
9372       color:#000 !important;
9373     }
9374     .GjNote {
9375       font-family:Georgia !important;
9376       font-size:13pt !important;
9377       color:#22a !important;
9378     }
9379     .textField {
9380       display:inline;
9381       border:0.5px solid #444;
9382       border-radius:3px;
9383       color:#000; background-color:#fff;
9384       width:106px; height:18pt;
9385       margin:2px;
9386       padding:2px;
9387       resize:none;
9388       vertical-align:middle;
9389       font-size:10pt; font-family:Courier New;
9390     }
9391     .textLabel {
9392       border:0px solid #000 !important;
9393     }
9394   </span>
9395 
```

```

9396     background-color:rgba(0,0,0,0);
9397   }
9398   .texture {
9399     width:300px !important;
9400     border:0px solid #000 !important;
9401     background-color:rgba(0,0,0,0);
9402   }
9403   .visibleText {
9404   }
9405   .BlinderText {
9406     color:#000; background-color:#eee;
9407   }
9408   .joinbutton {
9409     font-family:Georgia !important;
9410     font-size:1pt;
9411     line-height:1.1;
9412     height:18pt;
9413     width:50pt;
9414     padding:3pt !important;
9415     text-align:center !important;
9416     border:#000 1px !important;
9417     border-radius:5px;
9418     color:#fff; background-color:#4a4 !important;
9419     vertical-align:middle !important;
9420   }
9421   .Sendbutton {
9422     vertical-align:top;
9423   }
9424   .ws_log {
9425     font-size:10pt;
9426     color:#000 !important;
9427     line-height:1.8;
9428     background-color:rgba(255,255,255,0.7) !important;
9429     font-family:Courier New,monospace !important;
9430     width:99.3%;
9431     white-space:pre;
9432   }
9433 
```

</style>

```

9434 <!-- Form autofill test
9435 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafrm/formLogin" size="80">
9436 <form method="POST" id="xxform" action="https://192.168.10.1/boafrm/formLogin">
9437 dest? <input id="XSS" name="dest" type="text" size="80" value="/index_contents.html">
9438 <details><summary>Form Auto. Filling</summary>
9439 <style>
9440 .xinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9441   display:inline !important; font-size:10pt !important; padding:1px !important;
9442 }
9443 </style>
9444 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9445 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
9446 Location: <input id="xxserv" class="xinput" type="text" value="https://192.168.10.1/" size="80">
9447 Username: <input id="xxuser" class="xinput" name="user" type="text" autocomplete="on">
9448 Password: <input id="xxpass" class="xinput" name="pass" type="password" autocomplete="on">
9449 SessionId:<input id="xxsid" class="xinput" name="SESSION_ID" type="text" size="80">
9450 <input id="xxsum" class="xinput" type="submit" value="Submit"></form>
9451 </span>
9452 <script>
9453   function XXSetFormAction(){
9454     xform.setAttribute('action',xxserv.value);
9455   }
9456   xform.setAttribute('action',xxserv.value);
9457   xxserv.addEventListener('change',XXSetFormAction);
9458   //xxserv.value = location.href;
9459   </script>
9460 </details>
9461 </pre>
9462 
```

9463 /*

```

9464 <details id="BlinderTextClass"><summary>BlinderText</summary>
9465 <span class="shh-acc">
9466   <span id="BlinderTextEvent">
9467     // https://w3c.github.io/uievents/#event-type-keydown
9468     // 2020-09-21 class BlinderText - textarea element not to be readable
9469     // BlinderText attributes
9470     // bl_plainText - nulls
9471     // bl_hideChecksum - [false]
9472     // bl_showLength - [false]
9473     // bl_visible - [false]
9474     // data-bl_config - []
9475     // - max_length
9476     // - max_length
9477     // - acceptable charset in generete text
9478     //
9479     function BlinderChecksum(text){
9480       plain = text.bl_plainText;
9481       return strCRC32(plain,plain.length).toFixed(0);
9482     }
9483     function BlinderKeydown(ev){
9484       pass = ev.target
9485       if( ev.code == 'Enter' ){
9486         ev.preventDefault();
9487       }
9488       ev.stopPropagation()
9489     }
9490     function BlinderKeyup(ev){
9491       bind = ev.target
9492       if( ev.code == 'Backspace' ){
9493         bind.bl_plainText = bind.bl_plainText.slice(0,bind.bl_plainText.length-1)
9494       }else
9495       if( and(ev.code == 'KeyV', ev.ctrlKey) ){
9496         bind.bl_visible = !bind.bl_visible;
9497       }else
9498       if( and(ev.code == 'KeyL', ev.ctrlKey) ){
9499         bind.bl_showLength = !bind.bl_showLength;
9500       }else
9501       if( and(ev.code == 'KeyU', ev.ctrlKey) ){
9502         bind.bl_plainText = "";
9503       }else
9504       if( and(ev.code == 'KeyP', ev.ctrlKey) ){
9505         checksum = BlinderChecksum(bind);
9506         bind.bl_plainText = checksum; //toString(32);
9507       }else
9508       if( ev.code == 'Enter' ){
9509         ev.stopPropagation();
9510         ev.preventDefault();
9511         return;
9512       }
9513       if( ev.key.length == 1 ){
9514         console.log( 'KeyUp: '+ev.code+'/'+ev.key);
9515       }
9516     }
9517     if( ev.key.length == 1 ){
9518       bind.bl_plainText += ev.key;
9519     }else{
9520       bind.bl_plainText += ev.key;
9521     }
9522     leng = bind.bl_plainText.length;
9523     //console.log( 'KeyUp: '+ev.code+'/'+bind.bl_plainText);
9524     checksum = BlinderChecksum(bind) % 10; // show last one digit only
9525     visual = '';
9526     if( !bind.bl_hideChecksum || bind.bl_showLength ){
9527       visual += "[";
9528     }
9529     if( !bind.bl_hideChecksum ){
9530       visual += "#";
9531     }
9532     if( !bind.bl_hideChecksum ){
9533       visual += "#"+checksum.toString(10);
9534     }
9535     if( bind.bl_showLength ){
9536       visual += '/' + leng;
9537     }
9538     if( !bind.bl_hideChecksum || bind.bl_showLength ){
9539       visual += "]";
9540     }
9541     if( bind.bl_visible ){
9542       visual += bind.bl_plainText;
9543     }else{
9544       visual += '*' .repeat(leng);
9545     }
9546     bind.value = visual;
9547   }
9548   function BlinderKeyup(ev){
9549     BlinderKeyup(ev);
9550     ev.stopPropagation();
9551   }
9552 // https://w3c.github.io/uievents/#keyboardevent
9553 // https://w3c.github.io/uievents/#uievent
9554 // https://w3c.github.io/spec.whatwg.org/#event
9555 function BlinderTextEvent(){
9556   ev = event;
9557   bind = ev.target;
9558 }
```

```

9558 console.log('Event '+ev.type+'#'+blind.nodeName+'#'+blind.id)
9559 if( ev.type == 'keyup' ){
9560     Blinderkeyup(ev);
9561 }else{
9562     if( ev.type == 'keydown' ){
9563         Blinderkeydown(ev);
9564     }else{
9565         console.log("thru-event "+ev.type+'#'+blind.nodeName+'#'+blind.id)
9566     }
9567 }
9568 // textarea hidden id="BlinderTextClassDef" class="textField"
9569 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9570 // spellcheck="false"></textarea>
9571 //<txa id="bl_text" gj_pass1
9572 // class="textField BlinderText"
9573 // placeholder="PassWord"
9574 // onkeydown="BlinderTextEvent()"
9575 // onkeyup="BlinderTextEvent()"
9576 // spellcheck="false"></textarea>
9577 function SetupBlinderText(parent,txa,phold){
9578     if( txa == null ){
9579         txa = document.createElement('textarea');
9580         //txa.id = id;
9581     }
9582     txa.setAttribute('class','textField BlinderText');
9583     txa.setAttribute('placeholder','');
9584     txa.setAttribute('onkeydown','BlinderTextEvent()');
9585     txa.setAttribute('onkeyup','BlinderTextEvent()');
9586     txa.setAttribute('spellcheck','false');
9587     //txa.setAttribute('bl_plainText','false');
9588     txa.bl_plainText = '';
9589     //parent.appendChild(txn);
9590 }
9591 function DestroyBlinderText(txn){
9592     txa.removeAttribute('class');
9593     txa.removeAttribute('placeholder');
9594     txa.removeAttribute('onkeydown');
9595     txa.removeAttribute('onkeyup');
9596     txa.removeAttribute('spellcheck');
9597     txa.bl_plainText = '';
9598 }
9599 //
9600 // visible textarea like Username
9601 //
9602 function VisibleTextEvent(){
9603     if( event.code == 'Enter' ){
9604         if( event.target.NoEnter ){
9605             event.preventDefault();
9606         }
9607     }
9608     event.stopPropagation();
9609 }
9610 function SetupVisibleText(parent,txa,phold){
9611     if( false ){
9612         txa.setAttribute('class','textField VisibleText');
9613     }else{
9614         newclass = txa.getAttribute('class');
9615         if( and(newclass != null, newclass != '') ){
9616             newclass += ' ';
9617         }
9618         newclass += 'VisibleText';
9619         txa.setAttribute('class',newclass);
9620     }
9621     //console.log('SetupVisibleText class="'+txa.class");
9622     txa.setAttribute('placeholder',phold);
9623     txa.setAttribute('onkeydown','VisibleTextEvent()');
9624     txa.setAttribute('onkeyup','VisibleTextEvent()');
9625     txa.setAttribute('spellcheck','false');
9626     cols = txa.getAttribute('cols');
9627     if( cols != null ){
9628         txa.style.width = '580px';
9629         //console.log('VisibleText#'+txa.id+' cols='+cols)
9630     }else{
9631         //console.log('VisibleText#'+txa.id+' NO cols')
9632     }
9633     rows = txa.getAttribute('rows');
9634     if( rows != null ){
9635         txa.style.height = '30px';
9636         txa.style.resize = 'both';
9637         txa.NoEnter = false;
9638     }else{
9639         txa.NoEnter = true;
9640     }
9641 }
9642 function DestroyVisibleText(txn){
9643     txa.removeAttribute('class');
9644     txa.removeAttribute('placeholder');
9645     txa.removeAttribute('onkeydown');
9646     txa.removeAttribute('onkeyup');
9647     txa.removeAttribute('spellcheck');
9648     cols = txa.removeAttribute('cols');
9649 }
9650 </span>
9651 <script>
9652 jg = document.getElementById('BlinderTextScript');
9653 jg.innerHTML="";
9654 //jg.outerHTML = "";
9655 </script>
9656
9657 <span><-- end of class="gsh-src" -->
9658 </details>
9659 </span>
9660 </>
9661 /*
9662 <script id="GJLinkScript">
9663     function gj_log(text){
9664         return strCRC32(text, text.length) + 0x10000;
9665     }
9666     function gj_addlog(e,msg){
9667         now = (new Date().getTime() / 1000).toFixed(3);
9668         tsdp = '['+now'];
9669         e.value += tsdp + msg;
9670         e.scrollTop = e.scrollHeight;
9671     }
9672     function gj_addlog_ci(msg){
9673         ws0.log.value += '(console.log)' + msg + '\n';
9674     }
9675     ws0.log.value += '(console.error)' + msg + '\n';
9676 }
9677 var GJ_Channel = null;
9678 var GJ_Log = null;
9679 var gjx; // the global variable
9680 function GJ_Join(){
9681     target = gj_join;
9682     if( target.value == 'Leave' ){
9683         GJ_Channel.close();
9684         GJ_Channel = null;
9685         target.value = 'Join';
9686         return;
9687     }
9688     var ws0;
9689     var ws0_log;
9690     sav_console_log = console.error;
9691     console.error = gj_addlog_ci;
9692     ws0 = new WebSocket(gj_serv.innerHTML);
9693     console.error = sav_console_log;
9694
9695     GJ_Channel = ws0;
9696     ws0_log = document.getElementById('ws0_log');
9697     GJ_Log = ws0_log;
9698
9699     now = (new Date().getTime() / 1000).toFixed(3);
9700     const wstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
9701     cst = wstats[ws0.readyState];
9702     gj_addlog(ws0_log, stat 'ws0.readyState'+cst+') : GJ Linked\n');
9703
9704     ws0.addEventListener('error', function(event){
9705         gj_addlog(ws0_log,'stat error : transport error?\n');
9706     });
9707     ws0.addEventListener('open', function(event{
9708         GJLinkView.style.zIndex=10000;
9709         //console.log('GJLinkView.id#'+GJLinkView.id+'.zIndex=' + GJLinkView.style.zIndex);
9710         date1 = new Date().getTime();
9711         date2 = (date1 / 1000).toFixed(3);
9712         seed = date1.toString(16);
9713
9714         // user name and key
9715         user = document.getElementById('gj_user').value;
9716         if( user.length == 0 ){
9717             gj_user.value = 'nemo';
9718             user = 'nemo';
9719

```

```

9720     }
9721     key1 = document.getElementById('gj_ukey').bl_plaintext;
9722     ukey = gjkey_hash(seed+user+key1).toString(16);
9723 
9724     // session name and key
9725     chan = document.getElementById('gj_chan').value;
9726     if( chan.length == 0 ){
9727         gj_chan.value = 'main';
9728         chan = 'main';
9729     }
9730     key2 = document.getElementById('gj_ckey').bl_plaintext;
9731     ckey = gjkey_hash(seed+chan+key2).toString(16);
9732 
9733     msg = date2+' JOIN '+user+'|'+chan+' '+ukey+' : '+ckey;
9734     gj_addlog(ws0_log,'send '+msg+'\n');
9735     ws0.send(msg);
9736 
9737     target.value = 'Leave';
9738     //console.log(['|'+date2+'| '+target.id+' '+target.value+'\n']);
9739     //gj_addlog(ws0_log,'label '+target.value+'\n');
9740 
9741     ws0.addEventListener('message', function(event){
9742         now = (new Date()).getTime() / 1000).toFixed(3);
9743         msg = event.data;
9744         if( false ){
9745             gj_addlog(ws0_log, 'recv '+msg+'\n');
9746         }
9747         argv = msg.split(' ');
9748         timestamp = argv[0];
9749         argv.shift();
9750         if( argv[0] == 'reload' ){
9751             location.reload();
9752         }
9753         gjcmd = argv[0];
9754         otstamp = '';
9755         if( gjcmd == 'CAST' ){ // from reflector
9756             otstamp = argv[0];
9757             argv.shift(); // original time stamp
9758             argv[0] = argv[0];
9759             argv.shift(); // original from
9760         }
9761         argv.shift(); // command
9762         cmd = argv[0];
9763         argv.shift(); // from/to
9764         cmd1 = argv[0]; // xxxx command
9765         argv.shift(); // xxxx
9766 
9767         if( false ){
9768             gj_addlog(ws0_log, '-'
9769                 +' '+timestamp+'|'
9770                 +' '+gjcmd+'|'
9771                 +' '+from+'|'
9772                 +' '+cmd1+'|'
9773                 +' '+argv[0] );
9774     }
9775     if( cmd1 == 'auth' ){
9776         // doing authorization required
9777     }
9778     if( cmd1 == 'echo' ){
9779         now = (new Date().getTime() / 1000).toFixed(3);
9780         msg = now+'|'+RESP+'|'+argv.join('|');
9781         gj_addlog(ws0_log,'send '+msg+'\n');
9782         ws0.send(msg);
9783     }
9784     if( cmd1 == 'eval' ){
9785         argv.shift();
9786         js = argv.join(' ');
9787         ret = eval(js); // <----- eval()
9788         gj_addlog(ws0_log,'eval '+'js+' = '+ret+'\n');
9789         now = (new Date().getTime() / 1000).toFixed(3);
9790         msg = now+'|'+RESP+'|'+ret;
9791         ws0.send(msg);
9792         gj_addlog(ws0_log,'send '+msg+'\n');
9793     }
9794     if( cmd1 == 'DRAW' ){
9795         if( false ){
9796             gj_addlog(ws0_log,'DRAW '+argv[0]+'\n');
9797         }
9798         Pointillism_RemoteDraw(argv[0]);
9799     });
980     ws0.addEventListener('close', function(event){
981         if( GJ_Channel == null ){
982             gj_addlog(ws0_log, 'stat OK : GJ UnLinked\n');
983             return;
984         }
985         GJ_Channel.close();
986         GJ_Channel = null;
987         target.value = 'Join';
988         gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
989     });
990 }
991 function GJ_BcastMessageUserPass(user,chan,msgbody){
992     now = (new Date().getTime() / 1000).toFixed(3);
993     msg = now+'|BCAST|'+user+'|'+chan+'|'+msgbody;
994     if( false ){
995         gj_addlog(GJ_Log,'send '+msg+'\n');
996     }
997     GJ_Channel.send(msg);
998 }
999 function GJ_BcastMessage(msgbody){
1000     if( GJ_Channel == null ){
1001         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
1002         return;
1003     }
1004     //target = event.target;
1005     user = document.getElementById('gj_user').value;
1006     chan = document.getElementById('gj_chan').value;
1007     GJ_BcastMessageUserPass(user,chan,msgbody);
1008 }
1009 function GJ_SendMessageUserPass(user,chan,msgbody){
1010     now = (new Date().getTime() / 1000).toFixed(3);
1011     msg = now+'|'+user+'|'+chan+'|'+msgbody;
1012     gj_addlog(GJ_Log,'send '+msg+'\n');
1013     GJ_Channel.send(msg);
1014 }
1015 function GJ_SendMessage(msgbody){
1016     if( GJ_Channel == null ){
1017         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
1018         return;
1019     }
1020     //target = event.target;
1021     user = document.getElementById('gj_user').value;
1022     chan = document.getElementById('gj_chan').value;
1023     GJ_SendMessageUserPass(user,chan,msgbody);
1024 }
1025 function GJ_Send(){
1026     msgbody = gj_sendText.value;
1027     GJ_SendMessage(msgbody);
1028 }
1029 
```

<-- ----- GJLINK ----- >

```

1030 //<span id="GJLinkGolang">
1031 //<details id="GshWebsocket"><summary>Golang / JavaScript Link</summary>
1032 //<div class="list-item"><a href="#">Install</a><!-- { -->
1033 // 2020-0920 created
1034 // <a href="https://pkgs.go.dev/golang.org/x/net/websocket">WS</a>
1035 // <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
1036 // INSTALL: go get golang.org/x/net/websocket
1037 // INST: sudo apt-get install git (if git is not installed yet)
1038 // install "golang.org/x/net/websocket"
1039 const gshws_origin = "http://localhost:9999";
1040 const gshws_server = "localhost:9999";
1041 const gshws_port = 9999;
1042 const gshws_path = "gjlink";
1043 const gshws_protocol = "ws";
1044 const gshws_maxMessageSize = 65536;
1045 const gshws_maxFrameSize = 65536;
1046 const gshws_maxHeaderSize = 1024;
1047 func fmtString(fmts string, params ...interface{}) (string{
1048     return fmt.Sprintf(fmts, params...)
1049 })
1050 
```

