

```

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--0.7.5--2020-10-24--SatoxITS</span>
7 <title id="GshTitle">Gshell-0.7.5 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 0.7.5 // 2020-10-24 // SatoxITS</note></div>
17 </div>
18 </div>
19
20 <!-- Work { ----->
21 <span id="Topbar_WorkCodeSpan">
22 /*
23 <details><summary>Topbar</summary>
24 <!-- ----- Topbar // 2020-1008 SatoxITS { -->
25 <h2>Topbar</h2>
26 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
27 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
28 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
29 <span id="Topbar_WorkCodeView"></span>
30 </details>
31
32 <style>
33 #GshHeading {
34   display:inline;
35   overflow:visible;
36 }
37 .ConfigIcon {
38   position:absolute;
39   top:-6px;
40   left:92%;
41   width:32px;
42   height:32px;
43 }
44 .MetaWindow {
45   z-index:1000;
46   position:relative;
47   display:block;
48   overflow:visible !important;
49   width:99.9%;
50   height:22px;
51   top:-22px;
52   border:1px solid #22a;
53   margin:0px;
54   left:0.0%;
55   line-height:1.0;
56   font-family:Georgia;
57   color:#fff;
58   font-size:12pt;
59   text-align:center;
60   vertical-align:middle;
61   padding:4px;
62   xxxbackground-color:rgba(0,8,170,0.8);
63   background-color:#3a4861;xxx-PBlue;
64   vertical-align:middle;
65 }
66 .MetaWindow:hover {
67   color:#000;
68   border:1px solid #22a;
69   background-color:rgba(255,255,255,1.0);
70 }
71 #GshBanner {
72   overflow:visible;
73   display:block;
74   width:100%;
75   height:100px;
76   left:inherit !important;
77 }
78 </style>
79 <script>
80 function Topbar_openWorkCodeView(){
81   function Topbar_showWorkCode(){
82     showHtmlCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
83   }
84   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
85 }
86 Topbar_openWorkCodeView();
87 function ConfigClick(){
88   if( 0 <= AffView.style.zIndex ){
89     AffView.style.saved_zIndex = AffView.style.zIndex;
90     AffView.style.zIndex = -1000;
91     GshSidebar.style.zIndex = -1;
92     GshPerfMon.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('AffZidex='+AffView.style.zIndex);
101 }
102 function Gshell_initTopbar(){
103   GshTopbar.innerHTML = GshTitle.innerHTML;
104   </img id="ConfigIcon" class="ConfigIcon">
105   if( true ){
106     cfig = document.createElement('img');
107     cfig.id = 'ConfigIcon';
108     cfig.setAttribute('class','ConfigIcon');
109     GshTopbar.appendChild(cfig);
110     cfig.src = ConfigIcon_DATA;
111
112     //cfig.style.zIndex = 10000000000;
113     //cfig.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_WorkOpenSnapshot" 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   xxxoverflow:scroll !important;
137 }
138 #GshMain {
139   z-index:1;
140   position:relative;
141   display:block;
142   width:80% !important;
143   left:19.5% !important;
144 }
145 #GshSidebar {
146   z-index:0;
147   position:relative !important;
148   overflow:auto;
149   resize:both !important;
150   xxxoverflow:hidden !important;
151   xxxheight:100px !important;
152   xxxdisplay:inline !important;
153   left:0px;
154   top:0px;
155   width:19.5%;
156   min-width:80px;
157   xxxheight:100% !important;
158   height:0px;
159   color:#f00;
160   xxxbackground-color:rgba(64,64,64,0.5);

```

```

162   xxxbackground-color:#FFE3EB;xxx-PBlue;
163   background-color:#eeeeee;xxx-PBlue;
164 }
165 #GshPerfMon {
166   position:relative;
167   display:block;
168   overflow:visible;
169   z-index:0 !important;
170   width:120px;
171   font-family:monospace, Courier New !important;
172   font-size:9pt !important;
173   color:#F34;
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:100%;
190   left:0px;
191   top:0px;
192   scroll-behavior: overflow !important;
193   padding-left:4px;
194   font-size:0.5em;
195   white-space:nowrap;
196   xxx-background-color:rgba(64,160,64,0.6) !important;
197   color:#794c6;xxx-PBlue;
198   xxxbackground-color:#FFE3EB;xxx-PBlue;
199   background-color:#eeeeee;xxx-PBlue;
200 }
201 #GshIndexer:hover {
202   z-index:1000000;
203   overflow-x:visible !important;
204   color:#000000 !important;xxx-PBlue;
205   xxxbackground-color:#FFFFFF;xxx-PBlue;
206   background-color:rgba(255,255,255,0.7);
207   padding-right:0px;
208   width:80%;
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:#eff1f5;xxx-PBlue;
220   xxx-color:#41516d;xxx-PBlue;
221   xxx-color:#794c6;xxx-PBlue;
222   padding-right:4px;
223 }
224 .IndexLine:hover {
225   font-size:10pt !important;
226   xxx-color:#228;
227   xxx-background-color:#fff;
228   xxxcolor:#fff;xxx-PBlue;
229   color:#516487;xxx-PBlue;
230   background-color:rgba(220,220,255,1.0);xxx-PBlue;
231   xxxtext-shadow:1px 1px #f3f3;
232   text-shadow:1px 1px #eee;
233   xxxbackground-color:#516487;xxx-PBlue;
234   xxxtext-decoration:underline !important;
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   d = new Date();
252   perf = '';
253   perf += '<!--font color="gray">UA: ' + window.navigator.userAgent + '<!--<br><br>\n';
254   perf += DateShort0(startPerfDate) + '<br>\n';
255   perf += DateShort() + '<br>\n';
256   elps = d.getTime() - startPerfDate.getTime();
257   itvl = d.getTime() - prevPerfDate.getTime();
258   perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
259   perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
260   prevPerfDate = d;
261
262   if( performance.memory !== undefined ){
263     m0 = performance.memory;
264     mu0 = (m0.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+'/' +m0.totalHeapSize+'/' +m0.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       //
280       GshPerfMon.style.position = 'absolute';
281       GshPerfMon.style.display = 'block';
282       GshPerfMon.style.marginLeft = '4px';
283       //GshPerfMon.style.top = '45px';
284       GshPerfMon.style.position = 'relative';
285       //GshPerfMon.style.position = 'absolute';
286       //topy = GshTopbar.getBoundingClientRect().top;
287       //topy = parseInt(topy) + 40;
288       //GshPerfMon.style.top = topy + '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,chn,nch,ht){
302   // https://developer.mozilla.org/en-US/docs/Web/API/Element
303   c = '';
304   if( e.classList != null ){
305     c = e.classList.value;
306   }
307   //console.log('-- '<!--e.nodeName>' + '<!--e.id' + '<!--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   }else
316   if( e.nodeName == 'SUMMARY' ){
317     id = e.innerHTML;
318     GeneratedId += 1;
319     eid = 'GeneratedId-'+GeneratedId;
320     e.id = eid;
321   }else
322   if( and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxentry-content' )){
323     console.log('-- DIV entry-content begin');

```

```

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="GeneratedEref'+iserno+'" class="IndexLine" href="'+eid+'>'
338 + iserno + ' ' + 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 += ' (' + nch + ') ' + '</div>';
345 for (let i = 0; i < chv.length; i++) {
346     sec = ni + ' ' + i;
347     if (ni == ' ') { sec = i; }
348     ht += generateIndex(sec, chv[i], null, 0);
349 }
350 return ht;
351 }
352 function onClickIndex(e) {
353     tid = e.target.id;
354     tge = document.getElementById(tid);
355     eid = tge.getAttribute('href');
356     rx = tge.getBoundingClientRect().left.toFixed(0);
357     ry = tge.getBoundingClientRect().top.toFixed(0);
358     if (false) {
359         alert('index clicked mouse(x="' + e.x + '", y="' + e.y + '")'
360             + '\ntid=' + tid + ' ' + rx + 'x' + ' ' + ry + 'y'
361             + '\neid=' + eid + '\n'
362             + '\nhtml=' + tge.outerHTML);
363     }
364     ee = document.getElementById(eid);
365     sx = 'NaN';
366     sy = ee.getBoundingClientRect().top;
367     console.log('sx=' + sx + ', sy=' + sy);
368     ee.scrollTo(sx, sy);
369     window.scrollTo(sx, sy);
370     //window.scrollTo({left:'NaN',top:sy,behavior:'smooth'});
371 }
372 function Indexer_afterLoaded() {
373     sideindex = document.getElementById('GshIndexer');
374     ht = '<h3>G-Indexer</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 = '-1400px';
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 = '1400px';
402         gsh.appendChild(GshSidebar); // change parent
403     } else {
404         console.log('-- Seems not in WordPress');
405         GshSidebar.style.setProperty('position', 'fixed', 'important');
406     }
407 }
408 //document.addEventListener('load', Indexer_afterLoaded);
409
410 DestroyIndexer = function() {
411     sideindex = document.getElementById('GshIndexer');
412     sideindex.innerHTML = '';
413     sideindex.style = '';
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. --SatoxITS("-")
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="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
435 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
436 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
437 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
438 <span class="GshMenu" id="GshMenuWin" onclick="win_jump('0.1');">0</span>
439 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
440 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
441 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
442 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
443 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
444 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="">Source</span>
445 <!-- | <span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span> -->
446 </span>
447 </div>
448 </div>
449 </div>
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><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><p>
465 This 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><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, SatoxITS (satoxits-more.jp)</address>
476 </details>
477 </div>
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 </div>
485 <h3>Vi compatible command line editor</h3>

```

```

486 <g>
487 The command line of GShell can be edited with commands compatible with
488 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
489 As in vi, you can enter <i><b>command mode</b></i> by <b>ESC</b> key,
490 then move around in the history by <b>code</b> k / ? n W</code></b>,
491 or within the current line by <b>code</b> l h f v b o $ </code></b> or so.
492 </p>
493 </details>
494 */
495 /*
496 <details id="gsh-gindex">
497 <summary>index</summary><div class="gsh-src">
498 Documents
499 <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">import</a>
503 <a href="#struct">struct</a>
504 Main functions
505 <a href="#comexpansion">str-expansion</a> // macro processor
506 <a href="#finder">finder</a> // builtin find + du
507 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
508 <a href="#plugin">plugin</a> // plugin commands
509 <a href="#ex-commands">system</a> // external commands
510 <a href="#builtin">builtin</a> // builtin commands
511 <a href="#network">network</a> // socket handler
512 <a href="#remote-sh">remote-sh</a> // remote shell
513 <a href="#redirect">redirect</a> // Stdin/Out redirection
514 <a href="#history">history</a> // command history
515 <a href="#usage">usage</a> // resource usage
516 <a href="#encode">encode</a> // encode / decode
517 <a href="#IME">IME</a> // command line IME
518 <a href="#getline">getline</a> // line editor
519 <a href="#scanf">scanf</a> // string decomposer
520 <a href="#interpreter">interpreter</a> // command interpreter
521 <a href="#main">main</a>
522 </span>
523 JavaScript part
524 <a href="#script-src-view" class="gsh-link" onclick="jumpTo_JavaScriptView();">Source</a>
525 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpTo_DataView();">Builtin data</a>
526 CSS part
527 <a href="#style-src-view" class="gsh-link" onclick="jumpTo_StyleView();">Source</a>
528 References
529 <a href="#g" class="gsh-link" onclick="jumpTo_WholeView();">Internal</a>
530 <a href="#gsh-reference" class="gsh-link" onclick="jumpTo_ReferenceView();">External</a>
531 Whole parts
532 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Source</a>
533 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Download</a>
534 <a href="#whole-src-view" class="gsh-link" onclick="jumpTo_WholeView();">Dump</a>
535
536 </div>
537 </details>
538 */
539 <<details id="gsh-gocode">
540 <<summary>Go Source</summary><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 SatouITS (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" // <a href="https://golang.org/pkg/errors/">errors</a>
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     "html" // <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/go/types/">types</a>