```

9882 func GSHWS_MARK(what string)(string){
9883     now := time.Now()
9884     us := fmt.Sprintf("%06d",now.Nanosecond() / 1000)
9885     mark := ""
9886     if( !AtConsoleLineTop ){
9887         mark += "\n"
9888         AtConsoleLineTop = true
9889     }
9890     mark += "["+now.Format(time.Stamp)+"."+us+"] -GJ- "+ what + " "
9891     return mark
9892 }
9893 func gchk(what string,err error){
9894     if( err != nil ){
9895         panic(GSHWS_MARK(what)+err.Error())
9896     }
9897 }
9898 func glog(what string, fmts string, params ...interface{}{
9899     fmt.Println(GSHWS_MARK(what))
9900     fmt.Printf(fmts+"\n",params...)
9901 }
9902 )
9903 var WSV = []*websocket.Conn{
9904 func json(argv []string){
9905     if len(argv) <= 1 {
9906         fmt.Printf("-iJ %v [-m] command arguments\n",argv[0])
9907         return
9908     }
9909     argv = argv[1:]
9910     if( len(WSV) == 0 ){
9911         fmt.Printf("--Ej-- No link now\n")
9912         return
9913     }
9914     if( 1 < len(WSV) ){
9915         fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
9916     }
9917     multicast := false // should be filtered with regexp
9918     if( 0 < len(argv) && argv[0] == "-m" ){
9919         multicast = true
9920         argv = argv[1:]
9921     }
9922     args := strings.Join(argv, " ")
9923
9924     now := time.Now()
9925     msec := now.UnixNano() / 1000000;
9926     timestamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
9927     msg := fmt.Sprintf("%v SEND gshell* %v",timestamp,args)
9928
9929     if( multicast ){
9930         for i,ws := range WSV {
9931             wn,werr := ws.Write([]byte(msg))
9932             if( werr != nil ){
9933                 fmt.Println("[*v] wn=%v, werr=%v\n",i,wn,werr)
9934             }
9935             glog("SQ",fmt.Sprintf("(%v) %v",wn,msg))
9936         }
9937     }else{
9938         i := 0
9939         ws := WSV[i]
9940         wn,werr := ws.Write([]byte(msg))
9941         if( werr != nil ){
9942             fmt.Println("[*v] wn=%v, werr=%v\n",i,wn,werr)
9943         }
9944         glog("SQ",fmt.Sprintf("(%v) %v",wn,msg))
9945     }
9946 }
9947 func ws_broadcast(msg string)(wn int,werr error){
9948     for i,ws := range WSV {
9949         ws,werr = ws.Write([]byte(msg))
9950         if( werr != nil ){
9951             fmt.Printf("[*v] wn=%v, werr=%v\n",i,wn,werr)
9952         }
9953         glog("SQ",fmt.Sprintf("(%v) %v",wn,msg))
9954     }
9955     return wn,werr;
9956 }
9957 func serv(ws *websocket.Conn) {
9958     WSV = append(WSV,ws)
9959     //fmt.Println("v")
9960     glog("CO","accepted connections(%v)",len(WSV))
9961     //remoteAddr := ws.RemoteAddr
9962     //fmt.Printf("-- accepted %v\n",remoteAddr)
9963     //fmt.Printf("-- accepted %v\n",ws.Config())
9964     //fmt.Println("-- accepted %v",ws.Config().Header)
9965     //fmt.Println("-- accepted tv // %v",ws.servi)
9966     //fmt.Println("-- accepted tv // %v",ws.servi)
9967
9968     var reqb = make([]byte,GSHWS_MSGSIZE)
9969     for {
9970         rn, rerr := ws.Read(reqb)
9971         if( rerr != nil || rn < 0 ){
9972             glog("SQ",fmt.Sprintf("(%v) %v",rn,rerr))
9973             break
9974         }
9975         req := string(reqb[:rn])
9976         glog("SQ",fmt.Sprintf("(%v) %v",rn,req))
9977
9978         argv := strings.Split(req, " ")
9979         argv = argv[1:]
9980         if( 0 < len(argv) ){
9981             if( argv[0] == "RESP" ){
9982                 // should forward to the destination
9983                 continue;
9984             }
9985         }
9986         now := time.Now()
9987         msec := now.UnixNano() / 1000000;
9988         timestamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
9989         res := fmt.Sprintf("%v "+`CAST`+" %v",timestamp,req)
9990
9991         wn := 0;
9992         werr := error(nil);
9993         if( 0 < len(argv) && argv[0] == "BCAST" ){
9994             res, werr = ws.Broadcast(res);
9995         }else{
9996             wn, werr = ws.Write([]byte(res))
9997         }
9998         gchk("SE",werr)
9999         glog("SR",fmt.Sprintf("(%v) %v",wn,string(res)))
10000
10001     glog("SR","WS response finish")
10002
10003     wsv := []websocket.Conn{
10004         wsx := 0
10005         for i,v := range WSV {
10006             if( v != ws ){
10007                 wsx = i
10008                 wsv = append(wsv,v)
10009             }
10010         }
10011     }
10012     WSV = wsv
10013     //glog("CO","closed %v",ws)
10014     glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
10015 }
10016 func url ::= [scheme://]host[:port]/[path]
10017 func decom_URL(url string){
10018 }
10019 func full_wsURL(){
10020 }
10021 func gj_server(argv []string) {
10022     gjserver := gshell_url
10023     gshell := gshell_over
10024     gjpath := gshws_path
10025     gjscheme := "ws"
10026
10027     /cmd := argv[0]
10028     argv = argv[1:]
10029     if( 1 < len(argv) ){
10030         serv := argv[0]
10031         if( 0 < strings.Index(serv,"/") ){
10032             schemes := strings.Split(serv,"/")
10033             gjscheme = schemes[0]
10034             serv = schemes[1]
10035         }
10036         if( 0 < strings.Index(serv,".") ){
10037             pathv := strings.Split(serv,".")
10038             serv = pathv[0]
10039             gjpath = pathv[1]
10040         }
10041         servv := strings.Split(serv,":")
10042         host := "localhost"
10043         port := 9999

```

```

10044     if( servv[0] != "" ){
10045         host = servv[0]
10046     }
10047     if( len(servv) == 2 ){
10048         fmt.Sscanf(servv[1],"%d",&port)
10049     }
10050     //glog("LC","hostport=%v (%v : %v",servv,host,port)
10051     gjport = fmt.Sprintf("%v:%v",host,port)
10052     gjserver = gjscheme + ":" + gjport + "/" + gjpath
10053 }
10054 glog("LS",fmtstring("listening at %v",gjserver))
10055 http.Handle("/"+gjpath,websocket.Handler(serv1))
10056 err := error(nil)
10057 if( gjscheme == "ws" ){
10058     //err = https://golang.org/pkg/net/http/#ListenAndServeTLS
10059     //err = http.ListenAndServeTLS(gjport,nil)
10060 }else{
10061     err = http.ListenAndServe(gjport,nil)
10062 }
10063 gchk("LE",err)
10064 }
10065
10066 func gj_client(argv []string) {
10067     glog("CS",fmtstring("connecting to %v",gshws_url))
10068     ws,err := websocket.Dial(gshws_url,"",gshws_origin)
10069     gchk("C",err)
10070
10071     var resb = make([]byte, GSHWS_MSGSIZE)
10072     for qi := 0; qi < 3; qi++ {
10073         req := fmtstring("Hello, Gshell! (%v)",qi)
10074         wn, werr := ws.Write([]byte(req))
10075         gjlog("W",fmtstring("(%) %v",wn,req))
10076         gchk("WF",werr)
10077         rn, rerr := ws.Read(resb)
10078         gchk("RF",rerr)
10079         glog("RM",fmtstring("(%) %v",rn,string(resb)))
10080     }
10081     glog("CF","WS request finish")
10082 }
10083 //</span><-- end of class="gsh-src" > -->
10084 //</details></span>
10085 /*
10086 <details id="GJLink_Section"><summary>GJ Link</summary>
10087 <span id="GJLinkView" class="GJLinkView">
10088 <p>
10089 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
10090 </p>
10091 </span>
10092 <span id="GJLink_1">
10093 <div id="GJLink_ServerSet"></div>
10094 <div>
10095     <div>
10096         <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10097         <span id="GJLink_Account"></span>
10098     </div>
10099     <div>
10100         <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10101         <span id="GJLink_SendArea"></span>
10102     </div>
10103     <div id="ws0_log_container"></div>
10104     <span>
10105         <script>
10106             function setupGJLinkArea(){
10107                 GJLink_ServerSet.innerHTML = '<+'+span id="gj_serv_label"'+
10108                     + ' class="textField labelText">Server: <+'+span'+'
10109                     + '<+'+span id="gj_serv" class="textField textURL" contenteditable>'+'</span'+';
10110
10111                 GJLink_Account.innerHTML = '<+'+textarea id="gj_user" class="textField">'+'</textarea'+'
10112                     + '<+'+textarea id="gj_ukey" class="textField">'+'</textarea'+'
10113                     + '<+'+textarea id="gj_chan" class="textField">'+'</textarea'+'
10114                     + '<+'+textarea id="gj_ckey" class="textField">'+'</textarea'+';
10115
10116                 GJLink_SendArea.innerHTML =
10117                     '<+'+textarea id="gj_sendText" class="textField MssgText" cols=60 rows=2>'+'</textarea>';
10118
10119                 ws0_log_container.innerHTML = '<+'+textarea id="ws0_log" class="ws0_log"'+
10120                     + ' cols=100 rows=10 spellcheck="false">'+'</textarea>';
10121
10122             }
10123             function clearGJLinkArea(){
10124                 GJLink_ServerSet.innerHTML = "";
10125                 GJLink_Account.innerHTML = "";
10126                 GJLink_SendArea.innerHTML = "";
10127                 ws0_log_container.innerHTML = "";
10128             }
10129         </script>
10130     </span>
10131     <script>
10132         function SetupGJLink(){
10133             setupGJLinkArea();
10134             SetupLabelText(GJLink_1,gj_serv,'GJLinkServer');
10135             SetupLabelText(GJLink_1,gj_user,'UserName');
10136             SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
10137             SetupBlinderText(GJLink_1,gj_chan,'ChannelName');
10138             SetupBlinderText(GJLink_1,gj_ckey,'ChannelKey');
10139             SetupVisibleText(GJLink_1,gj_sendtext,'Message');
10140             gj_serv.innerHTML = 'ws://localhost:9999/gjlink1';
10141
10142         }
10143         function GJLink_init(){
10144             SetupGJLink();
10145         }
10146         function iselem(eid){
10147             return document.getElementById(eid);
10148         }
10149         function DestroyGJLink1(){
10150             clearGJLinkArea();
10151             if( !iselem('gj_user') ){
10152                 return;
10153             }
10154             if( gj_serv_label.parentNode != gj_user ){
10155                 return;
10156             }
10157             if( gj_serv_label.parentNode != gj_user ){
10158                 return;
10159             }
10160             if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10161             if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10162             if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10163             if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10164             if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10165             if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
10166             if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10167             if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10168
10169         DestroyGJLink = DestroyGJLink1;
10170     </script>
10171 </details>
10172 */
10173 /*
10174 */
10175 <style>
10176 .GJDigest {
10177     display:none;
10178 }
10179 </style>
10180 <script id="HTMLCodeview-script">
10181     function showNodeAsHtmlSource(xtxa,code,prefix,postfix,sign){
10182         txa = document.createElement('textarea');
10183         txa.id = otxa.id;
10184         txa.setAttribute('class','HTMLCodeviewText');
10185         txa.setAttribute('spellcheck','false');
10186         txa.setattribute('placeholder',replaceChild(txaxta));
10187         //txa.value = code.innerHTML;
10188         //txa.innerHTML = code.innerHTML + postfix;
10189         txa.innerHTML = prefix + code.outerHTML + postfix;
10190         if( sign != null ){
10191             text = txa.value;
10192             tlen = txaxta.value.length;
10193             digest = strCRC32(text,tlen) + ' ' + tlen
10194             + ' ' + code.id + ' (' + DateShort() + ')';
10195             //alert("digest: "+digest);
10196             console.log(digest);
10197             txa.innerHTML += //<+'+span class="GJDigest">' + digest + '</' + span + '\n';
10198         }
10199         txa.style.display = "block";
10200         txa.style.width = "100%";
10201         txa.style.height = "300px";
10202     }
10203     function showHTMLCode(xtxa,code,prefix,postfix,sign){
10204         if( event.target.value == "ShowCode" ){
10205             showNodeAsHtmlSource(xtxa,code,prefix,postfix,sign);

```