565     "go/token" // <a href="https://golang.org/pkg/go/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 #ifdef WIN32
577 #include <windows.h> // </windows.h>
578 // 2020-1022 added -- terminal mode on Windows
579 // https://docs.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         DWORD xmode = tmode;
586         xmode &= -ENABLE_ECHO_INPUT;
587         xmode &= -ENABLE_LINE_INPUT;
588         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
589         if( SetConsoleMode(hStdin, xmode) ){
590             return tmode;
591         }
592     }
593     return 0;
594 }
595 #endif
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 #include "poll.h" // <poll.h> // </poll.h> to be closed as HTML tag :-p
616 typedef struct { struct pollfd fdv[8]; } pollfdv;
617 int pollx(pollfdv *fdv, int nfds, int timeout){
618     return poll(fdv->fdv, nfdv, timeout);
619 }
620 import "C"
621
622 // 2020-1021 replaced poll() with channel/select
623 // 2020-0906 added,
624 func CPollin(fp*os.File, timeoutsUs int)(ready uintptr){
625     var fdv = C.pollfdv{}
626     var nfdv = 1
627     var timeout = timeoutUs/1000
628     fdv.fdv[0].fd = C.int(fp.Fd())
629     fdv.fdv[0].events = C.POLLIN
630     if( 0 < EventRecvFD ){
631         fdv.fdv[1].fd = C.int(EventRecvFD)
632         fdv.fdv[1].events = C.POLLIN
633         nfdv += 1
634     }
635     r := C.pollx(&fdv, C.int(nfdv), C.int(timeout))
636     if( r <= 0 ){
637         return 0
638     }
639     if( int(fdv.fdv[1].revents) & int(C.POLLIN) != 0 {
640         //fprintf(stderr, "--de- got Event\n");
641         return uintptr(EventFdOffset + fdv.fdv[1].fd)
642     }
643     if( int(fdv.fdv[0].revents) & int(C.POLLIN) != 0 {
644         return uintptr(NormalFdOffset + fdv.fdv[0].fd)
645     }
646     return 0
647 }

```

```

648 }
649 */
650
651 const {
652     NAME = "gsh"
653     VERSION = "0.7.5"
654     DATE = "2020-10-24"
655     AUTHOR = "SatoxITS("-")/"
656 }
657 var {
658     GSH_HOME = ".gsh" // under home directory
659     GSH_PORT = 9999
660     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
661     PROMPT = ">"
662     LINESIZE = (8*1024)
663     PATHSEP = "/" // should be ";" in Windows
664     DIRSEP = "/" // canbe \ in Windows
665     OnWindows = false;
666 }
667 func initGshEnv(){
668     if( runtime.GOOS == "windows" ){
669         PATHSEP = ";";
670         DIRSEP = "\\";
671         OnWindows = true;
672     }else{
673     }
674 }
675
676 // -XX logging control
677 // --A- all
678 // --I- info.
679 // --D- debug
680 // --T- time and resource usage
681 // --W- warning
682 // --E- error
683 // --F- fatal error
684 // --Xn- network
685
686 // <a name="struct">Structures</a>
687
688 // 2020-1022 Unix/Windows
689 // -----
690 //type aStat_t syscall.Stat_t;
691 //type aStat_t struct { syscall.Stat_t }
692 type aStat_t struct {
693     Size int64
694     Mode os.FileMode
695     Rdev int64
696     Blocks int64
697     Nlink int64
698 }
699 func aLstat(path string, astat *aStat_t)(error){
700     /*
701     sstat := syscall.Stat_t{};
702     err := syscall.Lstat(path,&sstat);
703     *astat = astat_t(sstat);
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
713 func aFstat(fd int, astat *aStat_t)(error){
714     /*
715     sstat := syscall.Stat_t{};
716     err := syscall.Fstat(fd,&sstat);
717     *astat = astat_t(sstat);
718     */
719     err := errors.New("NotImplemented-Fstat");
720     //fmt.Printf("----E-- fstat(%v)\n",fd,err);
721     return err;
722 }
723
724 func aAccess(path string, mode uint32)(error){
725     //err := syscall.Access(path,mode);
726     //err := errors.New("NotImplemented-Access");
727     fi,err := os.Stat(path);
728     //fmt.Printf("--- Access(%v,%v)\n",path,mode,err);
729     if( err == nil ){
730         fmode := fi.Mode();
731         if( fmode.IsRegular() ){
732             perm := fmode.Perm();
733             if( (uint32(perm) & mode) != 0 ){
734                 return nil;
735             }
736             return errors.New("NotAccessible");
737         }
738         return errors.New("NotRegularFile");
739     }
740     return err;
741 }
742 // 2020-1022 Unix/Windows
743 // -----
744 type aRusage struct {
745     syscall.Rusage
746     Utime 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)*100000000+int64(sysru.Utime.Usec)*1000);
761     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*100000000+int64(sysru.Stime.Usec)*1000);
762     */
763 }
764
765 func aSetrusage(ru *aRusage, ps *os.ProcessState){
766     ru.Utime = ps.Utime();
767     ru.Stime = ps.SystemTime();
768 }
769
770 func showRusage(what string,argv []string, ru *aRusage){
771     fmt.Printf("%s:",what);
772     //fmt.Printf("Sys=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
773     //fmt.Printf("Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
774     //fmt.Printf("User=%d.%06ds",ru.Utime/1000000000,(ru.Utime/1000)%10000000);
775     //fmt.Printf("Sys=%d.%06ds",ru.Stime/1000000000,(ru.Stime/1000)%10000000);
776
777     fmt.Printf(" Rss=%vB",ru.Maxrss)
778     if isin("-l",argv) {
779         fmt.Printf(" MinFlt=%v",ru.Minflt)
780         fmt.Printf(" MajFlt=%v",ru.Majflt)
781         fmt.Printf(" IxRSS=%vB",ru.Ixrss)
782         fmt.Printf(" IDRSS=%vB",ru.Idrss)
783         fmt.Printf(" Nswap=%vB",ru.Nswap)
784         fmt.Printf(" Read=%v",ru.Inblock)
785         fmt.Printf(" Write=%v",ru.Obblock)
786     }
787     fmt.Printf(" Snd=%v",ru.Msgsnd)
788     fmt.Printf(" Rcv=%v",ru.Msgrcv)
789     //if isin("-l",argv) {
790         //fmt.Printf(" Sig=%v",ru.Nsignals)
791     }
792     //
793     fmt.Printf("\n");
794 }
795
796 type GCommandHistory struct {
797     StartAt time.Time // command line execution started at
798     EndAt time.Time // command line execution ended at
799     ResCode int // exit code of (external command)
800     CmdError error // error string
801     OutData *os.File // output of the command
802     FoundFile []string // output - result of ufind
803     Rusagev [2]aRusage // Resource consumption, CPU time or so
804     Cmdid int // maybe with identified with arguments or impact
805     // redirection commands should not be the Cmdid
806     WorkDir string // working directory at start
807     WorkDirX int // index in ChdirHistory
808     CmdLine string // command line
809 }