```

10206     event.target.value = 'HideCode';
10207   }else{
10208     otxa.style.display = "none";
10209     event.target.value = 'ShowCode';
10210   }
10211 }
10212 function showNodeAsHtmlSource(otxa,code){
10213   showNodeAsHtmlSourceX(otxa,code,'','');
10214 }
10215 function showHtmlCodeValue(otxa,code){
10216   if( event.target.value == 'ShowCode' ){
10217     showNodeAsHtmlSource(otxa,code);
10218     event.target.value = 'HideCode';
10219   }else{
10220     otxa.style.display = "none";
10221     event.target.value = 'ShowCode';
10222   }
10223 }
10224</script>
10225<style id="HTMLCodeview-style">
10226 .HTMLCodeviewText {
10227   font-size:10pt;
10228   font-family:Courier New;
10229   white-space:pre;
10230 }
10231 .HTMLCodeviewButton {
10232   padding:2pt !important;
10233   line-height:1.1 !important;
10234   border:1px solid black !important;
10235   font-size:11pt !important;
10236   font-weight: normal !important;
10237   font-family: Georgia !important;
10238   border-radius:3px !important;
10239   color:#ddd; background-color:#228 !important;
10240 }
10241</style>
10242</pre>
10243</div>
10244<details><summary>Live HTML Snapshot</summary>
10245<pre>----- Live HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
10246<div class="GshMenu">
10247<span class="GshMenu1" onclick="html_edit();">>Edit</span>
10248<span class="GshMenu1" onclick="html_save();">>Save</span>
10249<span class="GshMenu1" onclick="html_load();">>Load</span>
10250<span class="GshMenu1" onclick="html_ver();">>Vers</span>
10251</div>
10252</div>
10253<div>
10254<input class="HTMLCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()"/>
10255<input class="HTMLCodeViewButton" type="button" value="HideCode" onclick="showLiveHTMLCode()"/>
10256</div>
10257</div>
10258<script id="LiveHTMLScript">
10259   function showLiveHTMLCode(){
10260     showHTMLCode(LiveHTML_Codeview,LiveHTML);
10261   }
10262   var _editable = false;
10263   var SuppressGJShell = false;
10264   function ToggleEditMode(){
10265     _editable = !_editable;
10266     if(_editable){
10267       SuppressGJShell = SuppressGJShell;
10268       GshMenuEdit.innerHTML = 'Lock';
10269       GshMenuEdit.style.color = 'rgba(255,0,0,1)';
10270       GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10271     }else{
10272       SuppressGJShell = savSuppressGJShell;
10273       GshMenuEdit.innerHTML = 'Edit';
10274       GshMenuEdit.style.color = 'rgba(16,160,16,1)';
10275       GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10276     }
10277   }
10278   function html_edit(){
10279     ToggleEditMode();
10280   }
10281   function html_save(){
10282     ToggleEditMode();
10283   }
10284
10285   // Live HTML (DOM) Snapshot onto browser's localStorage
10286   // 2020-0923 SatoxITS
10287   var rhRoot = gsh // -- Element-ID, should be selectable
10288   const snappedHTML = 'SnappedTMD'; // Item-ID of the HTML data in localStorage
10289   // -- should be a [map] of URL
10290   const htVersionTag = 'VersionTag'; // VersionTag Element-ID in the HTML (in DOM)
10291   function showVersion(note,w,u,t){
10292     w.alert(note+' : '+v+'\n'+u+'\n'+t);
10293     +--- URL: ' + u + '\n'
10294     +--- Time: ' + t + ' (' + DateLong0(t*1000) + ')'
10295   }
10296 }
10297
10298 function html_save(){
10299   u = document.URL;
10300   t = new Date().getTime() / 1000;
10301   v = '<!--spad id=' + htVersionTag + ' data-url=' + u + ' data-time=' + t + '-->';
10302   h = document.createElement('div');
10303   h.setAttribute('data-url',u);
10304   localStorage.setItem(snappedHTML,h);
10305   showVersion('Saved',window,v,u,t);
10306 }
10307
10308 function html_load(){
10309   h = localStorage.getItem(snappedHTML);
10310   if(h == null){
10311     alert('No snapshot taken yet');
10312     return;
10313   }
10314   w = window.open('','','');
10315   d = w.document;
10316   d.write(h);
10317   w.focus();
10318   html_ver('Loaded',w,d);
10319 }
10320
10321 function html_ver(note,w,u,t){
10322   if( (v = d.getElementById(htVersionTag)) != null ){
10323     h = v.outerHTML;
10324     u = v.getAttribute('data-url');
10325     t = v.getAttribute('data-time');
10326   }else{
10327     h = 'No version info. in the page';
10328     u = '';
10329     t = 0;
10330   }
10331   showVersion(note,w,u,t);
10332 }
10333
10334 function html_ver0(){
10335   v = 'LiveHTML';
10336   html_ver("Version",window,document);
10337 }
10338</script>
10339<!-- LiveHTML -->
10340</span>
10341</details>
10342<details><summary>Event sharing</summary>
10343<pre>----- Event sharing // 2020-0925 SatoxITS { -->
10344<div id="iftestTemplate" class="iftest" hidden=""><style>
10345   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10346   <div id="iftestBody" style="display:none;" onclick="frameclick()"><script>
10347     function dodacd(tx){}
10348     document.body.append(tx);
10349     window.scrollTo(0,100000);
10350   </div>
10351   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10352   <function frameClick(){}
10353   xy = '(x'+event.x+' y'+event.y+')';
10354   dodacd('Get Click on #' + event.target.id + ' ' + xy + '\n');
10355   dodacd('Get Click on #' + id.value + ' ' + xy + '\n');
10356   window.scrollTo(0,100000);
10357   window.parent.postMessage('onClick: ' + xy + ',');
10358   </function>
10359   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10360   <function frameMousemove(){}
10361   if( false )
10362     document.body.append('Mousemove on #' + event.target.id + ' '
10363     + 'x' + event.x + ' y' + event.y + '\n');
10364   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10365   window.scrollTo(0,100000);
10366   window.parent.postMessage('Hi!', '*');
10367   peerwin.postMessage('Hi!', '*');
10368
10369   </function>
10370   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10371   <function frameMouseover(){}
10372   if( false )
10373     document.body.append('Mouseover on #' + event.target.id + ' '
10374     + 'x' + event.x + ' y' + event.y + '\n');
10375   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10376   window.scrollTo(0,100000);
10377   window.parent.postMessage('Hi!', '*');
10378   peerwin.postMessage('Hi!', '*');
10379
10380   </function>
10381   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10382   <function frameMouseout(){}
10383   if( false )
10384     document.body.append('Mouseout on #' + event.target.id + ' '
10385     + 'x' + event.x + ' y' + event.y + '\n');
10386   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10387   window.scrollTo(0,100000);
10388   window.parent.postMessage('Hi!', '*');
10389   peerwin.postMessage('Hi!', '*');
10390
10391   </function>
10392   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10393   <function frameFocus(){}
10394   if( false )
10395     document.body.append('Focus on #' + event.target.id + ' '
10396     + 'x' + event.x + ' y' + event.y + '\n');
10397   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10398   window.scrollTo(0,100000);
10399   window.parent.postMessage('Hi!', '*');
10400   peerwin.postMessage('Hi!', '*');
10401
10402   </function>
10403   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10404   <function frameBlur(){}
10405   if( false )
10406     document.body.append('Blur on #' + event.target.id + ' '
10407     + 'x' + event.x + ' y' + event.y + '\n');
10408   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10409   window.scrollTo(0,100000);
10410   window.parent.postMessage('Hi!', '*');
10411   peerwin.postMessage('Hi!', '*');
10412
10413   </function>
10414   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10415   <function frameError(){}
10416   if( false )
10417     document.body.append('Error on #' + event.target.id + ' '
10418     + 'x' + event.x + ' y' + event.y + '\n');
10419   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10420   window.scrollTo(0,100000);
10421   window.parent.postMessage('Hi!', '*');
10422   peerwin.postMessage('Hi!', '*');
10423
10424   </function>
10425   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10426   <function frameLoad(){}
10427   if( false )
10428     document.body.append('Load on #' + event.target.id + ' '
10429     + 'x' + event.x + ' y' + event.y + '\n');
10430   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10431   window.scrollTo(0,100000);
10432   window.parent.postMessage('Hi!', '*');
10433   peerwin.postMessage('Hi!', '*');
10434
10435   </function>
10436   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10437   <function frameUnload(){}
10438   if( false )
10439     document.body.append('Unload on #' + event.target.id + ' '
10440     + 'x' + event.x + ' y' + event.y + '\n');
10441   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10442   window.scrollTo(0,100000);
10443   window.parent.postMessage('Hi!', '*');
10444   peerwin.postMessage('Hi!', '*');
10445
10446   </function>
10447   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10448   <function frameResize(){}
10449   if( false )
10450     document.body.append('Resize on #' + event.target.id + ' '
10451     + 'x' + event.x + ' y' + event.y + '\n');
10452   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10453   window.scrollTo(0,100000);
10454   window.parent.postMessage('Hi!', '*');
10455   peerwin.postMessage('Hi!', '*');
10456
10457   </function>
10458   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10459   <function frameScroll(){}
10460   if( false )
10461     document.body.append('Scroll on #' + event.target.id + ' '
10462     + 'x' + event.x + ' y' + event.y + '\n');
10463   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10464   window.scrollTo(0,100000);
10465   window.parent.postMessage('Hi!', '*');
10466   peerwin.postMessage('Hi!', '*');
10467
10468   </function>
10469   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10470   <function frameFocusin(){}
10471   if( false )
10472     document.body.append('Focusin on #' + event.target.id + ' '
10473     + 'x' + event.x + ' y' + event.y + '\n');
10474   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10475   window.scrollTo(0,100000);
10476   window.parent.postMessage('Hi!', '*');
10477   peerwin.postMessage('Hi!', '*');
10478
10479   </function>
10480   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10481   <function frameFocusout(){}
10482   if( false )
10483     document.body.append('Focusout on #' + event.target.id + ' '
10484     + 'x' + event.x + ' y' + event.y + '\n');
10485   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10486   window.scrollTo(0,100000);
10487   window.parent.postMessage('Hi!', '*');
10488   peerwin.postMessage('Hi!', '*');
10489
10490   </function>
10491   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10492   <function frameDblclick(){}
10493   if( false )
10494     document.body.append('Dblclick on #' + event.target.id + ' '
10495     + 'x' + event.x + ' y' + event.y + '\n');
10496   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10497   window.scrollTo(0,100000);
10498   window.parent.postMessage('Hi!', '*');
10499   peerwin.postMessage('Hi!', '*');
10500
10501   </function>
10502   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10503   <function frameDragstart(){}
10504   if( false )
10505     document.body.append('Dragstart on #' + event.target.id + ' '
10506     + 'x' + event.x + ' y' + event.y + '\n');
10507   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10508   window.scrollTo(0,100000);
10509   window.parent.postMessage('Hi!', '*');
10510   peerwin.postMessage('Hi!', '*');
10511
10512   </function>
10513   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10514   <function frameDragend(){}
10515   if( false )
10516     document.body.append('Dragend on #' + event.target.id + ' '
10517     + 'x' + event.x + ' y' + event.y + '\n');
10518   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10519   window.scrollTo(0,100000);
10520   window.parent.postMessage('Hi!', '*');
10521   peerwin.postMessage('Hi!', '*');
10522
10523   </function>
10524   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10525   <function frameDragover(){}
10526   if( false )
10527     document.body.append('Dragover on #' + event.target.id + ' '
10528     + 'x' + event.x + ' y' + event.y + '\n');
10529   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10530   window.scrollTo(0,100000);
10531   window.parent.postMessage('Hi!', '*');
10532   peerwin.postMessage('Hi!', '*');
10533
10534   </function>
10535   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10536   <function frameDragleave(){}
10537   if( false )
10538     document.body.append('Dragleave on #' + event.target.id + ' '
10539     + 'x' + event.x + ' y' + event.y + '\n');
10540   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10541   window.scrollTo(0,100000);
10542   window.parent.postMessage('Hi!', '*');
10543   peerwin.postMessage('Hi!', '*');
10544
10545   </function>
10546   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10547   <function frameDragenter(){}
10548   if( false )
10549     document.body.append('Dragenter on #' + event.target.id + ' '
10550     + 'x' + event.x + ' y' + event.y + '\n');
10551   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10552   window.scrollTo(0,100000);
10553   window.parent.postMessage('Hi!', '*');
10554   peerwin.postMessage('Hi!', '*');
10555
10556   </function>
10557   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10558   <function frameDrag(){}
10559   if( false )
10560     document.body.append('Drag on #' + event.target.id + ' '
10561     + 'x' + event.x + ' y' + event.y + '\n');
10562   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10563   window.scrollTo(0,100000);
10564   window.parent.postMessage('Hi!', '*');
10565   peerwin.postMessage('Hi!', '*');
10566
10567   </function>
10568   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10569   <function frameDragcancel(){}
10570   if( false )
10571     document.body.append('Dragcancel on #' + event.target.id + ' '
10572     + 'x' + event.x + ' y' + event.y + '\n');
10573   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10574   window.scrollTo(0,100000);
10575   window.parent.postMessage('Hi!', '*');
10576   peerwin.postMessage('Hi!', '*');
10577
10578   </function>
10579   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10580   <function frameError(){}
10581   if( false )
10582     document.body.append('Error on #' + event.target.id + ' '
10583     + 'x' + event.x + ' y' + event.y + '\n');
10584   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10585   window.scrollTo(0,100000);
10586   window.parent.postMessage('Hi!', '*');
10587   peerwin.postMessage('Hi!', '*');
10588
10589   </function>
10590   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10591   <function frameError(){}
10592   if( false )
10593     document.body.append('Error on #' + event.target.id + ' '
10594     + 'x' + event.x + ' y' + event.y + '\n');
10595   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10596   window.scrollTo(0,100000);
10597   window.parent.postMessage('Hi!', '*');
10598   peerwin.postMessage('Hi!', '*');
10599
10600   </function>
10601   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10602   <function frameError(){}
10603   if( false )
10604     document.body.append('Error on #' + event.target.id + ' '
10605     + 'x' + event.x + ' y' + event.y + '\n');
10606   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10607   window.scrollTo(0,100000);
10608   window.parent.postMessage('Hi!', '*');
10609   peerwin.postMessage('Hi!', '*');
10610
10611   </function>
10612   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10613   <function frameError(){}
10614   if( false )
10615     document.body.append('Error on #' + event.target.id + ' '
10616     + 'x' + event.x + ' y' + event.y + '\n');
10617   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10618   window.scrollTo(0,100000);
10619   window.parent.postMessage('Hi!', '*');
10620   peerwin.postMessage('Hi!', '*');
10621
10622   </function>
10623   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10624   <function frameError(){}
10625   if( false )
10626     document.body.append('Error on #' + event.target.id + ' '
10627     + 'x' + event.x + ' y' + event.y + '\n');
10628   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10629   window.scrollTo(0,100000);
10630   window.parent.postMessage('Hi!', '*');
10631   peerwin.postMessage('Hi!', '*');
10632
10633   </function>
10634   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10635   <function frameError(){}
10636   if( false )
10637     document.body.append('Error on #' + event.target.id + ' '
10638     + 'x' + event.x + ' y' + event.y + '\n');
10639   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10640   window.scrollTo(0,100000);
10641   window.parent.postMessage('Hi!', '*');
10642   peerwin.postMessage('Hi!', '*');
10643
10644   </function>
10645   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10646   <function frameError(){}
10647   if( false )
10648     document.body.append('Error on #' + event.target.id + ' '
10649     + 'x' + event.x + ' y' + event.y + '\n');
10650   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10651   window.scrollTo(0,100000);
10652   window.parent.postMessage('Hi!', '*');
10653   peerwin.postMessage('Hi!', '*');
10654
10655   </function>
10656   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10657   <function frameError(){}
10658   if( false )
10659     document.body.append('Error on #' + event.target.id + ' '
10660     + 'x' + event.x + ' y' + event.y + '\n');
10661   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10662   window.scrollTo(0,100000);
10663   window.parent.postMessage('Hi!', '*');
10664   peerwin.postMessage('Hi!', '*');
10665
10666   </function>
10667   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10668   <function frameError(){}
10669   if( false )
10670     document.body.append('Error on #' + event.target.id + ' '
10671     + 'x' + event.x + ' y' + event.y + '\n');
10672   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10673   window.scrollTo(0,100000);
10674   window.parent.postMessage('Hi!', '*');
10675   peerwin.postMessage('Hi!', '*');
10676
10677   </function>
10678   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10679   <function frameError(){}
10680   if( false )
10681     document.body.append('Error on #' + event.target.id + ' '
10682     + 'x' + event.x + ' y' + event.y + '\n');
10683   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10684   window.scrollTo(0,100000);
10685   window.parent.postMessage('Hi!', '*');
10686   peerwin.postMessage('Hi!', '*');
10687
10688   </function>
10689   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10690   <function frameError(){}
10691   if( false )
10692     document.body.append('Error on #' + event.target.id + ' '
10693     + 'x' + event.x + ' y' + event.y + '\n');
10694   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10695   window.scrollTo(0,100000);
10696   window.parent.postMessage('Hi!', '*');
10697   peerwin.postMessage('Hi!', '*');
10698
10699   </function>
10700   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10701   <function frameError(){}
10702   if( false )
10703     document.body.append('Error on #' + event.target.id + ' '
10704     + 'x' + event.x + ' y' + event.y + '\n');
10705   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10706   window.scrollTo(0,100000);
10707   window.parent.postMessage('Hi!', '*');
10708   peerwin.postMessage('Hi!', '*');
10709
10710   </function>
10711   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10712   <function frameError(){}
10713   if( false )
10714     document.body.append('Error on #' + event.target.id + ' '
10715     + 'x' + event.x + ' y' + event.y + '\n');
10716   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10717   window.scrollTo(0,100000);
10718   window.parent.postMessage('Hi!', '*');
10719   peerwin.postMessage('Hi!', '*');
10720
10721   </function>
10722   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10723   <function frameError(){}
10724   if( false )
10725     document.body.append('Error on #' + event.target.id + ' '
10726     + 'x' + event.x + ' y' + event.y + '\n');
10727   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10728   window.scrollTo(0,100000);
10729   window.parent.postMessage('Hi!', '*');
10730   peerwin.postMessage('Hi!', '*');
10731
10732   </function>
10733   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10734   <function frameError(){}
10735   if( false )
10736     document.body.append('Error on #' + event.target.id + ' '
10737     + 'x' + event.x + ' y' + event.y + '\n');
10738   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10739   window.scrollTo(0,100000);
10740   window.parent.postMessage('Hi!', '*');
10741   peerwin.postMessage('Hi!', '*');
10742
10743   </function>
10744   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10745   <function frameError(){}
10746   if( false )
10747     document.body.append('Error on #' + event.target.id + ' '
10748     + 'x' + event.x + ' y' + event.y + '\n');
10749   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10750   window.scrollTo(0,100000);
10751   window.parent.postMessage('Hi!', '*');
10752   peerwin.postMessage('Hi!', '*');
10753
10754   </function>
10755   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10756   <function frameError(){}
10757   if( false )
10758     document.body.append('Error on #' + event.target.id + ' '
10759     + 'x' + event.x + ' y' + event.y + '\n');
10760   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10761   window.scrollTo(0,100000);
10762   window.parent.postMessage('Hi!', '*');
10763   peerwin.postMessage('Hi!', '*');
10764
10765   </function>
10766   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10767   <function frameError(){}
10768   if( false )
10769     document.body.append('Error on #' + event.target.id + ' '
10770     + 'x' + event.x + ' y' + event.y + '\n');
10771   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10772   window.scrollTo(0,100000);
10773   window.parent.postMessage('Hi!', '*');
10774   peerwin.postMessage('Hi!', '*');
10775
10776   </function>
10777   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10778   <function frameError(){}
10779   if( false )
10780     document.body.append('Error on #' + event.target.id + ' '
10781     + 'x' + event.x + ' y' + event.y + '\n');
10782   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10783   window.scrollTo(0,100000);
10784   window.parent.postMessage('Hi!', '*');
10785   peerwin.postMessage('Hi!', '*');
10786
10787   </function>
10788   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10789   <function frameError(){}
10790   if( false )
10791     document.body.append('Error on #' + event.target.id + ' '
10792     + 'x' + event.x + ' y' + event.y + '\n');
10793   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10794   window.scrollTo(0,100000);
10795   window.parent.postMessage('Hi!', '*');
10796   peerwin.postMessage('Hi!', '*');
10797
10798   </function>
10799   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10800   <function frameError(){}
10801   if( false )
10802     document.body.append('Error on #' + event.target.id + ' '
10803     + 'x' + event.x + ' y' + event.y + '\n');
10804   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10805   window.scrollTo(0,100000);
10806   window.parent.postMessage('Hi!', '*');
10807   peerwin.postMessage('Hi!', '*');
10808
10809   </function>
10810   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10811   <function frameError(){}
10812   if( false )
10813     document.body.append('Error on #' + event.target.id + ' '
10814     + 'x' + event.x + ' y' + event.y + '\n');
10815   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10816   window.scrollTo(0,100000);
10817   window.parent.postMessage('Hi!', '*');
10818   peerwin.postMessage('Hi!', '*');
10819
10820   </function>
10821   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10822   <function frameError(){}
10823   if( false )
10824     document.body.append('Error on #' + event.target.id + ' '
10825     + 'x' + event.x + ' y' + event.y + '\n');
10826   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10827   window.scrollTo(0,100000);
10828   window.parent.postMessage('Hi!', '*');
10829   peerwin.postMessage('Hi!', '*');
10830
10831   </function>
10832   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10833   <function frameError(){}
10834   if( false )
10835     document.body.append('Error on #' + event.target.id + ' '
10836     + 'x' + event.x + ' y' + event.y + '\n');
10837   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10838   window.scrollTo(0,100000);
10839   window.parent.postMessage('Hi!', '*');
10840   peerwin.postMessage('Hi!', '*');
10841
10842   </function>
10843   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10844   <function frameError(){}
10845   if( false )
10846     document.body.append('Error on #' + event.target.id + ' '
10847     + 'x' + event.x + ' y' + event.y + '\n');
10848   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10849   window.scrollTo(0,100000);
10850   window.parent.postMessage('Hi!', '*');
10851   peerwin.postMessage('Hi!', '*');
10852
10853   </function>
10854   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10855   <function frameError(){}
10856   if( false )
10857     document.body.append('Error on #' + event.target.id + ' '
10858     + 'x' + event.x + ' y' + event.y + '\n');
10859   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10860   window.scrollTo(0,100000);
10861   window.parent.postMessage('Hi!', '*');
10862   peerwin.postMessage('Hi!', '*');
10863
10864   </function>
10865   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10866   <function frameError(){}
10867   if( false )
10868     document.body.append('Error on #' + event.target.id + ' '
10869     + 'x' + event.x + ' y' + event.y + '\n');
10870   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10871   window.scrollTo(0,100000);
10872   window.parent.postMessage('Hi!', '*');
10873   peerwin.postMessage('Hi!', '*');
10874
10875   </function>
10876   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10877   <function frameError(){}
10878   if( false )
10879     document.body.append('Error on #' + event.target.id + ' '
10880     + 'x' + event.x + ' y' + event.y + '\n');
10881   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10882   window.scrollTo(0,100000);
10883   window.parent.postMessage('Hi!', '*');
10884   peerwin.postMessage('Hi!', '*');
10885
10886   </function>
10887   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10888   <function frameError(){}
10889   if( false )
10890     document.body.append('Error on #' + event.target.id + ' '
10891     + 'x' + event.x + ' y' + event.y + '\n');
10892   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10893   window.scrollTo(0,100000);
10894   window.parent.postMessage('Hi!', '*');
10895   peerwin.postMessage('Hi!', '*');
10896
10897   </function>
10898   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10899   <function frameError(){}
10900   if( false )
10901     document.body.append('Error on #' + event.target.id + ' '
10902     + 'x' + event.x + ' y' + event.y + '\n');
10903   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10904   window.scrollTo(0,100000);
10905   window.parent.postMessage('Hi!', '*');
10906   peerwin.postMessage('Hi!', '*');
10907
10908   </function>
10909   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10910   <function frameError(){}
10911   if( false )
10912     document.body.append('Error on #' + event.target.id + ' '
10913     + 'x' + event.x + ' y' + event.y + '\n');
10914   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10915   window.scrollTo(0,100000);
10916   window.parent.postMessage('Hi!', '*');
10917   peerwin.postMessage('Hi!', '*');
10918
10919   </function>
10920   <!-- #iftestBody:color:#f22;font-family:Georgia;font-size:10pt; -->
10921   <function frameError(){}
10922   if( false )
10923     document.body.append('Error on #' + event.target.id + ' '
10924     + 'x' + event.x + ' y' + event.y + '\n');
10925   peerwin.window.frame.peerSendToPeer('#' + peerWin + ' ' + '\n');
10926   window.scrollTo(0,100000);
10927   window.parent.postMessage('Hi!', '*');
10928   peerwin.postMessage('Hi!', '*');
10929
10930   </function>
10931   <!-- #iftestBody:color:#f22;font-family
```

```

10368 }
10369 function frameKeyDown(){
10370     msg = 'Got Keydown: #' + Fid.value+ ', ('+event.code+')';
10371     window.parent.postMessage(msg,'*');
10372 }
10373 function frameOnMessage(){
10374     docadd(Message + event.data + '\n');
10375     window.scrollTo(0,100000);
10376 }
10377 if( document.getElementById('Fid') ){
10378     frameBody.id = Fid.value;
10379     h = '<'+style+'<';
10380     h += 'font-size:10pt;white-space:pre-wrap;';
10381     h += 'font-family:Courier New;';
10382     h += 'I am #' + Fid.value+'\n';
10383     document.write(h);
10384     window.addEventListener('click',frameClick);
10385     window.addEventListener('keydown',frameKeyDown);
10386     window.addEventListener('message',frameOnMessage);
10387     window.addEventListener('mousemove',frameMouseMove);
10388     window.parent.postMessage('Hi parent, I am #' + Fid.value,'*');
10389 }
10390 </script></span></div>
10391 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
10392 <h2>Inter-window communication</h2>
10393 <note>
10394 frame0 >>> frame1 and frame2<br>
10395 frame1 >>> frame0 and frame2<br>
10396 frame2 >>> frame0 and frame1<br>
10397 </note>
10398 <div id="iframe-test">
10399 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
10400 <iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
10401 <iframe id="iframe1" title="iframe1" class="iframeTest" style=""></iframe>
10402 <iframe id="iframe2" title="iframe2" class="iframeTest" style=""></iframe>
10403 </div>
10404 <script id="f10-test-script">
10405     function InterFrameComm_init(){
10406         SetupFrames0();
10407         SetupFrames12();
10408     }
10409     function SetFrameSrcDoc(dst,src){
10410         if( true ){
10411             dst.contentWindow.document.write(src);
10412             // this makes browser waits close, and crash if accumulated !?
10413             // so it should be closed after write
10414             dst.contentWindow.document.close();
10415         }else{
10416             to be erased before source dump
10417             // but should be set for live snapshot
10418             dst.srcDoc = src;
10419         }
10420     }
10421     function setupFrames0(){
10422         body = iframe0.contentWindow.document.body;
10423         iframe0.style.width = "755px";
10424         //iframe0.host.innerHTML = "Message exchange at iframes' host:\n";
10425         window.addEventListener('message',messageFromChild);
10426         if0 = '';
10427         if0 += '<+'+pre style="font-family:Courier New;">';
10428         if0 += '<input id="Fid" value="iframe0">';
10429         if0 += iftestTemplate.innerHTML;
10430         setFrameSrcDoc(iframe0,if0);
10431         function clickOnChild(){
10432             console.log(clickOn '#'+this.id);
10433         }
10434         function moveOnChild(){
10435             console.log(moveOn '#'+this.id);
10436         }
10437         iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10438         iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10439     }
10440     function setupFrames12(){
10441         if1 = '<input id="Fid" value="iframe1">';
10442         if1 += iftestTemplate.innerHTML;
10443         setFrameSrcDoc(iframe1,if1);
10444         //iframe1.name = "iframe1"; // this seems break contentWindow
10445         if2 = '<input id="Fid" value="iframe2">';
10446         if2 += iftestTemplate.innerHTML;
10447         setFrameSrcDoc(iframe2,if2);
10448         iframe1.addEventListener('message',messageFromChild);
10449         //iframe1.addEventListener('mouseover',moveOnChild);
10450         iframe2.addEventListener('message',messageFromChild);
10451         iframe2.addEventListener('mouseover',moveOnChild);
10452         iframe1.contentWindow.postMessage(['parent','Hi iframe1 -- from parent.','*']);
10453         iframe2.contentWindow.postMessage(['parent','Your peer is '+iframe2.contentWindow,'*']);
10454         //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow,'*');
10455     }
10456     function messageFromChild(){
10457         from = null;
10458         forw = null;
10459         if( event.source == iframe0.contentWindow ){
10460             from = '[iframe0]';
10461             forw = '[iframe12]';
10462         }else{
10463             if( event.source == iframe1.contentWindow ){
10464                 from = '[iframe1]';
10465                 forw = '[iframe2]';
10466             }else{
10467                 if( event.source == iframe2.contentWindow ){
10468                     from = '[iframe2]';
10469                     forw = '[iframe1]';
10470                 }else{
10471                     if( event.source == 'Message [unknown]' ||
10472                         + ' origin' + event.origin
10473                         + ' data' + event.data
10474                         //+ ' from' + event.source
10475                         ;
10476                     msglog1 = from + event.data + ' -- '
10477                         + ' from' + event.source
10478                         + ' origin' + event.origin
10479                         + ' name' + event.source.name
10480                         //+ ' ports' + event.ports
10481                         //+ ' evide' + event.lastEventId
10482                         + '\n'
10483                     );
10484                     if( true ){
10485                         if( forw == 'iframe1' || forw == 'iframe12' ){
10486                             iframe1.contentWindow.postMessage(from+event.data);
10487                         }
10488                         if( forw == 'iframe2' || forw == 'iframe12' ){
10489                             iframe2.contentWindow.postMessage(from+event.data);
10490                         }
10491                     }
10492                     txtadd0(msglog1);
10493                 }
10494             }
10495         }
10496     }
10497     function txtadd0(txt){
10498         iframe0.contentWindow.document.body.append(txt);
10499         iframe0.contentWindow.scrollTo(0,100000);
10500     }
10501     function es_ShowSelf(){
10502         iframe1.setAttribute('src',document.URL);
10503         iframe2.setAttribute('src',document.URL);
10504     }
10505     </script>
10506     <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
10507     <input class="CodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
10508     <span id="EventSharingCodeview"></span>
10509     <script id="EventSharingScript">
10510     function es_showHtmlCode(){
10511         showWtMLCode(EventSharingCodeview,EventSharingCodeSpan);
10512     }
10513     function es_ShareSelf(){
10514         iframe1.setAttribute('src',document.URL);
10515     }
10516 </script>
10517 <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
10518 <input class="CodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
10519 <span id="EventSharingCodeview"></span>
10520 <script id="EventSharingScript">
10521     function es_showHtmlCode(){
10522         showWtMLCode(EventSharingCodeview,EventSharingCodeSpan);
10523     }
10524     destroyEventSharingCodeview = function(){
10525         /EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
10526         EventSharingCodeview.innerHTML = '';
10527         iframe0.style = "";;
10528         /iframe0.srcDoc = "erased";
10529     }

```