810 type GChdirHistory 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 var Events []Event
825 type PluginInfo 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 = 0x000001 // items count
839     SUM_SIZE = 0x000002 // data length (simply added)
840     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)
841     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
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     SUM_WORDS = 0x000010 // word count is a kind of digest
845     SUM_LINES = 0x000020 // line count is a kind of digest
846     SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
847
848     SUM_SUM32_BITS = 0x000100 // the number of true bits
849     SUM_SUM32_2BYTE = 0x000200 // 16bits words
850     SUM_SUM32_4BYTE = 0x000400 // 32bits words
851     SUM_SUM32_8BYTE = 0x000800 // 64bits words
852
853     SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bed
854     SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
855     SUM_UNIXFILE = 0x004000
856     SUM_CRCIEEE = 0x008000
857 )
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 int
869     CTime time.Time
870     ATime time.Time
871     MTime time.Time
872     Start time.Time
873     Done time.Time
874     RusgAtStart [2]aRusage
875     RusgAtEnd [2]aRusage
876 }
877 type ValueStack []string
878 type GshContext struct {
879     StartDir string // the current directory at the start
880     GetLine string // gsh-getline command as a input line editor
881     ChdirHistory []GchdirHistory // the 1st entry is wd at the start
882     //gshPA syscall.ProcAttr
883     gshPA os.ProcAttr
884     CommandHistory []GCommandHistory
885     CmdCurrent GCommandHistory
886     Background bool
887     BackgroundJobs []os.ProcessState; //[]int
888     LastRusage aRusage
889     GshHomeDir string
890     TerminalId int
891     CmdTrace bool // should be [map]
892     CmdTime bool // should be [map]
893     PluginFuncs []PluginInfo
894     iValues []string
895     iDelimiter string // field sepearator of print out
896     iFormat string // default print format (of integer)
897     iValStack ValueStack
898     LastServer GServer
899     RSERVER string // [gsh://]host[:port]
900     RWD 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     nsleep(ns*1000)
909 }
910 func mSleep(ns time.Duration){
911     nsleep(ns*1000000)
912 }
913 func sleep(ns time.Duration){
914     nsleep(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
943 func env(opts []string) {
944     env := os.Environ()
945     if isin("-s", opts){
946         sort.Slice(env, func(i,j int) bool {
947             return env[i] < env[j]
948         })
949     }
950     for _, v := range env {
951         fmt.Printf("%v\n",v)
952     }
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, leng 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         ival, _ := strconv.Atoi(str[0:leng])
969         return ival, 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] == '|' {
981         hi = histlen - 1
982     } else {
983         leng = 1
984     }
985     hi, leng = scanInt(str[i+1:])
986     if leng == 0 {
987         return 0, rstr
988     }
989     if hi < 0 {
990         hi = histlen + hi
991     }
992     if 0 <= hi && hi < histlen {
993         var ext byte
994         if 1 < len(str[i+leng:]) {
995             ext = str[i+leng:][1]
996         }
997         //fmt.Printf("--D-- %v(%c)\n", str[i+leng:], str[i+leng:])
998         if ext == 'f' {
999             leng += 1
1000             xlist := []string{}
1001             list := gshCtx.CommandHistory[hi].FoundFile
1002             for _, v := range list {
1003                 //list[i] = escapeWhiteSP(v)
1004                 xlist = append(xlist, escapeWhiteSP(v))
1005             }
1006             //rstr += strings.Join(list, " ")
1007             rstr += strings.Join(xlist, " ")
1008         } else {
1009             if ext == 'd' || ext == 'd' {
1010                 // INH .. workdir at the start of the command
1011                 leng = 1
1012                 rstr += gshCtx.CommandHistory[hi].WorkDir
1013             } else {
1014                 rstr += gshCtx.CommandHistory[hi].CmdLine
1015             }
1016         }
1017         leng = 0
1018     }
1019     return leng, rstr
1020 }
1021 func escapeWhiteSP(str string)(string){
1022     if len(str) == 0 {
1023         return "\\r" // empty, to be ignored
1024     }
1025     rstr := ""
1026     for _, ch := range str {
1027         switch ch {
1028             case '\\': rstr += "\\\\"
1029             case ' ': rstr += "\\s"
1030             case '\t': rstr += "\\t"
1031             case '\r': rstr += "\\r"
1032             case '\n': rstr += "\\n"
1033             default: rstr += string(ch)
1034         }
1035     }
1036     return rstr
1037 }
1038 func unescapeWhiteSP(str string)(string){ // strip original escapes
1039     rstr := ""
1040     for i := 0; i < len(str); i++ {
1041         ch := str[i]
1042         if ch == '\\' {
1043             if i+1 < len(str) {
1044                 switch str[i+1] {
1045                     case 'r':
1046                         continue;
1047                 }
1048             }
1049             rstr += string(ch)
1050         }
1051     }
1052     return rstr
1053 }
1054 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
1055     ustrv := []string{}
1056     for _, v := range strv {
1057         ustrv = append(ustrv, unescapeWhiteSP(v))
1058     }
1059     return ustrv
1060 }
1061 // <a name="comexpansion">str-expansion</a>
1062 // - this should be a macro processor
1063 func strsubst(gshCtx *GshContext, str string, histonly bool) string {
1064     rbuff := []byte{}
1065     if false {
1066         //@@@U Unicode should be cared as a character
1067         return str
1068     }
1069     //rstr := ""
1070     inEsc := 0 // escape characer mode
1071     for i := 0; i < len(str); i++ {
1072         //fmt.Printf("--D--Subst %v%v\n", i, str[i:])
1073         ch := str[i]
1074         if inEsc == 0 {
1075             if ch == '|' {
1076                 //leng, xrstr := substHistory(gshCtx, str, i, rstr)
1077                 leng, rs := substHistory(gshCtx, str, i, "")
1078                 if 0 < leng {
1079                     //_, rs := substHistory(gshCtx, str, i, "")
1080                     rbuff = append(rbuff, []byte(rs)...)
1081                     i += leng
1082                     //rstr = xrstr
1083                     continue
1084                 }
1085             }
1086             switch ch {
1087                 case '\\': inEsc = '\\'; continue
1088                 //case '$': inEsc = '$'; continue
1089                 case '$':
1090             }
1091         }
1092         switch inEsc {
1093             case '\\':
1094                 switch ch {
1095                     case '\\': ch = '\\'
1096                     case 's': ch = ' '
1097                     case 't': ch = '\t'
1098                     case 'r': ch = '\r'
1099                     case 'n': ch = '\n'
1100                     case 'z': inEsc = 0; continue // empty, to be ignored
1101                 }
1102             case '$':
1103                 inEsc = 0
1104                 switch {
1105                     case ch == '$': ch = '$'
1106                     case ch == 'T':
1107                         //rstr = rstr + time.Now().Format(time.Stamp)
1108                         rs := time.Now().Format(time.Stamp)
1109                         rbuff = append(rbuff, []byte(rs)...)
1110                         inEsc = 0
1111                         continue;
1112                     default:
1113                         // postpone the interpretation
1114                         //rstr = rstr + "$" + string(ch)
1115                         rbuff = append(rbuff, ch)
1116                         inEsc = 0
1117                         continue;
1118                 }
1119             case 0:
1120                 inEsc = 0
1121             }
1122         //rstr = rstr + string(ch)
1123         rbuff = append(rbuff, ch)
1124     }
1125     //fmt.Printf("--D--subst(%s)(%s)\n", str, string(rbuff))
1126     return string(rbuff)
1127     //return rstr
1128 }
1129 func showFileInfo(path string, opts []string) {
1130     if isin("-l", opts) || isin("-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 %8v %8s ",fi.Mode(),fi.Size(),date)
1138     }
1139     }
1140     fmt.Printf("%s",path)
1141     if !isln("-sp",opts) {
1142         fmt.Printf(" ")
1143     }else
1144     if !isln("-n",opts) {
1145         fmt.Printf("\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.Printf("---I-- HOME=%v(kv)\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 {
1166     case pathv[0] == ".":
1167         pathv[0],_ = os.Getwd()
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] == "-":
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
1210 func (gshCtx *GshContext)Dec(argv[]string){
1211     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1212     li := 0
1213     buff := make([]byte,LINESIZE)
1214     for li = 0; li++ {
1215         count, err := decoder.Read(buff)
1216         if count <= 0 {
1217             break
1218         }
1219         if err != nil {
1220             break
1221         }
1222         os.Stdout.Write(buff[0:count])
1223     }
1224 }
1225
1226 // lnsip [N] [-crif][-c \\\
1227 func (gshCtx *GshContext)SplitLine(argv[]string){
1228     strRep := isln("-str",argv) // "...+"
1229     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1230     ni := 0
1231     toi := 0
1232     for ni = 0; ni++ {
1233         line, err := reader.ReadString('\n')
1234         if len(line) <= 0 {
1235             if err != nil {
1236                 fmt.Fprintf(os.Stderr,"--I-- lnsip %d to %d (%v)\n",ni,toi,err)
1237                 break
1238             }
1239             off := 0
1240             ilen := len(line)
1241             remlen := len(line)
1242             if strRep { os.Stdout.Write([]byte("\n")) }
1243             for oi = 0; oi < remlen; oi++ {
1244                 olen := remlen
1245                 addnl := false
1246                 if 72 < olen {
1247                     olen = 72
1248                     addnl = true
1249                 }
1250                 fmt.Fprintf(os.Stderr,"--D-- write %d (%d.%d) %d %d/%d\n",
1251                     toi,ni,oi,off,olen,remlen,ilen)
1252                 os.Stdout.Write([]byte(line[oi:olen]))
1253                 if addnl {
1254                     if strRep {
1255                         os.Stdout.Write([]byte("\n\n"))
1256                     }else{
1257                         //os.Stdout.Write([]byte("\r\n"))
1258                         os.Stdout.Write([]byte("\n"))
1259                         os.Stdout.Write([]byte("\n"))
1260                     }
1261                 }
1262                 line = line[olen:]
1263                 off += olen
1264                 remlen -= olen
1265             }
1266             if strRep { os.Stdout.Write([]byte("\n")) }
1267         }
1268         fmt.Fprintf(os.Stderr,"--I-- lnsip %d to %d\n",ni,toi)
1269     }
1270 }
1271
1272 // CRC32 <a href="http://golang.jp/pkg/hash/crc32">crc32</a>
1273 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1274 var CRC32UNIX uint32 = uint32(0x04c11db7) // Unix cksum
1275 var CRC32IEEE uint32 = uint32(0xedb88320)
1276 func byteCRC32add(crc uint32,str[]byte,len uint64)(uint32){
1277     var oi uint64
1278     for oi = 0; oi < len; oi++ {
1279         var oct = str[oi]
1280         for bi = 0; bi < 8; bi++ {
1281             //fprintf(stderr,"--CRC32 %d %X (%d.%d)\n",crc,oct,oi,bi)
1282             ovf1 := (crc & 0x80000000) != 0
1283             ovf2 := (oct & 0x80) != 0
1284             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1285             oct <<= 1
1286             crc <<= 1
1287             if ovf { crc ^= CRC32UNIX }
1288         }
1289     }
1290     //fprintf(stderr,"--CRC32 return %d %d\n",crc,len)
1291     return crc;
1292 }
1293
1294 func byteCRC32end(crc uint32, len uint64)(uint32){
1295     var olen = make([]byte,4)
1296     var li = 0
1297     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(str string,len uint64)(crc uint32){
1308     crc = byteCRC32add(0,[]byte(str),len)
1309     crc = byteCRC32end(crc,len)
1310     //fmt.Printf(stderr,"--CRC32 %d %d\n",crc,len)
1311     return crc
1312 }
1313 func CRC32Finish(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)xChecksum(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(stderr,"--E-- cksam %v (%v)\n",path,err)
1339         return -1
1340     }
1341     defer file.Close()
1342     if gsh.CmdTrace { fmt.Printf("--I-- cksam %v %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_CRCIEEE) != 0 {
1367         sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
1368     }
1369     // <a href="https://en.wikipedia.org/wiki/BSD_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_SVSV) != 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 "lmp" for "(text) line processor" or "scanner"
1398 // a,t,tab,c, - sequential combination of patterns
1399 // what "LINE" is should be definable
1400 // generic line-by-line processing
1401 // grep [-v]
1402 // cat -n -v
1403 // uniq [-c]
1404 // tail -f
1405 // sed s/x/y/ or awk
1406 // grep with line count like wc
1407 // rewrite contents if specified
1408 func (gsh*GshContext)xGrep(path string,rxpv[]string)(int){
1409     file, err := os.OpenFile(path,os.O_RDONLY,0)
1410     if err != nil {
1411         fmt.Printf(stderr,"--E-- grep %v (%v)\n",path,err)
1412         return -1
1413     }
1414     defer file.Close()
1415     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path,rxpv) }
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),rxpv[0]) {
1433             found += 1
1434             fmt.Printf("%s:%d: %s",path,li,line)
1435         }
1436     }
1437     //fmt.Printf("total %d lines %s\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 %x fmt list
1446 // ls -R
1447 // tar command by adding output
1448 type fileSum 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     DupBlocks int64 // Blocks pointed from hard links
1454     HLinks int64 // hard links
1455     Words int64
1456     Lines int64
1457     Files int64

```

```

1458 Dirs      int64 // the num. of directories
1459 SymLink   int64
1460 Flats     int64 // the num. of flat files
1461 MaxDepth  int64
1462 MaxNameLen int64 // max. name length
1463 nextRepo  time.Time
1464 }
1465 func showFusage(dir string, fusage *fileSum) {
1466     bsume := float64(((fusage.Blocks - fusage.DupBlocks)/2)*1024)/1000000.0
1467     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1468     fmt.Printf("tv: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1469         dir,
1470         fusage.Files,
1471         fusage.Dirs,
1472         fusage.SymLink,
1473         fusage.HLinks,
1474         float64(fusage.Size)/1000000.0, bsume);
1475 }
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 cumFinfo(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             showFusage(tstamp, fsum)
1491         }
1492         fsum.nextRepo = now.Add(time.Second)
1493     }
1494     if staterr != nil {
1495         fsum.Err += 1
1496         return fsum
1497     }
1498     fsum.Files += 1
1499     if l < fstat.Nlink {
1500         // must count only once...