```

10530 //iframe1.srcdoc = "erased";
10531 //iframe2.srcdoc = "erased";
10532 }
10533 </script>
10534 <!-- EventSharing -->
10535 </span>
10536 </details>
10537 </div>
10538 /*
10539 <!-- ----- "GShell Inside" Notification { -->
10540 <script id="script-gshell-inside">
10541 var notices = 0;
10542 function noticeGShellInside(){
10543   var ver = '';
10544   if( ver = document.getElementById('GshVersion') ){
10545     ver.innerHTML;
10546   }
10547   console.log('GShell Inside (^~)/*'+ver);
10548   notices += 1;
10549   if( 2 < notices ){
10550     document.removeEventListener('mousemove',noticeGShellInside);
10551   }
10552 }
10553 }
10554 document.addEventListener('mousemove',noticeGShellInside);
10555 noticedShellInside();
10556
10557 const FooterName = 'GshFooter';
10558 function DestroyFooter(){
10559   if( (footer = document.getElementById(FooterName)) != null ){
10560     //footer.parentNode.removeChild(footer);
10561     empty = document.createElement('div');
10562     empty.id = 'GshFooter0';
10563     footer.parentNode.replaceChild(empty,footer);
10564   }
10565 }
10566 function showFooter(){
10567   footer = document.createElement('div');
10568   footer.id = FooterName;
10569   footer.style.backgroundImage = "url("+ITSMoreQR+"")";
10570   GshFooter0.parentNode.appendChild(footer);
10571   GshFooter0.parentNode.replaceChild(footer,GshFooter0);
10572 }
10573 </script>
10574 <!-- } -->
10575
10576 <!-- border:20px inset #888;
10577 <!--
10578 <span id="VirtualDesktopCodeSpan">
10579 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
10580 <summary>Virtual Desktop</summary>
10581 <!--
10582 <details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
10583 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
10584 <style>
10585 .NVirtualSpace {
10586   z-index:0;
10587   width:1280px !important; xheight:720px !important;
10588   width:5120px; height:2880px;
10589   border:1px solid #fff;
10590   background-color:rgba(32,32,160,0.8);
10591   background-image:url("WD-WallPaper03.png");
10592   background-size:100% 100%;
10593   color:#22a; font-family:Georgia;font-size:10pt;
10594   xoverflow:scroll;
10595 }
10596 .VirtualGrid {
10597   z-index:0;
10598   position:absolute;
10599   width:800px; height:500px;
10600   border:1px solid #fff;
10601   color:#22a; font-size:12px;
10602   font-family:Georgia, Courier New;
10603   text-align:right;
10604   vertical-align:middle;
10605   font-size:20px;
10606   text-shadow:4px 4px #ccf;
10607 }
10608 .WD_Gridscroll {
10609   z-index:100000;
10610   background-color:rgba(200,200,200,0.1);
10611 }
10612 .VirtualDesktop {
10613   z-index:0;
10614   position:relative;
10615   resize:both !important;
10616   overflow:scroll;
10617   display:block;
10618   margin:120px !important; min-height:60px !important;
10619   width:800px;
10620   height:500px;
10621   border:10px inset #228;
10622   border-width:30px; border-radius:20px;
10623   background-image:url("WD-WallPaper03.png");
10624   background-size:100% 100%;
10625   color:#22a; font-family:Georgia;font-size:10pt;
10626 }
10627 .comment {
10628   // overflow=scroll seems to bound childrens' view in the element span
10629   // specifying overflow seems fix the position of the element
10630 }
10631 .VirtualBrowserSpan {
10632   z-index:10;
10633   xxxborder:0.5px dashed #fff !important;
10634   border-color:rgba(255,255,255,0.5) !important;
10635   position:relative;
10636   left:10px;
10637   top:10px;
10638   display:block;
10639   resize:both !important;
10640   width:540px;
10641   height:320px;
10642   border:4px solid #fff; min-height:120px !important;
10643   max-width:5120px !important; max-height:2880px !important;
10644   background-color:rgba(255,200,255,0.1);
10645   xoverflow:scroll;
10646 }
10647 .xVirtualBrowserLocationBar:focus {
10648   color:#f00;
10649   background-color:rgba(255,128,128,0.2);
10650 }
10651 .xVirtualBrowserLocationBar:active {
10652   color:#f00;
10653   background-color:rgba(128,255,128,0.2);
10654 }
10655 a.VirtualBrowserLocation {
10656   color:#ccc !important;
10657   text-decoration:none !important;
10658 }
10659 a.VirtualBrowserLocation:hover {
10660   color:#fff !important;
10661   text-decoration:underline;
10662 }
10663 .VirtualBrowserLocationBar {
10664   position:absolute;
10665   z-index:100000;
10666   display:block;
10667   width:400px;
10668   height:20px;
10669   padding-left:2px;
10670   line-height:1.1;
10671   vertical-align:middle;
10672   font-size:14px;
10673   color:#fff;
10674   background-color:rgba(128,128,128,0.2);
10675   font-family:Georgia;
10676 }
10677 .VirtualBrowserCommandBar {
10678   position:absolute;
10679   z-index:200000;
10680   xxxdisplay:inline;
10681   display:inline-block;
10682   width:60px;
10683   height:20px;
10684   line-height:1.1;
10685   vertical-align:middle;
10686   font-size:14px;
10687   color:#f00;
10688   background-color:rgba(128,128,128,0.1);
10689   font-family:Georgia;
10690   text-align:left;
10691   left:404px;

```

```

10692}
10693.VirtualBrowserFrame {
10694    position: relative;
10695    position: absolute;
10696    display: inline;
10697    display: block;
10698    z-index: 10;
10699    resize: none !important;
10700    width: 480px; height: 240px;
10701    min-width: 60px; min-height: 30px;
10702    max-width: 5120px; max-height: 2880px;
10703    border-radius: 6px;
10704    background-color:rgba(255,255,255,0.9);
10705    border-top:2px solid;
10706    border-left:4px solid;
10707    border-bottom:10px solid;
10708}
10709.WinFavicon {
10710    width:16px;
10711    height:16px;
10712    margin:0;
10713    margin-right:3px;
10714    vertical-align:middle;
10715    background-color:rgba(255,255,255,1.0);
10716}
10717.VirtualDesktopMenuBar {
10718    position: absolute;
10719    color:#fff;
10720    font-size:7pt;
10721    text-align:right;
10722    padding-right:4px;
10723    background-color:rgba(128,128,128,0.7);
10724}
10725.VirtualDesktopCalendar {
10726    color:#fff;
10727    font-size:22pt;
10728    text-align:right;
10729    padding-right:4px;
10730    background-color:rgba(255,255,255,0.2);
10731}
10732.xxxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
10733    display:inline !important; font-size:10pt !important; padding:1px !important;
10734}
10735.WD_Config {
10736    display:inline !important;
10737    padding:2px !important;
10738    font-size:10pt !important;
10739    width:60pt !important;
10740    height:12pt !important;
10741    line-height:1.0pt !important;
10742    height:15pt !important;
10743}
10744.WD_Button {
10745    display:inline !important;
10746    padding:1px !important;
10747    color:#fff !important;
10748    background-color:#28 !important;
10749    font-size:10pt !important;
10750    width:60pt !important;
10751    height:12pt !important;
10752    line-height:1.0pt !important;
10753    height:16pt !important;
10754    border:2px inset #4aa !important;
10755}
10756.WD_Href {
10757    display:inline !important;
10758    padding:2px !important;
10759    font-size:9pt !important;
10760    width:120pt !important;
10761    height:12pt !important;
10762    line-height:1.0pt !important;
10763    height:15pt !important;
10764}
10765}
10766.LiveHtmlCodeviewText {
10767    font-size:10pt;
10768    font-family:Courier New;
10769    white-space:pre;
10770}
10771}
10772.WD_Panel {
10773    x-index:100 !important;
10774    color:#000 !important;
10775    font-family:Courier New;
10776    width:800px !important;
10777    padding:4px !important;
10778    border:1px solid #888 !important;
10779    border-radius:6px !important;
10780    background-color:rgba(220,220,220,0.9) !important;
10781    font-size:9pt;
10782    font-family:Courier New;
10783}
10784.WD_Help {
10785    font-size:10pt !important;
10786    font-family:Courier New;
10787    line-height:1.2 !important;
10788    color:#000 !important;
10789    width:100% !important;
10790    background-color:rgba(240,240,255,0.8) !important;
10791}
10792}
10793.WB_Zoom {
10794}
10795</style>
10796<h2>CosmoScreen 0.0.8</h2>
10797<menu>
10798<span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
10799    <table border="1" grid-on/off><br>
10800    1 ... zoom in<br>
10801    o ... zoom out<br>
10802    s ... save current scope<br>
10803    r ... restore saved scope<br>
10804    p ...<br>
10805</table>
10806</span></menu>
10807<div class="WD_Panel" draggable="true">
10808    <p><!-- should be on the frame of the WD -->
10809    Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
10810    <input id="WD_Width_1" class="WD_Config" type="text" value="1440x900">
10811    <input id="WD_Height_1" class="WD_Config" type="text" value="1440x900">
10812    <a href="WD-WallPaler03.png" class="WD_Href" contenteditable="true">WD-WallPaler03.png</a>
10813<p>
10814    Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
10815    <input id="WS_1_Width" class="WD_Config" type="text" value="1440x900">
10816    <input id="WS_1_Height" class="WD_Config" type="text" value="1440x900">
10817</p>
10818<p>
10819    Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
10820    <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
10821    <input id="WD_Zoom_1_X" class="WD_Config" type="text" value="1440x900">
10822    <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
10823    <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
10824</p>
10825<p>
10826    Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
10827    <input id="WD_Left_1" class="WD_Config" type="text" value="0">
10828    Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
10829    shift+wheel for horizontal scroll
10830</p>
10831<p>
10832    Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
10833    <input id="WD_Zoom_1_ResizedScope" class="WD_Config" type="text" value="Scope0">
10834    <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
10835    <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
10836</p>
10837<p>
10838    Scopyy <input id="WD_Zoom_1_For" class="WD_Button" type="button" value="For">
10839    <input id="WD_Zoom_1_ForScope" class="WD_Config" type="text" value="Scope0">
10840    <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
10841    <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
10842</p>
10843<p>
10844    Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
10845    <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
10846    | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
10847    > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
10848</p>
10849<p>
10850    Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
10851</p>
10852<p>
10853    Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">

```

```

10854 "scroll" imprisons windows inside the display
10855 "</>";
10856 </div>
10857 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
10858 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
10859 <span id="CommoScreen 0.0.8-/"><span id="VirtualDesktop_1_Clock">></span>
10860 </div>
10861 <div id="VirtualDesktop_1_Calender" class="VirtualDesktopCalender" >00:00</div>
10862 <div align="right"><1>VirtualSpace 1.</1></div>
10863 <div id="VirtualDesktop_1_Content" class="VirtualSpace">
10864 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10865 <div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar" ></div>
10866 <span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar" >Reload</span>
10867 <div id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></div>
10868 </div>
10869 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10870 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar" ></div>
10871 <span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar" >Reload</span>
10872 <div id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></div>
10873 </div>
10874 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10875 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar" ></div>
10876 <span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar" >Reload</span>
10877 <div id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></div>
10878 </div>
10879 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
10880 </div>
10881 </div>
10882 </div>
10883 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
10884 </div>
10885 </div>
10886 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">>
10887 <span id="VirtualDesktopCodeview"></span>
10888 <script type="text/javascript">
10889 function vd_showHtmlCode(){
10890   codespan = document.getElementById('VirtualDesktopCodeSpan');
10891   showWmlCode(WmlCode(WmktopCodeview,codespan));
10892   VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
10893 }
10894 DestroyEventSharingCodeview = function(){
10895   VirtualDesktopCodeview.innerHTML = '';
10896 }
10897
10898 function wdlog(log){
10899   if( GJ_Channel != null ){
10900     GJ_SendMessage("WD "+log);
10901   }
10902   console.log(log);
10903 }
10904 var topMostWin = 10000;
10905 function onEnterWin(e){
10906   e.target.style.zIndex;
10907   cindex = e.target.style.zIndex;
10908   //if( oindex == '' ) oindex = 0;
10909   //t.saved.zIndex = oindex;
10910   t.style.zIndex = 10000;
10911   topMostWin = 1;
10912   t.savedzIndex = topMostWin;
10913   nindex = t.style.zIndex;
10914   window[e.target.id+'-'+t.id+'-'+oindex+'->'+nindex];
10915   nindex = t.style.zIndex;
10916   wdlog('Enter '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
10917   e.stopPropagation();
10918   e.preventDefault();
10919 }
10920 function onLeaveWin(e){ // can detect click on the thick border? t = e.target;
10921   cindex = t.style.zIndex;
10922   topMostWin += 1;
10923   t.style.zIndex = topMostWin;
10924   nindex = t.style.zIndex;
10925   window[e.target.id+'-'+t.id+'-'+oindex+'->'+nindex];
10926   //e.stopPropagation();
10927   //e.preventDefault();
10928 }
10929 function onLeaveWin(e){
10930   t = e.target;
10931   //t.saved.zIndex;
10932   //nindex = t.saved.zIndex;
10933   //t.style.zIndex = nindex;
10934   //wdlog('Leave '+e.target.id+' #'+e.target.id+'('+oindex+'->'+nindex+')');
10935   e.stopPropagation();
10936   e.preventDefault();
10937 }
10938
10939 var WinDragstartX; // event
10940 var WinDragstartY;
10941 var WinDragstartTX; // target
10942 var WinDragstartTY;
10943
10944 function onWinDragstart(e){
10945   WinDragstartX = e.x;
10946   WinDragstartY = e.y;
10947
10948   t = e.target;
10949
10950   //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
10951   //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
10952   if( t.style.left == '' ){
10953     WinDragstartTX = x0 = 0;
10954     t.style.left = '0px';
10955   }else{
10956     //WinDragstartTX = x0 = Number(t.style.left);
10957     WinDragstartTX = x0 = parseInt(t.style.left);
10958   }
10959   if( t.style.top == '' ){
10960     WinDragstartTY = y0 = 0;
10961     t.style.top = '0px';
10962   }else{
10963     //WinDragstartTY = y0 = Number(t.style.top);
10964     WinDragstartTY = y0 = parseInt(t.style.top);
10965   }
10966   if( true ){ // to be undo
10967     t.wasX = WinDragstartTX;
10968     t.wasY = WinDragstartTY;
10969   }
10970   wdlog('DraggTA #' + t.id +
10971   + ' event(' + e.x + ',' + e.y + ')'
10972   + ' position(' + t.style.position
10973   + ' style left,top(' + t.style.left + ',' + t.style.top + ')'
10974 );
10975   e.stopPropagation();
10976   //e.preventDefault();
10977   return true;
10978 }
10979 function onWinDragEvent(wh,e,set,dolog){
10980   t = e.target;
10981   dx = e.x - WinDragstartX;
10982   dy = e.y - WinDragstartY;
10983   nx = WinDragstartTX + dx;
10984   ny = WinDragstartTY + dy;
10985   log = 'Drag '+wh+' #' + t.id +
10986   + ' event(' + WinDragstartTX + ',' + WinDragstartTY + '
10987   + ' event(' + e.x + ',' + e.y + ')
10988   + ' diff(' + dx + ',' + dy + ')
10989   + ' nx ' + nx + ',' + ny + ')
10990   + ' (' + t.style.left + ',' + t.style.top + ')'
10991   + ' wasX(' + t.wasX + ',' + t.wasY + ')';
10992
10993   if( e.x != 0 || e.y != 0 ){
10994     if( t.style.left != t.wasX ){
10995       //t.style.left = nx + 'px'; // not effective
10996       //t.style.y = ny + 'px'; // not effective
10997       t.style.left = nx + 'px';
10998       t.style.top = ny + 'px';
10999       log += ' Set';
11000
11001     }else{
11002       log += ' NotSet';
11003       if( dolog ){
11004         log = '';
11005       }
11006     }
11007     log += ' What?'; // the type is event start?
11008     if( dolog ){
11009       log = '';
11010     }
11011   }
11012   if( and(dolog, log != '') ){
11013     wdlog(log);
11014   }
11015   if( true ){

```

```

11016     // should be propogated to parent in FireFox ?
11017     e.stopPropagation();
11018 }
11019 e.preventDefault();
11020 return false;
11021}
11022function onWinDrag(e){
11023     return onWinDragEvent('Ing',e,true,false);
11024}
11025function onWinDragend(e){
11026     return onWinDragEvent('End',e,false,true);
11027}
11028function onWinDragexit(e){
11029     return onWinDragEvent('Exit',e,false,true);
11030}
11031function onWinDragover(e){
11032     return onWinDragEvent('Over',e,false,true);
11033}
11034function onWinDragenter(e){
11035     return onWinDragEvent('Enter',e,false,true);
11036}
11037function onWinDragleave(e){
11038     return onWinDragEvent('Leave',e,false,true);
11039}
11040function onWinDragdrop(e){
11041     return onWinDragEvent('Drop',e,false,true);
11042}
11043function onFaviconChange(e){
11044     wdlog('---Favicon #' + e.target.id + ' href=' + e.details.href);
11045}
11046var savedSuppressGJShell = false;
11047function stopShell(e){
11048    //wdlog('Enter Gsh STOP');
11049    savedSuppressGJShell = SuppressGJShell;
11050    SuppressGJShell = true;
11051    e.stopPropagation();
11052    e.preventDefault();
11053}
11054function contShell(e){
11055    //wdlog('Leave Gsh STOP');
11056    SuppressGJShell = savedSuppressGJShell;
11057    e.stopPropagation();
11058    e.preventDefault();
11059}
11060function WD_onkeydown(e){
11061    keycode = e.code;
11062    console.log('Keydown #' + e.target.id + ' ' + keycode);
11063    if( keycode == 'KeyC' ){
11064        WD_SetGrid( WD_Grid_1 );
11065    }else{
11066        if( keycode == 'KeyI' ){
11067            WD_dozoomIN();
11068        }else{
11069            if( keycode == 'KeyO' ){
11070                WD_docomoOUT();
11071            }else{
11072                if( keycode == 'KeyR' ){
11073                    WD_RestoreScope(null);
11074                }else{
11075                    if( keycode == 'KeyS' ){
11076                        WD_SaveScope(null);
11077                    }
11078                }
11079            }
11080        }
11081    }
11082    function WD_onkeyup(e){
11083        e.stopPropagation();
11084        e.preventDefault();
11085    }
11086    function WD_Eventstartup(){
11087        VirtualDesktop_1.addEventListener('keydown', e => { WD_onkeydown(e); });
11088        VirtualDesktop_1.Content.addEventListener('keydown', e => { WD_onkeydown(e); });
11089        WD_Help_1.addEventListener('keydown', e => { WD_onkeydown(e); });
11090        WD_Help_1.addEventListener('keyup', e => { WD_onkeyup(e); });
11091    }
11092    function settleIn(s, l, cmd, f, u, w, h, x, y, c, scale){
11093        function VirtualBrowserCommand(e, s, l, cmd, f){
11094            command = cmd.innerHTML;
11095            if( command == "Reload" ){
11096                href_id = e.target.href_id;
11097                d = document.getElementById(href_id);
11098                wdlog('href_tag#' + href_id + '\n elem#' + href_id + '\n href=' + d.getAttribute('href'));
11099                url = d.getAttribute('href');
11100                wdlog('---- Load href_tag#' + href_id + '\n elem#' + href_id + '\n href=' + url);
11101                wdlog('---- Load target #' + f.id) + ' with url=' + url;
11102                f.src = url;
11103            }else{
11104                alert('unknown command' + command + ' ' + e.target.id + ', ' + l.id + ', ' + f.id);
11105            }
11106        }
11107    }
11108    function onKeyDown(e){
11109        if( e.code == 'Enter' ){
11110            e.stopPropagation();
11111            e.preventDefault();
11112        }
11113    }
11114    function onKeyUp(e){
11115        if( e.code == 'Enter' ){
11116            e.stopPropagation();
11117            e.preventDefault();
11118            // should reload immediately ?
11119        }
11120    }
11121    if( false ){
11122        wdlog('start settle VirtualBrowser url=' + u + '\n'
11123            + 'id=' + s.id + '\n'
11124            + 'width=' + s.style.width + '\n'
11125            + 'height=' + s.style.height
11126        );
11127    }
11128    // very important for WordPress ???
11129    s.style.width = f.style.width = 50; // for WordPress ...??
11130    s.style.height = f.style.height = 271; // for Wordpress ...??
11131    if( false ){
11132        wdlog('midway settle VirtualBrowser url=' + u + '\n'
11133            + 'id=' + s.id + '\n'
11134            + 'width=' + s.style.width + '\n'
11135            + 'height=' + s.style.height
11136        );
11137    }
11138    f.width = 502; // for WordPress ...??
11139    s.height = 272; // for Wordpress ...??
11140    if( false ){
11141        wdlog('midway-2 settle Virtualbrowser url=' + u + '\n'
11142            + 'id=' + s.id + '\n'
11143            + 'span-width' + s.width + '\n'
11144            + 'span-height' + s.height
11145        );
11146    }
11147    }
11148    s.style.width = w + 'px';
11149    s.style.height = h + 'px';
11150    f.style.width = w + 'px';
11151    f.style.height = h + 'px';
11152    //t.style.setProperty('-webkit-transform','scale('+scale+')');
11153    f.style.setProperty('transform','scale('+scale+')');
11154
11155    //wdlog('x1= ' + u + ' width=' + s.style.width + ' f=' + f.style.width);
11156    //wdlog('x2= ' + u + ' width=' + s.style.width + ' f=' + f.style.width);
11157    s.setAttribute('draggable','true');
11158    f.setAttribute('draggable','false'); // who necessary?
11159    l.setAttribute('draggable','false'); // who necessary?
11160    cmd.setAttribute('draggable','false'); // who necessary?
11161    s.addEventListener('dragstart', e => { onWinDrag(e); });
11162    s.addEventListener('drag', e => { onWinDrag(e); });
11163    s.addEventListener('dragend', e => { onWinDragend(e); });
11164    s.addEventListener('exit', e => { onWinDragexit(e); });
11165    s.addEventListener('dragover', e => { onWinDragover(e); });
11166    s.addEventListener('dragexit', e => { onWinDragexit(e); });
11167    s.addEventListener('dragenter', e => { onWinDragenter(e); });
11168    s.addEventListener('dragover', e => { onWinDragover(e); });
11169    s.addEventListener('dragleave', e => { onWinDragleave(e); });
11170    s.addEventListener('drop', e => { onWinDragdrop(e); });
11171
11172    s.addEventListener('mouseenter', e => { onEnterWin(e); });
11173    s.addEventListener('mouseleave', e => { onLeaveWin(e); });
11174
11175    if( false ){
11176        s.style.position = "absolute";
11177        s.style.x = x + 'px';
11178    }
11179}
11180
11181
11182
11183
11184
11185
11186
11187
11188
11189
11190
11191
11192
11193
11194
11195
11196
11197
11198
11199
11200
11201
11202
11203
11204
11205
11206
11207
11208
11209
11210
11211
11212
11213
11214
11215
11216
11217
11218
11219
11220
11221
11222
11223
11224
11225
11226
11227
11228
11229
11230
11231
11232
11233
11234
11235
11236
11237
11238
11239
11240
11241
11242
11243
11244
11245
11246
11247
11248
11249
11250
11251
11252
11253
11254
11255
11256
11257
11258
11259
11260
11261
11262
11263
11264
11265
11266
11267
11268
11269
11270
11271
11272
11273
11274
11275
11276
11277
11278
11279
11280
11281
11282
11283
11284
11285
11286
11287
11288
11289
11290
11291
11292
11293
11294
11295
11296
11297
11298
11299
11300
11301
11302
11303
11304
11305
11306
11307
11308
11309
11310
11311
11312
11313
11314
11315
11316
11317
11318
11319
11320
11321
11322
11323
11324
11325
11326
11327
11328
11329
11330
11331
11332
11333
11334
11335
11336
11337
11338
11339
11340
11341
11342
11343
11344
11345
11346
11347
11348
11349
11350
11351
11352
11353
11354
11355
11356
11357
11358
11359
11360
11361
11362
11363
11364
11365
11366
11367
11368
11369
11370
11371
11372
11373
11374
11375
11376
11377
11378
11379
11380
11381
11382
11383
11384
11385
11386
11387
11388
11389
11390
11391
11392
11393
11394
11395
11396
11397
11398
11399
11400
11401
11402
11403
11404
11405
11406
11407
11408
11409
11410
11411
11412
11413
11414
11415
11416
11417
11418
11419
11420
11421
11422
11423
11424
11425
11426
11427
11428
11429
11430
11431
11432
11433
11434
11435
11436
11437
11438
11439
11440
11441
11442
11443
11444
11445
11446
11447
11448
11449
11450
11451
11452
11453
11454
11455
11456
11457
11458
11459
11460
11461
11462
11463
11464
11465
11466
11467
11468
11469
11470
11471
11472
11473
11474
11475
11476
11477
11478
11479
11480
11481
11482
11483
11484
11485
11486
11487
11488
11489
11490
11491
11492
11493
11494
11495
11496
11497
11498
11499
11500
11501
11502
11503
11504
11505
11506
11507
11508
11509
11510
11511
11512
11513
11514
11515
11516
11517
11518
11519
11520
11521
11522
11523
11524
11525
11526
11527
11528
11529
11530
11531
11532
11533
11534
11535
11536
11537
11538
11539
11540
11541
11542
11543
11544
11545
11546
11547
11548
11549
11550
11551
11552
11553
11554
11555
11556
11557
11558
11559
11560
11561
11562
11563
11564
11565
11566
11567
11568
11569
11570
11571
11572
11573
11574
11575
11576
11577
11578
11579
11580
11581
11582
11583
11584
11585
11586
11587
11588
11589
11590
11591
11592
11593
11594
11595
11596
11597
11598
11599
11600
11601
11602
11603
11604
11605
11606
11607
11608
11609
11610
11611
11612
11613
11614
11615
11616
11617
11618
11619
11620
11621
11622
11623
11624
11625
11626
11627
11628
11629
11630
11631
11632
11633
11634
11635
11636
11637
11638
11639
11640
11641
11642
11643
11644
11645
11646
11647
11648
11649
11650
11651
11652
11653
11654
11655
11656
11657
11658
11659
11660
11661
11662
11663
11664
11665
11666
11667
11668
11669
11670
11671
11672
11673
11674
11675
11676
11677
11678
11679
11680
11681
11682
11683
11684
11685
11686
11687
11688
11689
11690
11691
11692
11693
11694
11695
11696
11697
11698
11699
11700
11701
11702
11703
11704
11705
11706
11707
11708
11709
11710
11711
11712
11713
11714
11715
11716
11717
11718
11719
11720
11721
11722
11723
11724
11725
11726
11727
11728
11729
11730
11731
11732
11733
11734
11735
11736
11737
11738
11739
11740
11741
11742
11743
11744
11745
11746
11747
11748
11749
11750
11751
11752
11753
11754
11755
11756
11757
11758
11759
11760
11761
11762
11763
11764
11765
11766
11767
11768
11769
11770
11771
11772
11773
11774
11775
11776
11777
11778
11779
11780
11781
11782
11783
11784
11785
11786
11787
11788
11789
11790
11791
11792
11793
11794
11795
11796
11797
11798
11799
11800
11801
11802
11803
11804
11805
11806
11807
11808
11809
11810
11811
11812
11813
11814
11815
11816
11817
11818
11819
11820
11821
11822
11823
11824
11825
11826
11827
11828
11829
11830
11831
11832
11833
11834
11835
11836
11837
11838
11839
11840
11841
11842
11843
11844
11845
11846
11847
11848
11849
11850
11851
11852
11853
11854
11855
11856
11857
11858
11859
11860
11861
11862
11863
11864
11865
11866
11867
11868
11869
11870
11871
11872
11873
11874
11875
11876
11877
11878
11879
11880
11881
11882
11883
11884
11885
11886
11887
11888
11889
11890
11891
11892
11893
11894
11895
11896
11897
11898
11899
11900
11901
11902
11903
11904
11905
11906
11907
11908
11909
11910
11911
11912
11913
11914
11915
11916
11917
11918
11919
11920
11921
11922
11923
11924
11925
11926
11927
11928
11929
11930
11931
11932
11933
11934
11935
11936
11937
11938
11939
11940
11941
11942
11943
11944
11945
11946
11947
11948
11949
11950
11951
11952
11953
11954
11955
11956
11957
11958
11959
11960
11961
11962
11963
11964
11965
11966
11967
11968
11969
11970
11971
11972
11973
11974
11975
11976
11977
11978
11979
11980
11981
11982
11983
11984
11985
11986
11987
11988
11989
11990
11991
11992
11993
11994
11995
11996
11997
11998
11999
11999
12000
12001
12002
12003
12004
12005
12006
12007
12008
12009
120010
120011
120012
120013
120014
120015
120016
120017
120018
120019
120020
120021
120022
120023
120024
120025
120026
120027
120028
120029
120030
120031
120032
120033
120034
120035
120036
120037
120038
120039
120040
120041
120042
120043
120044
120045
120046
120047
120048
120049
120050
120051
120052
120053
120054
120055
120056
120057
120058
120059
120060
120061
120062
120063
120064
120065
120066
120067
120068
120069
120070
120071
120072
120073
120074
120075
120076
120077
120078
120079
120080
120081
120082
120083
120084
120085
120086
120087
120088
120089
120090
120091
120092
120093
120094
120095
120096
120097
120098
120099
120099
120100
120101
120102
120103
120104
120105
120106
120107
120108
120109
120110
120111
120112
120113
120114
120115
120116
120117
120118
120119
120120
120121
120122
120123
120124
120125
120126
120127
120128
120129
120130
120131
120132
120133
120134
120135
120136
120137
120138
120139
120140
120141
120142
120143
120144
120145
120146
120147
120148
120149
120150
120151
120152
120153
120154
120155
120156
120157
120158
120159
120160
120161
120162
120163
120164
120165
120166
120167
120168
120169
120170
120171
120172
120173
120174
120175
120176
120177
120178
120179
120180
120181
120182
120183
120184
120185
120186
120187
120188
120189
120190
120191
120192
120193
120194
120195
120196
120197
120198
120199
120199
120200
120201
120202
120203
120204
120205
120206
120207
120208
120209
120210
120211
120212
120213
120214
120215
120216
120217
120218
120219
120220
120221
120222
120223
120224
120225
120226
120227
120228
120229
120230
120231
120232
120233
120234
120235
120236
120237
120238
120239
120240
120241
120242
120243
120244
120245
120246
120247
120248
120249
120250
120251
120252
120253
120254
120255
120256
120257
120258
120259
120260
120261
120262
120263
120264
120265
120266
120267
120268
120269
120270
120271
120272
120273
120274
120275
120276
120277
120278
120279
120280
120281
120282
120283
120284
120285
120286
120287
120288
120289
120290
120291
120292
120293
120294
120295
120296
120297
120298
120299
120299
120300
120301
120302
120303
120304
120305
120306
120307
120308
120309
120310
120311
120312
120313
120314
120315
120316
120317
120318
120319
120320
120321
120322
120323
120324
120325
120326
120327
120328
120329
120330
120331
120332
120333
120334
120335
120336
120337
120338
120339
120340
120341
120342
120343
120344
120345
120346
120347
120348
120349
120350
120351
120352
120353
120354
120355
120356
120357
120358
120359
120360
120361
120362
120363
120364
120365
120366
120367
120368
120369
120370
120371
120372
120373
120374
120375
120376
120377
120378
120379
120380
120381
120382
120383
120384
120385
120386
120387
120388
120389
120390
120391
120392
120393
120394
120395
120396
120397
120398
120399
120399
120400
120401
120402
120403
120404
120405
120406
120407
120408
120409
120410
120411
120412
120413
120414
120415
120416
120417
120418
120419
120420
120421
120422
120423
120424
120425
120426
120427
120428
120429
120430
120431
120432
120433
120434
120435
120436
120437
120438
120439
120440
120441
120442
120443
120444
120445
120446
120447
120448
120449
120450
120451
120452
120453
120454
120455
120456
120457
120458
120459
120460
120461
120462
120463
120464
120465
120466
120467
120468
120469
120470
120471
120472
120473
120474
120475
120476
120477
120478
120479
120480
120481
120482
120483
120484
120485
120486
120487
120488
120489
120490
120491
120492
120493
120494
120495
120496
120497
120498
120499
120499
120500
120501
120502
120503
120504
120505
120506
120507
120508
120509
120510
120511
120512
120513
120514
120515
120516
120517
120518
120519
120520
120521
120522
120523
120524
120525
120526
120527
120528
120529
120530
120531
120532
120533
120534
120535
120536
120537
120538
120539
120540
120541
120542
12054
```

```

1178     s.style.left = x+'px';
1179     s.style.y = y+'px';
1180     s.style.top = y+'px';
1181   }else{
1182     s.style.setProperty('position','absolute','important');
1183     s.style.setProperty('x','x','important');
1184     s.style.setProperty('left','x','important');
1185     s.style.setProperty('y','y','important');
1186     s.style.setProperty('top','y','important');
1187   }
1188 
1189   favicon = './favicon.ico';
1190   uv1 = u.split(":/");
1191   if( 2 <= uv1.length ){
1192     uv2 = uv1[1].split('/');
1193     if( 2 <= uv2.length ){
1194       if( //uv1[0] == 'file' ){
1195         //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/');
1196         // + '/favicon.ico';
1197         favicon = './favicon.ico';
1198       }else{
1199         favicon = uv1[0] + '://'+ uv2[0] + '/favicon.ico';
1200       }
1201     }
1202   }
1203   //wdlog("----- favicon url=" +favicon);
1204   href_id = 1.id + "href";
1205   l.innerHTML = '';
1206   +'' +u+''>+u+'</a>';
1207   /l.addEventListener('click', e => { onlickWin(e); });
1208   l.addEventListener('mousenter',e => { stopShell(e); });
1209   l.addEventListener('mouseleave',e => { contShell(e); });
1210   l.addEventListener('keydown', e => { onKeyDown(e); });
1211   l.addEventListener('keyup', e => { onKeyUp(e); });
1212 
1213   cmd.href_id = href_id;
1214   wdlog('0cmd_id '+cmd.id);
1215   wdlog('1href_id '+href_id);
1216   wdlog('2href_id=' +cmd.href_id);
1217   cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
1218 
1219   f.style.borderColor = c;
1220   f.style.outline = 'none';
1221   f.addEventListener('mousenter',e => { onEnterWin(e); });
1222   f.addEventListener('mouseleave',e => { onLeaveWin(e); });
1223 
1224   /s.addEventListener('click', e => { onlickWin(e); });
1225   /f.addEventListener('click', e => { wdlog('click wb'); });
1226   f.addEventListener('mozbrowsericonchange',onFaviconChange);
1227 
1228   wdlog('done settle VirtualBrowser url=' +u +'\n';
1229   + 'id' + s.id + '\n'
1230   + 'width' + s.style.width + '\n'
1231   + 'height' + s.style.height + '\n'
1232   + 'cma' + cmd.id
1233 );
1234 
1235 
1236 function WD_EventSetup2(){
1237   dt = VirtualDesktop();
1238   dt.style.width = "800px";
1239   dt.style.height = "500px";
1240   dt.addEventListener('dragstart',e => { onWinDragstart(e); });
1241   dt.addEventListener('drag', e => { onWinDrag(e); });
1242   dt.addEventListener('exit', e => { onWinDragexit(e); });
1243 }
1244 
1245 function GOnonClick(){
1246   WD_SaveScope(null); // should be push
1247   t = event.target;
1248   x = t.getAttribute('data-leftx');
1249   y = t.getAttribute('data-topy');
1250   zoom = WD_Zoom_1_XY.value;
1251   x *= zoom;
1252   y *= zoom;
1253   WD_doScrollXY(event,x,y);
1254   //alert('scroll #' +t.id+ ' x=' +x+ ', y=' +y);
1255 }
1256 
1257 function WD_SetGrid(e){
1258   t = e.target;
1259   WD_SetGrid(t);
1260 }
1261 
1262 function WD_SetGrid(t){
1263   //ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
1264   ds = VirtualDesktop_1_GridPlane;
1265   if( t.value == 'GridOn' ){
1266     for( col = 0; col < 16; col++ ){
1267       for( row = 0; row < 16; row++ ){
1268         g1 = document.createElement('span');
1269         g1.setAttribute('class','VirtualGrid');
1270         leftx = col * 800;
1271         topy = row * 500;
1272         gid = col + '.' + row;
1273         label = 'grid-' + gid;
1274         g1.innerHTML = label;
1275         g1.position = 'relative';
1276         g1.leftx = leftx;
1277         g1.topy = topy;
1278         g1.style.left = g1.leftx + 'px';
1279         g1.style.top = g1.topy + 'px';
1280         if( col % 2 == row % 2 ){
1281           g1.style.backgroundColor = 'rgba(255,255,255,0.3)';
1282         }
1283         ds.appendChild(g1);
1284       }
1285     }
1286   }
1287   t.value = 'GridOff';
1288 }else{
1289   ds.innerHTML = '';
1290   t.value = 'GridOn';
1291 }
1292 }
1293 
1294 var sav_scrollLeft;
1295 var sav_scrollTop;
1296 var sav_nscale;
1297 
1298 function WD_SaveScope(e){
1299   sav_scrollLeft = WD_Left_1.value;
1300   sav_scrollTop = WD_Top_1.value;
1301   sav_nscale = WD_Zoom_1_XY.value;
1302   //console.log('saved zoom=' +sav_nscale+ ',' +sav_nscale);
1303 }
1304 
1305 function WD_RestoreScope(e){
1306   WD_Zoom_1_XY.value = sav_nscale;
1307   WD_dotZoom();
1308 }
1309 
1310 WD_Left_1.value = sav_scrollLeft;
1311 WD_Top_1.value = sav_scrollTop;
1312 WD_dotZoom();
1313 
1314 function ignoreEvent(e{
1315   e.stopPropagation();
1316   //e.preventDefault();
1317 }
1318 
1319 function zoomMsg(){
1320   return WD_Zoom_1_MAG.value;
1321 }
1322 
1323 function WD_EventSetup(){
1324   WD_Grid_1.addEventListener('click', e => { WD_SetGrid(e); });
1325   WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
1326   WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
1327   WD_Width_1.value = dt.style.width;
1328   WD_Height_1.value = dt.style.height;
1329   WD_Height_1.addEventListener('keydown',ignoreEvent);
1330   WD_Height_1.addEventListener('keyup',ignoreEvent);
1331   WD_Zoom_1_MAG.addEventListener('keydown',ignoreEvent);
1332   WD_Zoom_1_MAG.addEventListener('keyup',ignoreEvent);
1333 }
1334 
1335 function escale(e,oscale,nscale){
1336   e.style.setProperty('transform','scale('+nscale+')');
1337   oscale = oscale / nscale;
1338   w = parseint(e.style.width);
1339   h = parseint(e.style.height);

```

```

11340 w = w * rscale; //oscale/nscale;
11341 h = h * rscale; //oscale/nscale;
11342 e.style.width = w + 'px';
11343 e.style.height = h + 'px';
11344 }
11345 function scaleWD(ds,oscale,nscale){
11346 if( true ){
11347   escale1(VirtualBrowser_1,oscale,nscale);
11348   escale1(VirtualBrowser_1_Location,oscale,nscale);
11349   escale1(VirtualBrowser_1_Command,oscale,nscale);
11350   escale1(VirtualBrowser_1_Frame,oscale,nscale);
11351 
11352   escale1(VirtualBrowser_2,oscale,nscale);
11353   escale1(VirtualBrowser_2_Location,oscale,nscale);
11354   escale1(VirtualBrowser_2_Command,oscale,nscale);
11355   escale1(VirtualBrowser_2_Frame,oscale,nscale);
11356 
11357   escale1(VirtualBrowser_3,oscale,nscale);
11358   escale1(VirtualBrowser_3_Location,oscale,nscale);
11359   escale1(VirtualBrowser_3_Command,oscale,nscale);
11360   escale1(VirtualBrowser_3_Frame,oscale,nscale);
11361 }
11362 }
11363 function WD_doZoom(){
11364   ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11365   oscale = WD_Zoom_1_XY.value; // hidden value for current zoom
11366   nscale = WD_Zoom_1_XY.value;
11367   ds.style.zoom = nscale;
11368   WD_Zoom_1_XY.value = ds.style.zoom;
11369   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11370   scaleWD(ds,oscale,nscale);
11371 }
11372 function WD_EventSetup(){
11373   WD_Zoom_1.addEventlistener('click',WD_doZoom);
11374   WD_Zoom_1_XY.addEventlistener('keydown',ignoreEvent);
11375   WD_Zoom_1_XY.addEventlistener('keyup',ignoreEvent);
11376 }
11377 function WD_doZoomOUT(){
11378   ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11379   oscale = WD_Zoom_1_XY.value;
11380   nscale = WD_Zoom_1_XY.value;
11381   if( oscale == 0 || oscale == '' ){
11382     oscale = 1;
11383   }
11384   nscale = oscale * zoomMag();
11385   ds.style.zoom = nscale;
11386   WD_Zoom_1_XY.value = ds.style.zoom;
11387   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11388   scaleWD(ds,oscale,nscale);
11389 }
11390 function WD_doZoomIN(){
11391   ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11392   oscale = WD_Zoom_1_XY.value;
11393   if( oscale == 0 || oscale == '' ){
11394     oscale = 1;
11395   }
11396   nscale = oscale * zoomMag();
11397   if( 4 < nscale ){
11398     alert('maybe too large, zoom=' + nscale);
11399     return;
11400   }
11401   ds.style.zoom = nscale;
11402   WD_Zoom_1_XY.value = ds.style.zoom;
11403   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11404   scaleWD(ds,oscale,nscale);
11405 }
11406 function WD_EventSetup(){
11407   WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11408   WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11409 }
11410
11411 function WD_doResize(e){
11412   dt = VirtualDesktop_1;
11413   dt.style.width = WD_Width_1.value;
11414   dt.style.height = WD_Height_1.value;
11415   WD_Width_1.value = dt.style.width;
11416   WD_Height_1.value = dt.style.height;
11417 }
11418 WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });
11419
11420
11421 function WD_dORSResize(e){
11422   /*alert('Resize Space');*/
11423   ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11424   ds.style.width = WS_1_Width.value;
11425   ds.style.height = WS_1_Height.value;
11426   WS_1_Width.value = ds.style.width;
11427   WS_1_Height.value = ds.style.height;
11428 }
11429
11430 function WD_EventSetup(){
11431   ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11432   ds.style.width = '5120px';
11433   ds.style.height = '2880px';
11434   WS_1_Width.value = ds.style.width;
11435   WS_1_Width.value = style.width;
11436   WS_1_Width.addEventListener('keydown',ignoreEvent);
11437   WS_1_Width.addEventListener('keyup',ignoreEvent);
11438   WS_1_Height.addEventListener('keydown',ignoreEvent);
11439   WS_1_Height.addEventListener('keyup',ignoreEvent);
11440   WS_Resize_1.addEventListener('click',e => { WD_dRSResize(e); });
11441 }
11442
11443 function WD_doScrollXY(e,sleft,stop){
11444   dt = VirtualDesktop_1;
11445   dt.scrollLeft = sleft;
11446   dt.scrollTop = stop;
11447   WD_Left_1.value = dt.scrollLeft;
11448   WD_Top_1.value = dt.scrollTop;
11449   console.log('--Scroll #' + dt.id+ ' ('+sleft+','+stop+')');
11450 }
11451
11452 function WD_doScroll(e){
11453   /*dt = VirtualDesktop_1_Content; */
11454   sleft = parseInt(WD_Left_1.value);
11455   stop = parseInt(WD_Top_1.value);
11456   dt.scrollLeft = sleft;
11457   dt.scrollTop = stop;
11458   WD_Left_1.value = dt.scrollLeft;
11459   WD_Top_1.value = dt.scrollTop;
11460   console.log('--Scroll #' + dt.id+ ' ('+sleft+','+stop+')');
11461 }
11462
11463 function showScrollPosition(){
11464   /*dt = VirtualDesktop_1_Content; */
11465   'wstop' + VirtualDesktop_1.style.top + ',' +
11466   'wsx' + VirtualDesktop_1.style.y + ',' +
11467   'wsy' + VirtualDesktop_1.scrollTop + ',' +
11468   'wdtop' + VirtualDesktop_1_Content.style.top + ',' +
11469   'wdx' + VirtualDesktop_1_Content.style.y + ',' +
11470   'wds' + VirtualDesktop_1_Content.scrollTop + ',';
11471   WD_Top_1.value = VirtualDesktop_1.scrollTop;
11472   WD_Top_1.value = VirtualDesktop_1.scrollTop;
11473 }
11474
11475 function WD_EventSetup(){
11476   WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
11477   WD_Left_1.addEventListener('keydown',ignoreEvent);
11478   WD_Left_1.addEventListener('keyup',ignoreEvent);
11479   WD_Top_1.addEventListener('keydown',ignoreEvent);
11480   WD_Top_1.addEventListener('keyup',ignoreEvent);
11481 }
11482
11483 function WD_EventSetup(){
11484   VirtualDesktop_1.addEventListener('scroll',showScrollPosition);
11485   VirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
11486 }
11487
11488 if( false ){
11489   w = 1000 + 'px';
11490   dt.style.width = w;
11491   dt.style.height = '300px';
11492   dt.style.resize = 'both';
11493   dt.style.borderWidth = '50 + 'px';
11494   dt.style.borderColor = '#'+dt.id+' border' + '50px';
11495   dt.style.borderRadius = '25 + 'px';
11496   console.log('----- #' + dt.id + ' style=' + dt.style);
11497   console.log('----- #' + dt.id + ' width=' + dt.style.width);
11498   console.log('----- #' + dt.id + ' left=' + dt.style.left);
11499   console.log('----- #' + dt.id + ' border=' + dt.style.borderWidth);
11500 }
11501
11502 function onmTResize(e){
11503   dt = e.target;
11504   h = parseInt(dt.style.height);
11505   dt.style.borderWidth = (h * 0.075) + 'px';

```