1501         // at least ignore ones in the same directory
1502         //if finfo.Mode().IsRegular() {
1503             if (fstat.Mode & S_IFMT) == S_IFREG {
1504                 fsum.HLinks += 1
1505                 fsum.DupBlocks += int64(fstat.Blocks)
1506                 //fmt.Printf("---Dup HardLink %v %v\n", fstat.Nlink, path)
1507             }
1508         }
1509         //fsum.Size += finfo.Size()
1510         fsum.Size += fstat.Size
1511         fsum.Blocks += int64(fstat.Blocks)
1512         //if verb { fmt.Printf("%8dBlk %s", fstat.Blocks/2, path) }
1513         if isin("-ls", argv) {
1514             //if verb { fmt.Printf("%4d %8d ", fstat.Blksize, fstat.Blocks) }
1515             //fmt.Printf("%8d\t", fstat.Blocks/2)
1516         }
1517         //if finfo.IsDir()
1518         if (fstat.Mode & S_IFMT) == S_IFDIR {
1519             fsum.Dirs += 1
1520         }
1521         //if (finfo.Mode() & os.ModeSymlink) != 0
1522         if (fstat.Mode & S_IFMT) == S_IFLNK {
1523             //if verb { fmt.Printf("symlink(%v, %v)\n", fstat.Mode, finfo.Name()) }
1524             //if verb { fmt.Printf("symlink(%s, %s)\n", fstat.Mode, finfo.Name()) }
1525             fsum.SymLink += 1
1526         }
1527         return fsum
1528     }
1529 func (gsh*GshContext)xxFindEntv(depth int, total *fileSum, dir string, dstat aStat_t, ei int, entv []string, npatv []string, argv[]string)(*fileSum){
1530     nols := isin("-grep", argv)
1531     // sort entv
1532     /*
1533     if isin("-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     /*
1540     if isin("-u", argv){
1541         sort.Slice(filev, func(i, j int) bool {
1542             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1543         })
1544     }
1545     if isin("-U", argv){
1546         sort.Slice(filev, func(i, j int) bool {
1547             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1548         })
1549     }
1550     */
1551     /*
1552     if isin("-s", argv){
1553         sort.Slice(filev, func(i, j int) bool {
1554             return filev[j].Size() < filev[i].Size()
1555         })
1556     }
1557     */
1558     for _, filename := range entv {
1559         for _, npat := range npatv {
1560             match := true
1561             if npat == "*" {
1562                 match = true
1563             } else {
1564                 match, _ = filepath.Match(npat, filename)
1565             }
1566             path := dir + DIRSEP + filename
1567             if !match {
1568                 continue
1569             }
1570             var fstat aStat_t
1571             staterr := aStat(path, &fstat)
1572             if staterr != nil {
1573                 if !isin("-w", argv){fmt.Printf("ufind: %v\n", staterr) }
1574                 continue;
1575             }
1576             if isin("-du", argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1577                 // should not show size of directory in "-du" mode ...