```

11502 console.log('----- borderWidgh=' + dt.style.borderWidth);
11503 }
11504
11505 VirtualDesktopDetails.addEventListener('toggle', VirtualDesktop_init);
11506 function VirtualDesktop_init(){
11507   if( !VirtualDesktopDetails.open ){
11508     return;
11509   }
11510   //CJ Join();
11511   VirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
11512   //console.log("----- #"+dt.id);
11513   // + borderWidgh'+dt.style.getProperty('border-width'));
11514
11515   settleWin();
11516   VirtualBrowser_1,
11517   VirtualBrowser_1_Location,
11518   VirtualBrowser_1_Command,
11519   VirtualBrowser_1_Frame,
11520   document.UB,
11521   500,280,50,20,'#262',1.0);
11522   settleWin();
11523   VirtualBrowser_2,
11524   VirtualBrowser_2_Location,
11525   VirtualBrowser_2_Command,
11526   VirtualBrowser_2_Frame,
11527   'https://lits-more.jp/ja/jp/',
11528   500,280,150,100,'#444',1.0);
11529   settleWin();
11530   VirtualBrowser_3,
11531   VirtualBrowser_3_Location,
11532   VirtualBrowser_3_Command,
11533   VirtualBrowser_3_Frame,
11534   '//gshell/gsh.go.html',
11535   'http://gshell.org/gshell/gsh.go.html',
11536   'https://golang.org',
11537   500,280,250,180,'#444',1.0);
11538   //1000,280,0,0,'#444',0.125);
11539   //720,280,100,-50,'#444',0.4);
11540   function WD_ClockUpdate(s){
11541     VirtualDesktop_1.Clock.innerHTML = DateShort();
11542     VirtualDesktop_1_Calender.innerHTML = DateHourMin();
11543   }
11544   window.setInterval(WD_ClockUpdate,500);
11545
11546   WD_EventSetup1();
11547   WD_EventSetup2();
11548   WD_EventSetup3();
11549   WD_EventSetup4();
11550   WD_EventSetup5();
11551   WD_EventSetup6();
11552   WD_EventSetup7();
11553   WD_EventSetup8();
11554 }
11555 //VirtualDesktop_init();
11556
11557 Destroy_VirtualDesktop = function(){
11558   VirtualDesktop_1.style = "";
11559
11560   VirtualBrowser_1.removeAttribute('style');
11561   VirtualBrowser_1_Location.innerHTML = '';
11562   VirtualBrowser_1_Command.removeAttribute('src');
11563   VirtualBrowser_1_Frame.removeAttribute('style');
11564   VirtualBrowser_1_Frame.style="";
11565
11566   VirtualBrowser_2.removeAttribute('style');
11567   VirtualBrowser_2_Location.innerHTML = '';
11568   VirtualBrowser_2_Frame.removeAttribute('src');
11569   VirtualBrowser_2_Frame.style="";
11570
11571   VirtualBrowser_3.removeAttribute('style');
11572   VirtualBrowser_3_Location.innerHTML = '';
11573   VirtualBrowser_3_Frame.removeAttribute('src');
11574   VirtualBrowser_3_Frame.style="";
11575
11576   GJfactory_1.style = "";
11577   iframe0.style = "";
11578   VirtualDesktop_1.style = "";
11579 }
11580
11581</script>
11582<!-- VirtualDesktop -->
11583<details>
11584<!--</span>
11585<!--<br>
11586<!-- ===== Work { ===== -->
11587<span id="SBSidebar_WorkCodeSpan">
11588</span>
11589<details><summary>SBSidebar // 2020-0928 SatoxITS { -->
11590<div>
11591<h3>SBSidebar</h3>
11592<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11593<input id="SBSidebar_CodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11594<input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11595<span id="SBSidebar_WorkCodeView"></span>
11596<script>SBSidebar.openWorkCodeView();</script>
11597function SBSidebar_showWorkCode(){
11598  showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
11599}</div>
11600<SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
11601
11602SBSidebar_openWorkCodeView(); // should be invoked by an event
11603
11604console.log('-- SBSlider // 2020-1006-01 SatoxITS --');
11605function SetSlider(){
11606  slider = document.getElementById('secondary');
11607  slider.parentNode.style['primary'+ 'secondary' + 'main'+ 'main+ '];
11608  // slider.style['primary'+ 'secondary'+ 'main'+ 'main+ '];
11609  wrap = slider.parentNode;
11610  console.log('-- SBSlider parent is '+wrap+', #' +wrap.id+ ' .' +wrap.class);
11611  //wrap = wrap.parentNode;
11612  console.log('-- SBSlider parent is '+wrap+', #' +wrap.id+ ' .' +wrap.class);
11613  //wrap = wrap.parentNode;
11614  //wrap.style['primary'+ 'secondary'+ 'main'+ 'main+ '];
11615  //wrap.style['primary'+ 'secondary'+ 'main'+ 'main+ '];
11616  nsb = sidebar.cloneNode();
11617  nsb = sidebar;
11618  nsb.style.width = '100%';
11619  slider = document.createElement('div');
11620  slider.id = 'SBSlider';
11621  slider.appendChild(sidebar);
11622  slider.setAttribute('class','SBSlider');
11623  nsb.style.position = 'relative';
11624  slider.style.position = 'fixed';
11625  slider.style.display = 'block'; // inline';
11626  slider.style.width = 100000;
11627  // nsb.style.zIndex = 100000;
11628  nsb.style.position = 'absolute';
11629  nsb.style.minWidth = '80px';
11630  nsb.style.left = '0px';
11631  nsb.style.top = '0px';
11632  w = window.innerWidth;
11633  console.log('SliderWidth '+w+: ' ,(w/3)+'px');
11634  if( w < 640 )
11635    slider.style.setProperty('width',(w/3) + 'px','important');
11636
11637  main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
11638
11639  slider.style.resize = "both";
11640  slider.draggable = "true";
11641  wrap.appendChild(slider);
11642  console.log('-- SBSlider');
11643  //nsb.addEventListener('scroll',SBSrolled);
11644
11645  buttons = document.createElement('div');
11646  buttons.id = 'NaviButtons';
11647  buttons.setAttribute('class','NaviButtons');
11648  buttons.innerHTML = '<+' +'

<a href="#PopOffPost" draggable="true">>TOP<+' +'/a><+' +'/p>';
11649  buttons.innerHTML += '<+' +'

<a href="#EndOffPost" draggable="true">>END<+' +'/p>';
11650  page.appendChild(buttons);
11651  buttons.style.position = 'fixed';
11652  buttons.style.top = '30000';
11653  buttons.style.width = '100%';
11654  buttons.style.height = '100%';
11655  buttons.style.top = '320px';
11656  buttons.style.left = parseInt(w) * 1.0 + 'px';
11657  console.log('-- SBSlider installed (~) / SatoxITS');
11658
11659//window.addEventListener('load',SetSidebar); // after load
11660SetSidebar = function(){
11661  nb = document.getElementById('NaviButtons');
11662  if( nb != null ){
11663    nb.parentNode.removeChild(NaviButtons);


```

```

11664 }
11665 }
11666</script>
11667
11668// 2020-1006 its-more.jp-blog-60000-style.css
11669<!-- {
11670 #navButtons {
11671 position:fixed;
11672 width:100px;
11673 xwldth:100%;
11674 xxtlpt:100px;
11675 xxeleft:10px;
11676 xxeleft:10px;
11677 z-index:3000;
11678 font-size:10pt;
11679 color:#fff !important;
11680 text-align:center;
11681 background-color:rgba(230,230,230,0.01);
11682 }
11683 #navButtons a {
11684 color:#f2f !important;
11685 font-size:12px;
11686 text-align:center;
11687 xtext-shadow:2px 2px #0ff;
11688 resize:both;
11689 padding:10px;
11690 margin:10px;
11691 border:1px solid #288 !important;
11692 border-radius:3px;
11693 background-color:rgba(160,160,160,0.05);
11694 }
11695 #SBSlider {
11696 overflow:auto;
11697 resize:both !important;
11698 xoverflow-yhidden !important;
11699 height:100px !important;
11700 display:inline-block;
11701 position:fixed !important;
11702 left:0px;
11703 top:0px;
11704 xwidht:180px;
11705 width:24px;
11706 margin:0 auto 80px;
11707 height:100% !important;
11708 background-color:rgba(100,100,200,0.1);
11709 }
11710 #secondary {
11711 position:fixed;
11712 left:0px;
11713 top:0px;
11714 xxxx-index:60000;
11715 scroll-behavior: overflow !important;
11716 xxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11717 padding-left:4pt;
11718 color:#000;
11719 font-size:0.5em;
11720 background-color:rgba(64,160,64,0.6) !important;
11721 white-space:nowrap;
11722 }
11723 #secondary a {
11724 color:#fff !important;
11725 text-decoration:underline !important;
11726 }
11727 #primary {
11728 position:relative;
11729 width:758 !important;
11730 left:259 !important;
11731 }
11732 #main {
11733 position:relative;
11734 width:758 !important;
11735 left:259 !important;
11736 }
11737 #site-navigation {
11738 position:relative;
11739 left:120px;
11740 }
11741 #adswe_cowntext {
11742 color:#119ae;
11743 font-size:16pt !important;
11744 xfont-size:10% !important;
11745 font-weight:bold;
11746 }
11747 #nowTime {
11748 color:#00ffa0;
11749 font-size:16pt !important;
11750 xfont-size:10% !important;
11751 font-weight:bold;
11752 text-shadow:1px 1px #fff;
11753 }
11754 .navigation-top {
11755 color:#22a !important;
11756 border:0px;
11757 background-color:rgba(220,220,220,0.1);
11758 }
11759
11760 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11761 display: block;
11762 xxwidth: 1em;
11763 xoverflow: auto;
11764 xxheight: 1em;
11765 }
11766 .invisible-scrollbar ::-webkit-scrollbar {
11767 xdisplay: none;
11768 width:1px !important;
11769 height:px !important;
11770 }
11771 .mostly-customized-scrollbar ::-webkit-scrollbar {
11772 width: 2px;
11773 height: 2px;
11774 xxbackground-color: #aaa; xxxor add it to the track;
11775 }
11776 .mostly-customized-scrollbar ::-webkit-scrollbar-thumb {
11777 background: #000;
11778 }
11779</style>
11780 }-->
11781
11782
11783</details>
11784<!-- SBSideBar_WorkCodeSpan } -->
11785<!-- //<span>
11786//<-- ===== Work } ===== -->
11787
11788
11789//<-- ===== Work { ===== -->
11790<span id="Affiliate_WorkCodeSpan">
11791</*
11792<details id="Affiliate_Test"><summary>Affiliate</summary>
11793<!-- ===== Affiliate // 2020-1010 SatoxITS { -->
11794<div id="AffViewDock">
11795<div id="AffView" class="AffView" draggable="true" style="">
11796<div id="AffSet" class="AffPlate">
11797 <iframe id="aff_0" class="AffItem"></iframe>
11798 <iframe id="aff_1" class="AffItem"></iframe>
11799 <iframe id="aff_2" class="AffItem"></iframe>
11800 <iframe id="aff_3" class="AffItem"></iframe>
11801 <iframe id="aff_4" class="AffItem"></iframe>
11802 <iframe id="aff_5" class="AffItem"></iframe>
11803</div></div>
11804<h2>Supportive Affiliate</h2>
11805<style>
11806.AffView {
11807 z-index:0;
11808 overflow-x:scroll;
11809 overflow-y:scroll;
11810 position:fixed;
11811 position:fixed;
11812 max-width:2560px;
11813 max-height:100%;
11814 width:270px;
11815 left:75px;
11816 height:95%;
11817 resize:both;
11818 xleft:-10px;
11819 margin-top:40px;
11820 xleft:0;
11821 xright:0;
11822 display:block;
11823 border:4px inset rgba(255,255,255,0.1);
11824 background-color:rgba(255,255,255,0.1);
11825 }

```

```

11826 .Affview:hover {
11827   z-index:1;
11828   width:300px;
11829   overflow:scroll;
11830   border:4px inset #fcc;
11831   background-color:rgba(80,80,255,0.2);
11832   background-color:#ffc;
11833 }
11834 .Affplate:hover {
11835   border-left:4px dashed #088;
11836 }
11837 .Affplate{
11838   overflow-x:visible;
11839   border-left:4px dashed rgba(255,255,255,0.1);
11840   max-width:2560px;
11841   max-height:2880px;
11842   margin-top:10px;
11843   margin-bottom:10px;
11844   margin-left:4px;
11845   width:300px;
11846   xheight:1440px;
11847 }
11848 .Affter {
11849   overflow-x:visible;
11850   xoverflow-y:scroll;
11851   max-height:2560px;
11852   min-height:1440px;
11853   z-index:0;
11854   display:block;
11855   xposition:fixed;
11856   xposition:absolute;
11857   transform:translate(-0.5);
11858   /*left:100px;
11859   xresizerboth;
11860   padding:0px;
11861   width:600px;
11862   height:100px;
11863   max-height:800px !important;
11864   margin-top:0px;
11865   margin-left:0px;
11866   margin-right:0px !important;
11867   border:16px inset #ccc;
11868   transform:translate(-0.5);
11869   background-color:rgba(255,255,255,0.2);
11870   xalign:right;
11871 }
11872 .Afftemp:hover {
11873   border:16px inset #bbf;
11874 }
11875 </style>
11876 <input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11877 <input id="Affiliate_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11878 <input id="Affiliate_CodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11879 <span id="Affiliate_WorkCodeView"></span>
11880 <script>
11881   function Affiliate_OpenWorkCodeView(){
11882     function Affiliate_showWorkCode(){
11883       showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
11884     }
11885     Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
11886   }
11887 Affiliate_OpenWorkCodeView(); // should be invoked by an event
11888 </script>
11889 <!--<iframe id="aff_8" xsrc="https://stackoverflow.com/tags" class="Affitem"></iframe>
11890 --<iframe id="aff_9" xsrc="https://developer.mozilla.org/en-US/docs/Web" class="Affitem"></iframe>
11891 var Aff = {};
11892 Affiliate_Temp.addEventListener('click',Aff_Setup);
11893 function Aff_Setup(){
11894   if(Aff.isSetup){return;} Aff.isSetup = true;
11895   parent = document.documentElement;
11896   parent.appendChild(AffView);
11897   AffView.style.top = '0px';
11898   AffView.style.left = (window.innerWidth - 280) + 'px';
11899   var off = 100;
11900   zoom = 0.5;
11901   ozoom = 0.3;
11902   lwidth = window.innerWidth - 300;
11903   left = lwidth + 'px';
11904   left = -130 + 'px';
11905   left = -130 + 'px';
11906   console.log('aff-init window.innerWidth=' + window.innerWidth);
11907   w = 1000;
11908   h = 560;
11909   aff_0.src = './gshell1/gsh.go.html';
11910   aff_1.src = 'https://golang.org';
11911   aff_2.src = 'https://drafts.csswg.org/';
11912   aff_3.src = 'https://html.spec.whatwg.org/dev/';
11913   aff_4.src = 'https://www.wikipedia.org';
11914   aff_5.src = 'https://www.bing.com/translator';
11915   /parent.appendChild(aff_0);
11916   aff_0.style.width = 'zoom*w+px';
11917   aff_0.style.height = 'zoom*h+px';
11918   aff_0.style.left = left;
11919   /aff_0.style.top = off+'px'; off += ozoom*h;
11920   aff_0.draggable = 'true';
11921   /parent.appendChild(aff_1);
11922   aff_1.style.width = 'zoom*w+px';
11923   aff_1.style.height = 'zoom*h+px';
11924   aff_1.style.left = left;
11925   /aff_1.style.top = off+'px'; off += ozoom*h;
11926   aff_1.style.top = '-150px';
11927   aff_1.draggable = 'true';
11928   /parent.appendChild(aff_2);
11929   aff_2.style.width = 'zoom*w+px';
11930   aff_2.style.height = 'zoom*h+px';
11931   aff_2.style.left = left;
11932   /aff_2.style.top = off+'px'; off += ozoom*h;
11933   aff_2.style.top = '-300px';
11934   aff_2.draggable = 'true';
11935   /parent.appendChild(aff_3);
11936   aff_3.style.transform = 'scale(0.25)';
11937   aff_3.style.width = '2*zoom*w+px';
11938   aff_3.style.height = '2*zoom*h+px';
11939   aff_3.style.border = '2px inset #ccc';
11940   aff_3.style.left = -390 + 'px'; //left*2;
11941   /aff_3.style.left = (leftx - 265) + 'px';
11942   aff_3.style.left = -395 + 'px';
11943   /aff_3.style.top = (-155*off) + 'px'; off += ozoom*h;
11944   aff_3.style.top = '-600px';
11945   aff_3.draggable = 'true';
11946   /parent.appendChild(aff_4);
11947   aff_4.style.width = 'zoom*w+px';
11948   aff_4.style.height = 'zoom*h+px';
11949   aff_4.style.left = left;
11950   /aff_4.style.top = off+'px'; off += ozoom*h;
11951   aff_4.style.top = '-900px';
11952   aff_4.draggable = 'true';
11953   /parent.appendChild(aff_5);
11954   aff_5.style.transform = 'scale(0.300)';
11955   aff_5.style.width = 'zoom*(w*1.67)+px';
11956   aff_5.style.height = 'zoom*(h*1.67)+px';
11957   aff_5.style.border = '2px inset #ccc';
11958   aff_5.style.left = -308 + 'px';
11959   /aff_5.style.left = (-98*off) + 'px'; off += ozoom*h;
11960   aff_5.style.left = '0px';
11961   /aff_5.style.top = '0px';
11962   aff_5.style.top = '-1150px';
11963   //aff_5.style.align = 'right';
11964   aff_5.draggable = 'true';
11965   window.addEventListener('resize',affresize);
11966 }
11967 function affresize(){
11968   AffView.style.left = (window.innerWidth - 280) + 'px';
11969   left = window.innerWidth - 400;
11970   left = leftx + 'px';
11971   console.log('aff-resize window.innerWidth=' + window.innerWidth);
11972 }
11973 //window.addEventListener('resize',affresize);
11974 //document.addEventListener('resize',affresize);
11975 //gsh.addEventListener('resize',affresize);
11976 
```

11977 Affiliate_Temp.append(AffView);

11978

```

11988 AffView.removeAttribute('style');
11989 aff_0.removeAttribute('src');
11990 aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
11991 aff_1.removeAttribute('style');
11992 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
11993 aff_2.removeAttribute('style');
11994 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
11995 aff_3.removeAttribute('style');
11996 aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
11997 aff_4.removeAttribute('src');
11998 aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
11999 aff_5.removeAttribute('src');
12000 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12001 }
12002 </script>
12003 <!--Details-->
12004 <!-- Affiliate_WorkCodeSpan -->
12005 <!-- Affiliate_Span -->
12006 <!-- ===== Work ===== -->
12007
12008
12009
12010 //<!-- ===== Work { ===== -->
12011 <span id="FontSelector_WorkCodeSpan">
12012 /*
12013 <details><summary>Font Selector</summary>
12014 <!-- ===== FontSelector // 2020-1013 SatoxiTS ( -->
12015 <h2>Font Selection</h2>
12016 <div id="FontList"></div>
12017 </div>
12018 <style>
12019 #FontList {
12020   overflow:visible;
12021   background-color:rgba(240,245,255,1.0) !important;
12022 }
12023 #FontList td {
12024   font-size:20px;
12025   padding:0px;
12026   padding-left:2px;
12027   padding-right:2px;
12028   margin:0px;
12029   line-height:1.2;
12030   border:1px solid #000;
12031 }
12032 #FontList tr:hover {
12033   color:#fff;
12034   background-color:#000;
12035   border:1px solid #000;
12036 }
12037 .courier { color:#000; font-size:16px; font-family:courier; }
12038 .cursive { color:#000; font-size:16px; font-family:cursive; }
12039 .fantasy { color:#000; font-size:16px; font-family:fantasy; }
12040 .georgia { color:#000; font-size:16px; font-family:georgia; }
12041 .monospace { color:#000; font-size:16px; font-family:monospace; }
12042 .xmonospad { color:#000; font-size:16px; font-family:xmonospad; }
12043 </style>
12044 <script>
12045 <!--
12046 function fontstr(name,text){
12047   tx = '<'+tr style="font-family:'+name+'>\n';
12048   tr += '<'+tr style="font-family:'+name+'>\n';
12049   tr += '<'+td style="font-family:Arial;font-size:12pt;'+>'+name+'<'+/td>';
12050   tr += '<'+td style="font-family:Times;font-size:12pt;'+>'+text+'<'+/td>';
12051   tr += '<'+td style="font-family:Times;font-size:12pt;'+>'+text+'<'+/td>';
12052   tr += '<'+td style="font-family:Times;font-size:12pt;'+>'+text+'<'+/td>';
12053   tr += '<'+tr style="font-family:Times;font-size:12pt;'+>'+text+'<'+/td>';
12054   return tr;
12055 }
12056 <!--
12057 function lsfont(){
12058   text = 'GSShell-Go012';
12059
12060   fl = '';
12061   fl += '<table>\n';
12062   fl += fontstr('Arial',text);
12063   fl += fontstr('Courier',text);
12064   fl += fontstr('Courier New',text);
12065   fl += fontstr('Georgia',text);
12066   fl += fontstr('Helvetica',text);
12067   fl += fontstr('Verdana',text);
12068   fl += fontstr('Times',text);
12069
12070   fl += fontstr('Osaka',text);
12071   fl += fontstr('Meiryo',text);
12072   fl += fontstr('YuMincho',text);
12073
12074   fl += fontstr('Roman',text);
12075   //document.fonts.load('30px cursive');
12076   fl += fontstr('Serif',text);
12077   fl += fontstr('Sans-Serif',text);
12078   fl += fontstr('System-U1',text);
12079   fl += fontstr('Monospace',text);
12080   fl += fontstr('Cursive',text);
12081   fl += fontstr('Fantasy',text);
12082   fl += '</table>\n';
12083
12084 if( false ){
12085   document.fonts.entries(); // FontfaceSet
12086   console.log('FSSe: '+fss);
12087   while( true ){
12088     font = fss.next();
12089     if( font.done ) {
12090       break;
12091     }
12092     fl += font.value[0] + '<br>';
12093   }
12094 }
12095 FontList.innerHTML = fl;
12096
12097 function FontList_Setup(){
12098   lsfont();
12099 }
12100 document.fonts.onloadingdone = function(fsse){
12101   //alert('font-loaded '+fssse.fontfaces.length);
12102 }
12103 </script>
12104
12105 <h2>Drawing Text on Canvas</h2>
12106 <h3>2020-1012 Drawing Text on Canvas // SatoxiTS ( -->
12107 <div id="TextCanvasPanel" class="TextCanvasPanel">
12108 <input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
12109 <input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
12110 <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
12111 <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12112 <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">>Bold
12113 <input id="TextCanvas_1_Iitalic" class="CanvasBox" type="checkbox" checked="">>italic
12114 <input id="TextCanvas_1_Text" type="text" size="50" value="GSshell">
12115 </div>
12116 <div id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></div>
12117 <style>
12118 .CommandUsageText {
12119   font-family:Courier New;
12120 }
12121 .TextCanvas {
12122   border:1px solid #000;
12123   resize:both;
12124   display:inline-block;
12125 }
12126 .CanvasBox {
12127   vertical-align:middle;
12128   margin-left:4px !important;
12129   margin-right:2px !important;
12130 }
12131 .CanvasPanel {
12132   vertical-align:middle;
12133   height:4pt !important;
12134   width:inherit !important;
12135   padding:2px !important;
12136   margin:4px !important;
12137   margin-top:4px !important;
12138   margin-right:2px !important;
12139   font-size:10pt !important;
12140   font-family:Georgia !important;
12141   color:#000;
12142   display:inline-block;
12143 }
12144 .TextCanvasPanel {
12145   vertical-align:middle;
12146   height:14pt !important;
12147   width:14pt !important;
12148   border:1px solid #000;
12149   border-radius:50% !important;
12150   font-size:14pt !important;
12151   font-family:Georgia !important;
12152   color:#000;
12153   display:inline-block;
12154 }

```