1578             } else {
1579                 if !nols && !isin("-s", argv) && (!isin("-du", argv) || isin("-a", argv)) {
1580                     if isin("-du", argv) {
1581                         fmt.Printf("%8d\t", fstat.Blocks/2)
1582                     }
1583                     showFileInfo(path, argv)
1584                 }
1585                 if true { // && isin("-du", argv)
1586                     total = cumFinfo(total, path, staterr, fstat, argv, false)
1587                 }
1588             }
1589             if isin("-wc", argv) {
1590             }
1591             /*
1592             if gsh.lastCheckSum.SumType != 0 {
1593                 gsh.xChecksum(path, argv, &gsh.lastCheckSum);
1594             }
1595             x := isinK("-grep", argv); // -grep will be convenient like -ls
1596             if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1597                 if !isRegFile(path){
1598                     found := gsh.xGrep(path, argv[x+1:]);
1599                     if 0 < found {
1600                         foundv := gsh.CmdCurrent.FoundFile
1601                         if len(foundv) < 10 {
1602                             gsh.CmdCurrent.FoundFile =
1603                                 append(gsh.CmdCurrent.FoundFile, path)
1604                         }
1605                     }
1606                 }
1607             }
1608             if !isin("-r0", argv) { // -d 0 in du, -depth n in find
1609                 //total.Depth += 1
1610                 if (fstat.Mode & S_IFMT) == S_IFLNK {
1611                     continue
1612                 }
1613                 if dstat.Rdev != fstat.Rdev {
1614                     fmt.Printf("--i-- don't follow differnet device %v(%v) %v(%v)\n",
1615                         dir, dstat.Rdev, path, fstat.Rdev)
1616                 }
1617                 if (fstat.Mode & S_IFMT) == S_IFDIR {
1618                     total = gsh.xxFind(depth+1, total, path, npatv, argv)
1619                 }

```

```

1620     }
1621   }
1622 }
1623 return total
1624 }
1625 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npats[]string,argv[]string)(*fileSum){
1626   nols := isin("-grep",argv)
1627   dirfile,ocrr := os.OpenFile(dir,os.O_RDONLY,0)
1628   if ocrr == nil {
1629     //fmt.Printf("--I-- %v(%s)\n",dir,dirfile,dirfile.Fd())
1630     defer dirfile.Close()
1631   }else{
1632   }
1633
1634   prev := *total
1635   var dstat alstat_t
1636   staterr := alstat(dir,dstat) // should be flstat
1637
1638   if staterr != nil {
1639     if !isin("-x",argv){ fmt.Printf("ufind: %v\n",staterr) }
1640     return total
1641   }
1642   //filev,err := ioutil.ReadDir(dir)
1643   //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1644   /*
1645   if err != nil {
1646     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1647     return total
1648   }
1649   */
1650   if depth == 0 {
1651     total = cumFinfo(total,dir,staterr,dstat,argv,true)
1652     if !nols && !isin("-s",argv) && (!isin("-du",argv) || !isin("-a",argv)) {
1653       showFileinfo(dir,argv)
1654     }
1655   }
1656   // it is not a directory, just scan it and finish
1657
1658   for ei := 0; ; ei++ {
1659     entv,r derr := dirfile.Readdirnames(8*1024)
1660     if len(entv) == 0 || derr != nil {
1661       //if derr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),derr) }
1662       break
1663     }
1664     if 0 < ei {
1665       fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1666     }
1667     total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npats,argv)
1668   }
1669   if !isin("-du",argv) {
1670     // if in "du" mode
1671     fmt.Printf("%d\t%s\n",total.Blocks-prev.Blocks)/2,dir)
1672   }
1673   return total
1674 }
1675
1676 // {ufind[fulls] [Files] [-- Expressions]
1677 // Files is "." by default
1678 // Names is "*" by default
1679 // Expressions is "-print" by default for "ufind", or "-du" for "fu" command
1680 func (gsh*GshContext)xFind(argv[]string){
1681   if 0 < len(argv) && strbegins(argv[0],"?"){
1682     showFound(gsh,argv)
1683     return
1684   }
1685   if !isin("-cksum",argv) || !isin("-sum",argv) {
1686     gsh.lastCheckSum = CheckSum{}
1687     if !isin("-sum",argv) && !isin("-add",argv) {
1688       gsh.lastCheckSum.SumType |= SUM_SUM64
1689     }else
1690     if !isin("-sum",argv) && !isin("-size",argv) {
1691       gsh.lastCheckSum.SumType |= SUM_SIZE
1692     }else
1693     if !isin("-sum",argv) && !isin("-bsd",argv) {
1694       gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1695     }else
1696     if !isin("-sum",argv) && !isin("-sysv",argv) {
1697       gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1698     }else
1699     if !isin("-sum",argv) {
1700       gsh.lastCheckSum.SumType |= SUM_SUM64
1701     }
1702     if !isin("-unix",argv) {
1703       gsh.lastCheckSum.SumType |= SUM_UNIXFILE
1704       gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32UNIX)
1705     }
1706     if !isin("-ieee",argv){
1707       gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1708       gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32IEEE)
1709     }
1710     gsh.lastCheckSum.RusgAtStart = Getrusagev()
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   cwd := "."
1727   // if to be fullpath ::: 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(byteCrc32end(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     showFusage("total",fusage)
1766   }
1767   if !isin("-s",argv){
1768     hits = len(gsh.CmdCurrent.FoundFile)
1769     if 0 < hits {
1770       fmt.Printf("--I-- %d files hits // can be refered with %sdf\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.Done = time.Now()
1778       gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1779       elps := sum.Done.Sub(sum.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)
1781     }

```

```

1782     nanos := int64(elps)
1783     dnanos := time.Duration(nanos);
1784     fmt.Printf("--csum-time: %v/total, %v/file, %f files/s, %v\r\n",
1785         abbttime(dnanos),
1786         abbttime(time.Duration(nanos/sum.Files)),
1787         (float64(sum.Files)*100000000.0)/float64(nanos),
1788         abbspeed(sum.Size,nanos))
1789     diff := RusageSubv(sum.RusgAtEnd,sum.RusgAtStart)
1790     fmt.Printf("--csum-rusg: %v\n",sRusagef("",argv,diff))
1791 }
1792 }
1793 }
1794 }
1795 }
1796 }
1797 }
1798 }
1799 }
1800 }
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("%d (%d) ",i,len(v.FoundFile))
1807             if !isIn("-ls",argv){
1808                 fmt.Printf("\n")
1809                 for _,file := range v.FoundFile {
1810                     fmt.Printf("%s //sub number?")