```

12150 font-size:10pt !important;
12151 font-family:Georgia !important;
12152 color:#000;
12153 display:inline !important;
12154 }
12155 .PanelButton {
12156 font-size:10pt !important;
12157 font-family:Georgia !important;
12158 vertical-align:middle;
12159 width:45pt !important;
12160 height:14pt !important;
12161 line-height:1.2 !important;
12162 padding:2px !important;
12163 margin:1px !important;
12164 display:inline !important;
12165 padding:1px !important;
12166 color:#fff !important;
12167 background-color:#228 !important;
12168 }
12169 <style>
12170 </script>
12171 function DrawTextCanvas(){
12172 ctx = TextCanvas_1.getContext('2d');
12173 var textfont = '';
12174 if( TextCanvas_1_Italic.checked ) textfont += ' italic';
12175 if( TextCanvas_1_Bold.checked ) textfont += ' bold';
12176 textfont += 'TextCanvas_1_Size.value'+'px';
12177 textfont += ' '+TextCanvas_1_Font.value;
12178 //ctx.font = 'italic bold 64px Georgia';
12179 //console.log('TxFont'+textfont);
12180 ctx.font = textfont;
12181 ctx.fillText(TextCanvas_1_Text.value,10,80);
12182 }
12183 DrawTextCanvas();
12184 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12185 function ClearTextCanvas(){
12186 cv = TextCanvas_1_;
12187 ctx = cv.getContext('2d');
12188 ctx.clearRect(0,0,cv.width,cv.height);
12189 }
12190 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12191 <!-- } -->
12192 <!-- } -->
12193 <script>
12194 //TextCanvas_1_Panel.addEventListener('mousein',OffGJShell);
12195 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
12196 </script>
12197 </details>
12198 <div class="FontSelector_WorkCodeSpan" -->
12199 <!-- ===== Work -->
12200 </div>
12201 <!-- ===== Work -->
12202
12203 <!-- ===== Work -->
12204 <span id="Shading_WorkCodeSpan">
12205 <details><summary>Shading Canvas</summary>
12206 <!-- Shading Canvas // 2020-10-11 SatoxITS { -->
12207 <div>Shading Canvas</div>
12208 <div class="CommandUsageText">
12209 <hr>
12210 Placement Mode<br>
12211 a ... apply (int absolute position)<br>
12212 bring down (ArrowDown)<br>
12213 k ... bring up (ArrowUp)<br>
12214 l ... bring left (ArrowLeft)<br>
12215 l ... bring right (ArrowRight)<br>
12216 z-index = 0<br>
12217 z-index += 1<br>
12218 z-index ...<br>
12219 z-index ...<br>
12220 z-index ...<br>
12221 z-index ... to here (relative position)<br>
12222 c ... clear the log text<br>
12223 Note: the HTML text must be contenteditable to catch Key Event.<br>
12224 </note>
12225
12226
12227 <div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
12228 <div id="Shading_1_Html" class="ShadingHtml" draggable="true"></div>
12229 <div id="Shading_1_Log" class="ShadingLog" draggable="true" style=""></div>
12230
12231 <canvas id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</canvas></div>
12232 <style>
12233 .ShadingPlate {
12234 z-index:0;
12235 position:static;
12236 overflow:scroll;
12237 display:block;
12238 width:100px;
12239 height:400px;
12240 font-size:9pt;
12241 font-family:Courier New;
12242 border:1px dashed #000;
12243 color:#444;
12244 }
12245 .ShadingLog {
12246 z-index:0;
12247 position:relative;
12248 display:block;
12249 top:0px;
12250 left:0px;
12251 overflow:scroll;
12252 width:100px;
12253 font-size:9pt;
12254 font-family:Courier New;
12255 color:#666;
12256 overflow:scroll;
12257 background:rgba(200,255,200,0.4);
12258 height:400px;
12259 }
12260 .ShadingHtml {
12261 z-index:2;
12262 position:relative;
12263 display:block;
12264 top:0px;
12265 left:0px;
12266 overflow:scroll;
12267 width:100px;
12268 font-size:12pt;
12269 font-family:Courier New;
12270 color:#666;
12271 overflow:scroll;
12272 background:rgba(200,255,200,0.4);
12273 height:400px;
12274 }
12275 .ShadingCanvas {
12276 z-index:3;
12277 position:relative;
12278 display:block;
12279 top:0px;
12280 left:100px;
12281 resize:both;
12282 border:1px solid #000;
12283 }
12284 </style>
12285 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12286 <input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12287 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12288 <span id="Shading_WorkCodeView"></span>
12289 <script id="Shading_WorkScript">
12290 function Shading_OpenWorkView(){
12291 Shading_ShowWorkCode();
12292 showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
12293 }
12294 Shading_WorkCodeViewOpen.addEventListener('click',Shading_ShowWorkCode);
12295 }
12296 const BR = '<'+<br>;
12297 Shading_OpenWorkCodeView(); // should be invoked by an event
12298 function sh_onClick(e){
12299 Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
12300 + ' offset('+e.offsetX+', '+e.offsetY+')'
12301 + ' client('+e.clientX+', '+e.clientY+')'
12302 + ' page('+e.pageX+', '+e.pageY+')'
12303 + ' screen('+e.screenX+', '+e.screenY+')'
12304 + BR;
12305 e.stopPropagation();
12306 e.preventDefault();
12307 }
12308 function sh_onKeyup(e){
12309 if( Shading_1.style.zIndex == '' ){
12310 Shading_1.style.zIndex = 0;
12311 }
}

```

```

12312     zi = parseInt(Shading_1.style.zIndex);
12313 
12314     if( e.key.length == 1 ){
12315         Shading_1.Html.innerHTML += e.key;
12316     }
12317 
12318     if( e.key == '0' ){ zi = 0; }else
12319     if( e.key == '+' ){ zi += 1; }else
12320     if( e.key == '-' ){ zi -= 1; }else
12321     if( e.key == 'c' ){
12322         Shading_1_Log.innerHTML = '';
12323     }else
12324     if( e.key == 'r' ){
12325         Shading_1.style.position = "relative";
12326         Shading_1.style.top = '0px';
12327         Shading_1.style.left = '0px';
12328         zi = 0;
12329     }else
12330     if( e.key == 'j' || e.code == 'ArrowDown' ){
12331         topx = parseInt(Shading_1.style.top) + 50;
12332         Shading_1.style.top = topx + 'px';
12333     }else
12334     if( e.key == 'k' || e.code == 'ArrowUp' ){
12335         topx = parseInt(Shading_1.style.top) - 50;
12336         Shading_1.style.top = topx + 'px';
12337     }else
12338     if( e.key == 'l' || e.code == 'ArrowRight' ){
12339         lefty = parseInt(Shading_1.style.left) + 50;
12340         Shading_1.style.left = lefty + 'px';
12341     }else
12342     if( e.key == 'h' || e.code == 'ArrowLeft' ){
12343         lefty = parseInt(Shading_1.style.left) - 50;
12344         Shading_1.style.left = lefty + 'px';
12345     }else
12346     if( e.key == 'a' ){
12347         Shading_1.style.position = "absolute";
12348         Shading_1.style.top = '0px';
12349         Shading_1.style.left = '0px';
12350     }else{
12351         Shading_1.style.zIndex = zi;
12352         Shading_1_Log.innerHTML += 'Keypdown'+e.target.nodeName+'#'+e.target.id
12353         +'Up'+e.key+'#'+e.code+
12354         '+'index'+zi+'#'+shading_1.style.zIndex
12355         +'top:' + Shading_1.style.top
12356         +'BR';
12357     }
12358     e.stopPropagation();
12359     e.preventDefault();
12360 }
12361 function sh_onKeydown(e){
12362     Shading_1_Log.innerHTML += 'Keypadown'+e.target.nodeName+'#'+e.target.id
12363     +'Down'+e.key+'/' +e.code+')' +BR;
12364     e.stopPropagation();
12365     e.preventDefault();
12366 }
12367 function Shading_Setup(){
12368     Shading_1_Log.innerHTML += '<'+'h4>Click here' +'/h4>';
12369     Shading_1.addEventListener('keydown',sh_onKeydown);
12370     Shading_1.addEventListener('keyup',sh_onKeyUp);
12371     Shading_1.addEventListener('click',sh_onClick);
12372     Shading_1.addEventListener('click',sh_onClick);
12373 
12374     Shading_1_Log.style.top = "-400px";
12375     Shading_1_Log.style.left = "200px";
12376 
12377     Shading_1.appendChild(Shading_1_Canvas);
12378     Shading_1_Canvas.style.width = "300px";
12379     Shading_1_Canvas.style.height = "300px";
12380     Shading_1_Canvas.style.position = "relative";
12381     Shading_1_Canvas.style.top = "-750px";
12382     Shading_1_Canvas.style.left = "100px";
12383 
12384     const ctx = Shading_1_Canvas.getContext('2d');
12385     ctx.fillStyle = 'rgba(160,0,0,0.9)';
12386     ctx.fillRect(50,50,40,40);
12387     ctx.fillStyle = 'rgba(0,160,0,0.9)';
12388     ctx.fillRect(60,60,40,40);
12389     ctx.fillStyle = 'rgba(0,0,160,0.9)';
12390     ctx.fillRect(70,70,40,40);
12391 }
12392 function Reset_ShadingCanvas(){
12393     Shading_1_Log.removeAttribute('style');
12394     Shading_1_Log.innerHTML = '';
12395     Shading_1_Canvas.style = "#fff";
12396     //Shading_1_Canvas.removeAttribute('style');
12397 }
12398 
12399 </script>
12400 <details>
12401 <!-- Shading_WorkCodeSpan -->
12402 <!-- //</span>
12403 //<-- ===== Work -->
12404 
12405 <!-- ===== Work { -->
12406 //<span id="Charmap_WorkCodeSpan">
12407 <!--
12408 <details>
12409 <!--
12410 <summary>Character Map</summary>
12411 <!--
12412 <!--
12413 <!--
12414 <!--
12415 <!--
12416 <!--
12417 <!--
12418 <!--
12419 <style>
12420 #charmap_1_Frame {
12421     overflow:scroll;
12422     height:320px; width:320px;
12423     xtransform:scale(0.5);
12424     border:1px solid black;
12425     resize:both;
12426     background-color:#fff;
12427 }
12428 .Charmap {
12429     xfont-size:10.25;
12430     font-size:16px;
12431     line-height:1.0;
12432     xfont-family:Georgia;
12433     color:#000;
12434 }
12435 </style>
12436 <script>
12437 function charmagpen(){
12438     text = '';
12439     for( cc = 0; cc < 0x10000; cc++ ){
12440         text += String.fromCharCode(cc);
12441     }
12442     Charmap_1_Text.innerHTML = text;
12443 }
12444 Charmap_Work.addEventListener('click',charmagpen);
12445 //charmagpen();
12446 </script>
12447 
12448 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12449 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12450 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12451 <span id="Charmap_WorkCodeView"></span>
12452 <script id="Charmap_WorkScript">
12453 function Charmap_OpenWorkCodeView(){
12454     function Charmap_ShowWorkCode(){
12455         showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12456     }
12457     Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_ShowWorkCode);
12458 }
12459 Charmap_OpenWorkCodeView(); // should be invoked by an event
12460 </script>
12461 </details>
12462 <!-- Charmap_WorkCodeSpan -->
12463 <!-- //</span>
12464 <!-- ===== Work -->
12465 
12466 
12467 <!-- ===== Work { -->
12468 //<span id="Pointillism_WorkCodeSpan">
12469 <!--
12470 <details>
12471 <!--
12472 <summary>Collaborated Pointillism</summary>
12473 <!--
12474 <!--
12475 <!--
12476 <!--
12477 <!--
12478 <!--
12479 <!--
12480 <!--
12481 <!--
12482 <!--
12483 <!--
12484 <!--
12485 <!--
12486 <!--
12487 <!--
12488 <!--
12489 <!--
12490 <!--
12491 <!--
12492 <!--
12493 <!--
12494 <!--
12495 <!--
12496 <!--
12497 <!--
12498 <!--
12499 <!--
12500 <!--
12501 <!--
12502 <!--
12503 <!--
12504 <!--
12505 <!--
12506 <!--
12507 <!--
12508 <!--
12509 <!--
12510 <!--
12511 <!--
12512 <!--
12513 <!--
12514 <!--
12515 <!--
12516 <!--
12517 <!--
12518 <!--
12519 <!--
12520 <!--
12521 <!--
12522 <!--
12523 <!--
12524 <!--
12525 <!--
12526 <!--
12527 <!--
12528 <!--
12529 <!--
12530 <!--
12531 <!--
12532 <!--
12533 <!--
12534 <!--
12535 <!--
12536 <!--
12537 <!--
12538 <!--
12539 <!--
12540 <!--
12541 <!--
12542 <!--
12543 <!--
12544 <!--
12545 <!--
12546 <!--
12547 <!--
12548 <!--
12549 <!--
12550 <!--
12551 <!--
12552 <!--
12553 <!--
12554 <!--
12555 <!--
12556 <!--
12557 <!--
12558 <!--
12559 <!--
12560 <!--
12561 <!--
12562 <!--
12563 <!--
12564 <!--
12565 <!--
12566 <!--
12567 <!--
12568 <!--
12569 <!--
12570 <!--
12571 <!--
12572 <!--
12573 <!--
12574 <!--
12575 <!--
12576 <!--
12577 <!--
12578 <!--
12579 <!--
12580 <!--
12581 <!--
12582 <!--
12583 <!--
12584 <!--
12585 <!--
12586 <!--
12587 <!--
12588 <!--
12589 <!--
12590 <!--
12591 <!--
12592 <!--
12593 <!--
12594 <!--
12595 <!--
12596 <!--
12597 <!--
12598 <!--
12599 <!--
12600 <!--
12601 <!--
12602 <!--
12603 <!--
12604 <!--
12605 <!--
12606 <!--
12607 <!--
12608 <!--
12609 <!--
12610 <!--
12611 <!--
12612 <!--
12613 <!--
12614 <!--
12615 <!--
12616 <!--
12617 <!--
12618 <!--
12619 <!--
12620 <!--
12621 <!--
12622 <!--
12623 <!--
12624 <!--
12625 <!--
12626 <!--
12627 <!--
12628 <!--
12629 <!--
12630 <!--
12631 <!--
12632 <!--
12633 <!--
12634 <!--
12635 <!--
12636 <!--
12637 <!--
12638 <!--
12639 <!--
12640 <!--
12641 <!--
12642 <!--
12643 <!--
12644 <!--
12645 <!--
12646 <!--
12647 <!--
12648 <!--
12649 <!--
12650 <!--
12651 <!--
12652 <!--
12653 <!--
12654 <!--
12655 <!--
12656 <!--
12657 <!--
12658 <!--
12659 <!--
12660 <!--
12661 <!--
12662 <!--
12663 <!--
12664 <!--
12665 <!--
12666 <!--
12667 <!--
12668 <!--
12669 <!--
12670 <!--
12671 <!--
12672 <!--
12673 <!--
12674 <!--
12675 <!--
12676 <!--
12677 <!--
12678 <!--
12679 <!--
12680 <!--
12681 <!--
12682 <!--
12683 <!--
12684 <!--
12685 <!--
12686 <!--
12687 <!--
12688 <!--
12689 <!--
12690 <!--
12691 <!--
12692 <!--
12693 <!--
12694 <!--
12695 <!--
12696 <!--
12697 <!--
12698 <!--
12699 <!--
12700 <!--
12701 <!--
12702 <!--
12703 <!--
12704 <!--
12705 <!--
12706 <!--
12707 <!--
12708 <!--
12709 <!--
12710 <!--
12711 <!--
12712 <!--
12713 <!--
12714 <!--
12715 <!--
12716 <!--
12717 <!--
12718 <!--
12719 <!--
12720 <!--
12721 <!--
12722 <!--
12723 <!--
12724 <!--
12725 <!--
12726 <!--
12727 <!--
12728 <!--
12729 <!--
12730 <!--
12731 <!--
12732 <!--
12733 <!--
12734 <!--
12735 <!--
12736 <!--
12737 <!--
12738 <!--
12739 <!--
12740 <!--
12741 <!--
12742 <!--
12743 <!--
12744 <!--
12745 <!--
12746 <!--
12747 <!--
12748 <!--
12749 <!--
12750 <!--
12751 <!--
12752 <!--
12753 <!--
12754 <!--
12755 <!--
12756 <!--
12757 <!--
12758 <!--
12759 <!--
12760 <!--
12761 <!--
12762 <!--
12763 <!--
12764 <!--
12765 <!--
12766 <!--
12767 <!--
12768 <!--
12769 <!--
12770 <!--
12771 <!--
12772 <!--
12773 <!--
12774 <!--
12775 <!--
12776 <!--
12777 <!--
12778 <!--
12779 <!--
12780 <!--
12781 <!--
12782 <!--
12783 <!--
12784 <!--
12785 <!--
12786 <!--
12787 <!--
12788 <!--
12789 <!--
12790 <!--
12791 <!--
12792 <!--
12793 <!--
12794 <!--
12795 <!--
12796 <!--
12797 <!--
12798 <!--
12799 <!--
12800 <!--
12801 <!--
12802 <!--
12803 <!--
12804 <!--
12805 <!--
12806 <!--
12807 <!--
12808 <!--
12809 <!--
12810 <!--
12811 <!--
12812 <!--
12813 <!--
12814 <!--
12815 <!--
12816 <!--
12817 <!--
12818 <!--
12819 <!--
12820 <!--
12821 <!--
12822 <!--
12823 <!--
12824 <!--
12825 <!--
12826 <!--
12827 <!--
12828 <!--
12829 <!--
12830 <!--
12831 <!--
12832 <!--
12833 <!--
12834 <!--
12835 <!--
12836 <!--
12837 <!--
12838 <!--
12839 <!--
12840 <!--
12841 <!--
12842 <!--
12843 <!--
12844 <!--
12845 <!--
12846 <!--
12847 <!--
12848 <!--
12849 <!--
12850 <!--
12851 <!--
12852 <!--
12853 <!--
12854 <!--
12855 <!--
12856 <!--
12857 <!--
12858 <!--
12859 <!--
12860 <!--
12861 <!--
12862 <!--
12863 <!--
12864 <!--
12865 <!--
12866 <!--
12867 <!--
12868 <!--
12869 <!--
12870 <!--
12871 <!--
12872 <!--
12873 <!--
12874 <!--
12875 <!--
12876 <!--
12877 <!--
12878 <!--
12879 <!--
12880 <!--
12881 <!--
12882 <!--
12883 <!--
12884 <!--
12885 <!--
12886 <!--
12887 <!--
12888 <!--
12889 <!--
12890 <!--
12891 <!--
12892 <!--
12893 <!--
12894 <!--
12895 <!--
12896 <!--
12897 <!--
12898 <!--
12899 <!--
12900 <!--
12901 <!--
12902 <!--
12903 <!--
12904 <!--
12905 <!--
12906 <!--
12907 <!--
12908 <!--
12909 <!--
12910 <!--
12911 <!--
12912 <!--
12913 <!--
12914 <!--
12915 <!--
12916 <!--
12917 <!--
12918 <!--
12919 <!--
12920 <!--
12921 <!--
12922 <!--
12923 <!--
12924 <!--
12925 <!--
12926 <!--
12927 <!--
12928 <!--
12929 <!--
12930 <!--
12931 <!--
12932 <!--
12933 <!--
12934 <!--
12935 <!--
12936 <!--
12937 <!--
12938 <!--
12939 <!--
12940 <!--
12941 <!--
12942 <!--
12943 <!--
12944 <!--
12945 <!--
12946 <!--
12947 <!--
12948 <!--
12949 <!--
12950 <!--
12951 <!--
12952 <!--
12953 <!--
12954 <!--
12955 <!--
12956 <!--
12957 <!--
12958 <!--
12959 <!--
12960 <!--
12961 <!--
12962 <!--
12963 <!--
12964 <!--
12965 <!--
12966 <!--
12967 <!--
12968 <!--
12969 <!--
12970 <!--
12971 <!--
12972 <!--
12973 <!--
12974 <!--
12975 <!--
12976 <!--
12977 <!--
12978 <!--
12979 <!--
12980 <!--
12981 <!--
12982 <!--
12983 <!--
12984 <!--
12985 <!--
12986 <!--
12987 <!--
12988 <!--
12989 <!--
12990 <!--
12991 <!--
12992 <!--
12993 <!--
12994 <!--
12995 <!--
12996 <!--
12997 <!--
12998 <!--
12999 <!--
13000 <!--
13001 <!--
13002 <!--
13003 <!--
13004 <!--
13005 <!--
13006 <!--
13007 <!--
13008 <!--
13009 <!--
13010 <!--
13011 <!--
13012 <!--
13013 <!--
13014 <!--
13015 <!--
13016 <!--
13017 <!--
13018 <!--
13019 <!--
13020 <!--
13021 <!--
13022 <!--
13023 <!--
13024 <!--
13025 <!--
13026 <!--
13027 <!--
13028 <!--
13029 <!--
13030 <!--
13031 <!--
13032 <!--
13033 <!--
13034 <!--
13035 <!--
13036 <!--
13037 <!--
13038 <!--
13039 <!--
13040 <!--
13041 <!--
13042 <!--
13043 <!--
13044 <!--
13045 <!--
13046 <!--
13047 <!--
13048 <!--
13049 <!--
13050 <!--
13051 <!--
13052 <!--
13053 <!--
13054 <!--
13055 <!--
13056 <!--
13057 <!--
13058 <!--
13059 <!--
13060 <!--
13061 <!--
13062 <!--
13063 <!--
13064 <!--
13065 <!--
13066 <!--
13067 <!--
13068 <!--
13069 <!--
13070 <!--
13071 <!--
13072 <!--
13073 <!--
13074 <!--
13075 <!--
13076 <!--
13077 <!--
13078 <!--
13079 <!--
13080 <!--
13081 <!--
13082 <!--
13083 <!--
13084 <!--
13085 <!--
13086 <!--
13087 <!--
13088 <!--
13089 <!--
13090 <!--
13091 <!--
13092 <!--
13093 <!--
13094 <!--
13095 <!--
13096 <!--
13097 <!--
13098 <!--
13099 <!--
13100 <!--
13101 <!--
13102 <!--
13103 <!--
13104 <!--
13105 <!--
13106 <!--
13107 <!--
13108 <!--
13109 <!--
13110 <!--
13111 <!--
13112 <!--
13113 <!--
13114 <!--
13115 <!--
13116 <!--
13117 <!--
13118 <!--
13119 <!--
13120 <!--
13121 <!--
13122 <!--
13123 <!--
13124 <!--
13125 <!--
13126 <!--
13127 <!--
13128 <!--
13129 <!--
13130 <!--
13131 <!--
13132 <!--
13133 <!--
13134 <!--
13135 <!--
13136 <!--
13137 <!--
13138 <!--
13139 <!--
13140 <!--
13141 <!--
13142 <!--
13143 <!--
13144 <!--
13145 <!--
13146 <!--
13147 <!--
13148 <!--
13149 <!--
13150 <!--
13151 <!--
13152 <!--
13153 <!--
13154 <!--
13155 <!--
13156 <!--
13157 <!--
13158 <!--
13159 <!--
13160 <!--
13161 <!--
13162 <!--
13163 <!--
13164 <!--
13165 <!--
13166 <!--
13167 <!--
13168 <!--
13169 <!--
13170 <!--
13171 <!--
13172 <!--
13173 <!--
13174 <!--
13175 <!--
13176 <!--
13177 <!--
13178 <!--
13179 <!--
13180 <!--
13181 <!--
13182 <!--
13183 <!--
13184 <!--
13185 <!--
13186 <!--
13187 <!--
13188 <!--
13189 <!--
13190 <!--
13191 <!--
13192 <!--
13193 <!--
13194 <!--
13195 <!--
13196 <!--
13197 <!--
13198 <!--
13199 <!--
13200 <!--
13201 <!--
13202 <!--
13203 <!--
13204 <!--
13205 <!--
13206 <!--
13207 <!--
13208 <!--
13209 <!--
13210 <!--
13211 <!--
13212 <!--
13213 <!--
13214 <!--
13215 <!--
13216 <!--
13217 <!--
13218 <!--
13219 <!--
13220 <!--
13221 <!--
13222 <!--
13223 <!--
13224 <!--
13225 <!--
13226 <!--
13227 <!--
13228 <!--
13229 <!--
13230 <!--
13231 <!--
13232 <!--
13233 <!--
13234 <!--
13235 <!--
13236 <!--
13237 <!--
13238 <!--
13239 <!--
13240 <!--
13241 <!--
13242 <!--
13243 <!--
13244 <!--
13245 <!--
13246 <!--
13247 <!--
13248 <!--
13249 <!--
13250 <!--
13251 <!--
13252 <!--
13253 <!--
13254 <!--
13255 <!--
13256 <!--
13257 <!--
13258 <!--
13259 <!--
13260 <!--
13261 <!--
13262 <!--
13263 <!--
13264 <!--
13265 <!--
13266 <!--
13267 <!--
13268 <!--
13269 <!--
13270 <!--
13271 <!--
13272 <!--
13273 <!--
13274 <!--
13275 <!--
13276 <!--
13277 <!--
13278 <!--
13279 <!--
13280 <!--
13281 <!--
13282 <!--
13283 <!--
13284 <!--
13285 <!--
13286 <!--
13287 <!--
13288 <!--
13289 <!--
13290 <!--
13291 <!--
13292 <!--
13293 <!--
13294 <!--
13295 <!--
13296 <!--
13297 <!--
13298 <!--
13299 <!--
13300 <!--
13301 <!--
13302 <!--
13303 <!--
13304 <!--
13305 <!--
13306 <!--
13307 <!--
13308 <!--
13309 <!--
13310 <!--
13311 <!--
13312 <!--
13313 <!--
13314 <!--
13315 <!--
13316 <!--
13317 <!--
13318 <!--
13319 <!--
13320 <!--
13321 <!--
13322 <!--
13323 <!--
13324 <!--
13325 <!--
13326 <!--
13327 <!--
13328 <!--
13329 <!--
13330 <!--
13331 <!--
13332 <!--
13333 <!--
13334 <!--
13335 <!--
13336 <!--
13337 <!--
13338 <!--
```

```

12474<input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
12475<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
12476<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
12477<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
12478<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
12479<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Save">
12480<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Load">
12481<div id="Pointillism_1" class="Pointillism">
12482</div>
12483<span id="Pointillism_1_Unit_1" class="Pointillism_XY">
12484<span id="Pointillism_1_XY_1" class="Pointillism_XY_X1">
12485<span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
12486<canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12487</span>
12488<span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
12489<span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
12490<span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
12491<span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
12492<canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12493</span>
12494</div>
12495<br>
12496
12497<style>
12498.Pointillism {
12499    display:block;
12500    resize:both;
12501    width:680px;
12502    height:380px;
12503    min-height:240px;
12504    max-height:70px;
12505    background-color:#eee;
12506    overflow:scroll;
12507    font-size:16px;
12508    font-family:Georgia;
12509    color:#000;
12510    vertical-align:middle;
12511}
12512.Pointillism_Unit {
12513    position:relative;
12514    top:0px;
12515    display:block;
12516    overflow:scroll;
12517    width:300px;
12518    height:350px;
12519    margin:5px;
12520    padding:10px;
12521    background-color:rgba(255,255,127,0.7);
12522}
12523.Pointillism_XY {
12524    display:block;
12525    vertical-align:middle;
12526    width:290px;
12527    height:290px;
12528    font-size:12px;
12529    line-height:1.2;
12530    padding:5px;
12531    margin:0px;
12532    color:#fff;
12533    background-color:#44c;
12534}
12535.Pointillism_XY_Remote {
12536    display:block;
12537    vertical-align:middle;
12538    width:290px;
12539    height:290px;
12540    font-size:12px;
12541    line-height:1.2;
12542    padding:5px;
12543    margin:0px;
12544    color:#fff;
12545    background-color:#4a4;
12546}
12547.Pointillism_Canvas {
12548    display:block;
12549    position:relative;
12550    padding:0px;
12551    xleft:20px;
12552    xtop:20px;
12553    background-color:#333;
12554}
12555</style>
12556<script>
12557    var points = [];
12558    var replay = [];
12559    var replayx = 0;
12560    function pClearCanvas(can){
12561        ctx = can.getContext('2d');
12562        ctx.clearRect(0,0,can.width,can.height);
12563    }
12564    function Pointillism_1_ClearCanvas(){
12565        pClearCanvas(Pointillism_1_Canvas_1);
12566        pClearCanvas(Pointillism_1_Canvas_2);
12567    }
12568    function PointsReset(){
12569        points = [];
12570        replay = [];
12571        inRepeat = false;
12572        inReplay = false;
12573        Pointillism_1_ClearCanvas();
12574    }
12575    function Pointillism_1_ResetCanvas(){
12576        PointsReset();
12577        if( Pointillism_1_Share.checked ){
12578            //alert('--broad cast reset\n');
12579            GJ_BroadcastMessage('DRAW RESET');
12580        }
12581    }
12582    function Pointillism_1_ResetCanvasReceive(){
12583        //alert('--received reset\n');
12584        PointsReset();
12585    }
12586    function drawPoint(can,x,y,c,g){
12587        const ctx = can.getContext('2d');
12588        ctx.fillStyle = `rgba(${r},${g},${b},0.7)`;
12589        ctx.fillRect(x,y,8,8);
12590    }
12591    function waitMs(serno,ms){
12592        console.log(`-- wait ${#}#${serno}` + ms + 'ms');
12593        until = new Date();
12594        now = until.getTime();
12595        untilMs = now + ms;
12596        for( wi = 0; wi < untilMs; ){
12597            now = new Date();
12598            nowMs = now.getTime();
12599            remMs = untilMs - nowMs;
12600            //console.log(`wait ${#}#${wi}` + remMs + '/' + ms);
12601            if( remMs < 0 ) {
12602                break;
12603            }
12604        }
12605    }
12606    var inReplay = false;
12607    function replay1(){
12608        rx = replayx;
12609        if( replay.length <= rx ){
12610            return;
12611        }
12612        replayx += 1;
12613        pl = replay[rx];
12614        if( pl[1] == 1 ){
12615            can = Pointillism_1_Canvas_1;
12616        }else{
12617            can = Pointillism_1_Canvas_2;
12618        }
12619        drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
12620        if( inReplay == false ){
12621            console.log(`wait ${#}#${replayx}` + ' Stopped');
12622            return;
12623        }
12624        if( rx < replay.length-1 ){
12625            prevMs = replay[rx][0].getTime();
12626            nextMs = replay[rx+1][0].getTime();
12627            delayMs = nextMs - prevMs;
12628            //console.log(`wait ${#}#${replayx}` + delayMs + 'ms');
12629            window.setTimeout(replay1,delayMs);
12630        }else{
12631            console.log(`wait ${#}#${replayx}` + ' Finished');
12632            if( inRepeat ){
12633                window.setTimeout(replay1,1000);
12634            }
12635        }
    }

```

```

12636}
12637function Pointillism_1_ReplayCanvas(can){
12638    Pointillism_1_ClearCanvas();
12639    replay = points;
12640    replayx = 0;
12641    inReplay = true;
12642    replay1();
12643}
12644var inRepeat = false;
12645function repeat1(){
12646    Pointillism_1_ClearCanvas();
12647    replay = points;
12648    replayx = 0;
12649    replay1();
12650    if(inRepeat ){
12651        //window.setTimeout(repeat1,1000);
12652    }
12653}
12654function Pointillism_1_RepeatCanvas(can){
12655    if(inRepeat ){
12656        inRepeat = false;
12657        inReplay = false;
12658    }else{
12659        inRepeat = true;
12660        inReplay = true;
12661        repeat1();
12662    }
12663}
12664function CopyLocal(){ return Pointillism_1_Share.checked == false; }
12665function Pointillism_Setup(){
12666    var moveCount1 = 0;
12667    var moveCount2 = 0;
12668}
12669var gjlinked = false;
12670function GJdraw(msg){
12671    if( gjlinked == false ){
12672        if(ask.Section.open == true;
12673            GJ_Join();
12674            gjlinked = true;
12675        }
12676    }
12677    GJ_BcastMessage('DRAW '+msg);
12678}
12679function showXY1(e){
12680    moveCount1 += 1;
12681    x = e.offsetX;
12682    y = e.offsetY;
12683    Pointillism_1_XY_1_Remote.innerHTML = 'XY1: '+ 'x='+x +' , y='+y +' /'+moveCount1+'/'+points.length;
12684    Pointillism_1_XY_2_Remote.innerHTML = 'XY1: '+ 'x='+x +' , y='+y +' /'+moveCount1;
12685    if( e.buttons [ ] || CopyLocal() ){
12686        points.push(new Date(),x,y,64,64,255);
12687        drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
12688        if( CopyLocal() ){
12689            drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
12690        }
12691        GJdraw('1,'+x+','+y);
12692    }
12693}
12694function showXY2(e){
12695    moveCount2 += 1;
12696    x = e.offsetX;
12697    y = e.offsetY;
12698    Pointillism_1_XY_2_innerHTML = 'XY2: '+ 'x='+x +' , y='+y +' /'+moveCount2+'/'+points.length;
12699    Pointillism_1_XY_1_Remote.innerHTML = 'XY2: '+ 'x='+x +' , y='+y +' /'+moveCount1;
12700    if( e.buttons [ ] || CopyLocal() ){
12701        points.push(new Date(),x,y,64,255,64);
12702        drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
12703        if( CopyLocal() ){
12704            drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
12705        }
12706        //GJdraw('2,'+x+','+y);
12707    }
12708}
12709Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
12710Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);
12711Pointillism_1_Unit_2.style.left = '340px';
12712Pointillism_1_Unit_2.style.top = '-375px';
12713}
12714function Pointillism_RemoteDraw(arg){
12715    //alert('Draw at '+arg);
12716    //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
12717    if( arg == 'RESET' ){
12718        Pointillism_1_ResetCanvasReceive();
12719    }else{
12720        arg = arg.split(',');
12721        x = argv[1];
12722        y = argv[2];
12723        Pointillism_1_XY_2_Remote.innerHTML = 'XYR: '+ 'x='+x +' , y='+y +' /'+points.length;
12724        drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
12725    }
12726}
12727</script>
12728
12729<input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeviewButton" type="button" value="ShowCode">
12730<input id="Pointillism_WorkOpenSnapshot" class="HtmlCodeviewButton" type="button" value="Snapshot">
12731<input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12732<span id="Pointillism_WorkCodeView"></span>
12733<script id="Pointillism_WorkScript">
12734function Pointillism_OpenWorkCodeView(){
12735    function Pointillism_ShowWorkCode(){
12736        showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
12737    }
12738    Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_ShowWorkCode);
12739}
12740Pointillism_OpenWorkCodeView(); // should be invoked by an event
12741</script>
12742<details>
12743<summary>Pointillism_WorkCodeSpan </summary>
12744<!-- Pointillism_WorkCodeSpan -->
12745<!-- /<span>
12746//<!-- ===== Work ===== -->
12747
12748
12749//<!-- ===== Work ===== -->
12750<span id="StatCounter_WorkCodeSpan">
12751</span>
12752/*
12753<details><summary>StatCounter</summary>
12754<!-- StatCounter--> 2020-1010 SatoxITS -->
12755<h2>StatCounter</h2>
12756
12757<div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
12758    (counter as image tag)</div>
12759<style>
12760    .statcounter {
12761        vertical-align:middle;
12762    }
12763    #SatoxITS {
12764        color:#000;
12765        font-size:12pt;
12766        height:30px;
12767        width:100px;
12768        background-color:#ddd;
12769    }
12770</style>
12771<div>
12772<script>
12773    var sc_project=12411639;
12774    var sc_invisible=0;
12775    var sc_security="laeb2a3a";
12776    var sc_js_host="https://";
12777    var scJsHost = "https://";
12778</script>
12779<!-- script src="https://statcounter.com/counter/counter.js" -->
12780<!-- /script --> (counter by inline script)
12781</div>
12782
12783<input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12784<input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeviewButton" type="button" value="Snapshot">
12785<input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12786<span id="StatCounter_WorkCodeView"></span>
12787<script id="StatCounter_WorkScript">
12788function StatCounter_openWorkCodeView(){
12789    function StatCounter_showWorkCode(){
12790        showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
12791    }
12792    StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
12793    StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
12794}
12795StatCounter_OpenWorkCodeView(); // should be invoked by an event
12796</script>

```

```

12798</details>
12799<!-- StatCounter_WorkCodeSpan -->
12800 //</span>
12801//<!-- ===== Work -->
12802
12803
12804
12805
12806
12807
12808//<!-- ===== Work { ===== -->
12809//<span id="Template_WorkCodeSpan">
12810/*
12811<details><summary>Work Template</summary>
12812<!-- Template of Work-->
12813<h2>Template of Work</h2>
12814<input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12815<input id="Template_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12816<input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12817<span id="Template_WorkCodeView"></span>
12818<script id="Template_WorkCodeView">
12819 function Template_OpenWorkCodeView(){
12820     function Template_ShowWorkCode(){
12821         showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
12822     }
12823     Template_WorkCodeViewOpen.addEventListener('click',Template_ShowWorkCode);
12824 }
12825 Template_OpenWorkCodeView(); // should be invoked by an event
12826</script>
12827</details>
12828<!-- Template_WorkCodeSpan -->
12829/* </span>
12830//<!-- ===== Work -->
12831
12832
12833//<!-- ===== Work { ===== -->
12834//<span id="OriginalSource_WorkCodeSpan">
12835/*
12836<details open=""><summary>Original Source</summary>
12837<!-- Original Source // 2020-10-09 SatoxITS -->
12838<h2>Original Source of GShell</h2>
12839<input id="OriginalSource_ViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12840<input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12841<input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12842<span id="OriginalSource_TextElement"></span>
12843<span id="OriginalSource_WorkCodeView"></span>
12844<script id="OriginalSource_WorkScript">
12845 function OriginalSource_OpenWorkCodeView(){
12846     function OriginalSource_ShowWorkCode(){
12847         //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
12848         //OriginalSourceTextElement = OriginalSourceNode;
12849         //console.log('src3\n'+OriginalSourceNode.outerHTML);
12850         showHtmlCode(OriginalSource_WorkCodeView,OriginalSourceNode,
12851             /*\n',
12852             '\n',true);
12853     }
12854     OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_ShowWorkCode);
12855 }
12856//OriginalSourceNode = document.documentElement.cloneNode();
12857//OriginalSourceNode = gsh.cloneNode(true); //=====
12858//console.log('src0\n'+document.documentElement.outerHTML);
12859//console.log('src1\n'+gsh.outerHTML);
12860//console.log('src2\n'+OriginalSourceNode.innerHTML);
12861 OriginalSource_OpenWorkCodeView(); // should be invoked by an event
12862//showCode_asHTMLSource(OriginalSource_WorkCodeView,OriginalSourceNode);
12863function OriginalSource_Node(){
12864    if( false ){
12865        m0 = performance.memory;
12866        m00 = m0.usedJSHeapSize;
12867        console.log('-- heap bef clone: '
12868            +'m0.usedJSHeapSize+'\'+m0.totalHeapSizeLimit);
12869    }
12870    OriginalSourceNode = gsh.cloneNode(true);
12871    if( false ){
12872        m1 = performance.memory;
12873        m11 = m1.usedJSHeapSize;
12874        m10 = m10.usedJSHeapSize;
12875        //alert('--- one: used heap '+m00+' -> '+m11+' = '+m10+' bytes');
12876        console.log('-- heap aft clone: '
12877            +'m1.usedJSHeapSize+'\'+m1.totalHeapSize+'\'+m1.jsHeapSizeLimit);
12878    }
12879    //OriginalSourceNode = document.documentElement.cloneNode(true);
12880 }
12881
12882function Gsh_SetupPage(){
12883    GshSetImages();
12884    /Indexer_afterLoaded();
12885    /GshAfterTopKeyCommands();
12886    //Console_init();
12887    GJConsole_initFactory();
12888    GJLink_init();
12889    InterFrameComm_init();
12890    Gshell_InitTopbar();
12891    /#/MainTopBarTop.init();
12892    Indexer_init();
12893//    Aff_Setup();
12894//    Shading_Setup();
12895    window.setInterval(ShowResourceUsage,1000);
12896    //document.addEventListener('keydown',jgshCommand); // should be applied later?
12897    Pointillism_Setup();
12898    Pointillism_Setup();
12899    Footer_Setup();
12900    showFooter();
12901    GshInsideIconSetup();
12902    Sightglass_Setup();
12903}
12904function OnLoad(){
12905    SaveOriginalNode();
12906    Gsh_SetupPage();
12907}
12908document.addEventListener('load',Gsh_SetupPage);
12909</script>
12910</details>
12911<!-- OriginalSource_WorkCodeSpan -->
12912/* </span>
12913//<!-- ===== Work -->
12914
12915
12916
12917//</div>
12918//<b><script>OnLoad();</script></b>
12919

```