1811                     showFileInfo(file,argv)
1812                 }
1813             }else{
1814                 showFiles(v.FoundFile)
1815                 fmt.Printf("\n")
1816             }
1817         }
1818     }
1819 }
1820 }
1821 func showMatchFile(filev []os.FileInfo, npat,dir string, argv []string)(string,bool){
1822     fname := ""
1823     found := false
1824     for _,v := range filev {
1825         match, _ := filepath.Match(npat,(v.Name()))
1826         if match {
1827             fname = v.Name()
1828             found = true
1829             //fmt.Printf("%d %s\n",i,v.Name())
1830             showIfExecutable(fname,dir,argv)
1831         }
1832     }
1833     return fname,found
1834 }
1835 }
1836 func showIfExecutable(name,dir string,argv []string)(ffullpath string,ffound bool){
1837     var fullpath string
1838     if strBegins(name,DIRSEP){
1839         fullpath = name
1840     }else{
1841         if len(dir) == 0 {
1842             fullpath = name;
1843         }else{
1844             fullpath = dir + DIRSEP + name
1845         }
1846     }
1847     fi, err := os.Stat(fullpath)
1848     //fmt.Printf("--Dp-- %v\n\n-- %v\n",fullpath,err);
1849     if err != nil {
1850         fullpath = ".exe";
1851         fi, err = os.Stat(fullpath)
1852     }
1853     if err != nil {
1854         fullpath = dir + DIRSEP + name + ".go"
1855         fi, err = os.Stat(fullpath)
1856     }
1857     if err == nil {
1858         fm := fi.Mode()
1859         if fm.IsRegular() {
1860             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1861             if aAccess(fullpath,5) == nil {
1862                 ffullpath = fullpath
1863                 ffound = true
1864                 if !isIn("-s", argv) {
1865                     showFileInfo(fullpath,argv)
1866                 }
1867             }
1868         }
1869     }
1870     return ffullpath, ffound
1871 }
1872 }
1873 }
1874 }
1875 }
1876 }
1877 }
1878 }
1879 }
1880 }
1881 }
1882 }
1883 }
1884 }
1885 }
1886 }
1887 }
1888 }
1889 }
1890 }
1891 }
1892 }
1893 }
1894 }
1895 }
1896 }
1897 }
1898 }
1899 }
1900 }
1901 }
1902 }
1903 }
1904 }
1905 }
1906 }
1907 }
1908 }
1909 }
1910 }
1911 }
1912 }
1913 }
1914 }
1915 }
1916 }
1917 }
1918 }
1919 }
1920 }
1921 }
1922 }
1923 }
1924 }
1925 }
1926 }
1927 }
1928 }
1929 }
1930 }
1931 }
1932 }
1933 }
1934 }
1935 }
1936 }
1937 }
1938 }
1939 }
1940 }
1941 }
1942 }
1943 }
1944 }

```

```

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 1 < ai {
1953             fmt.Printf(" ");
1954         }
1955         arg := argv[ai]
1956         found, val := getval(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 resF *File
1972 func resmap() {
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("ref could not open: %s\n",err)
1978     }else{
1979         fmt.Printf("ref 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 arg1 = 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,arg1[0:ax],str[si:])
2003             }
2004         }
2005         ch := str[si]
2006         if ch == '{' {
2007             inBracket += 1
2008             if 0 < strip && inBracket <= strip {
2009                 //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
2010                 continue
2011             }
2012         }
2013         if 0 < inBracket {
2014             if ch == '}' {
2015                 inBracket -= 1
2016                 if 0 < strip && inBracket < strip {
2017                     //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
2018                     continue
2019                 }
2020             }
2021             arg1[ax] = ch
2022             ax += 1
2023             continue
2024         }
2025         if str[si] == ' ' {
2026             argv = append(argv,string(arg1[0:ax]))
2027             if debug {
2028                 fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2029                     -1*len(argv),sb,si,str[sb:si],string(str[si:]))
2030             }
2031             sb = si+1
2032             ax = 0
2033             continue
2034         }
2035         arg1[ax] = ch
2036         ax += 1
2037     }
2038     if sb < si {
2039         argv = append(argv,string(arg1[0:ax]))
2040         if debug {
2041             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2042                 -1*len(argv),sb,si,string(arg1[0:ax]),string(str[si:]))
2043         }
2044     }
2045     if debug {
2046         fmt.Printf("--Da- %d [%s] => [%d]%v\n",strip,si,len(argv),argv)
2047     }
2048     return argv
2049 }
2050
2051 // should get stderr (into tmpfile ?) and return
2052 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
2053     //var pv = [1]int{-1}
2054     //syscall.Pipe(pv)
2055
2056     xarg := gshScanArg(name,1)
2057     name = strings.Join(xarg, " ")
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     fdix := 0
2064     dir := "r"
2065     if mode == "r" {
2066         dir = "<"
2067         fdix = 1 // read from the stdout of the process
2068     }else{
2069         dir = ">"
2070         fdix = 0 // write to the stdin of the process
2071     }
2072     gshPA := gsh.gshPA
2073     savfd := gshPA.Files[fdix]
2074
2075     var fd uintptr = 0
2076     if mode == "r" {
2077         //fd = pout.Fd()
2078         //gshPA.Files[fdix] = pout.Fd()
2079         gshPA.Files[fdix] = pout;
2080     }else{
2081         //fd = pin.Fd()
2082         //gshPA.Files[fdix] = pin.Fd()
2083         gshPA.Files[fdix] = pin;
2084     }
2085     // should do this by Goroutine?
2086     if false {
2087         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2088         fmt.Printf("--RDL [%d,%d,%d]->[%d,%d,%d]\n",
2089             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2090             pin.Fd(),pout.Fd(),pout.Fd())
2091     }
2092     savi := os.Stdin
2093     savo := os.Stdout
2094     save := os.Stderr
2095     os.Stdin = pin
2096     os.Stdout = pout
2097     os.Stderr = pout
2098     gsh.BackGround = true
2099     gsh.gshellih(name)
2100     gsh.BackGround = false
2101     os.Stdin = savi
2102     os.Stdout = savo
2103     os.Stderr = save
2104
2105     gshPA.Files[fdix] = savfd

```

```

2106     return pin,pout,false
2107 }
2108
2109 // <a name="ex-commands">External commands</a>
2110 func (gshCtx *GshContext)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,false
2117     }
2118     fullpath := fullpath[0]
2119     argv = unescapeWhiteSPV(argv)
2120     if 0 < strings.Index(fullpath,"go") {
2121         nargv := argv // []string{}
2122         gofullpathv, itis := which("PATH",[]string{"which","go","-s"})
2123         if itis == false {
2124             fmt.Printf("--P-- Go not found\n")
2125             return false,true
2126         }
2127         gofullpath := gofullpath[0]
2128         nargv = []string{gofullpath, "run", fullpath }
2129         fmt.Printf("--I-- %s %s %s\n",gofullpath,
2130             nargv[0],nargv[1],nargv[2])
2131         if exec {
2132             syscall.Exec(gofullpath,nargv,os.Environ())
2133         }else{
2134             //pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
2135             proc, _ := os.StartProcess(gofullpath,nargv,&gshPA);
2136             pstat, _ := proc.Wait();
2137             pid := pstat.Pid();
2138             if gsh.BackGround {
2139                 fmt.Fprintf(stderr,"--Ip- in Background pid[%d](%v)\n",pid,len(argv),nargv)
2140                 //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2141                 gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
2142             }else{
2143                 /*
2144                 rusage := rUsage {}
2145                 syscall.Wait4(pid,nil,0,&rusage)
2146                 gsh.LastRusage = rusage
2147                 gsh.CmdCurrent.Rusagev[1] = rusage
2148                 */
2149             /*
2150             gsh.LastRusage = *pstat.SysUsage().(*rUsage);
2151             gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*rUsage);
2152             */
2153             aSetRusage(&gsh.LastRusage,pstat);
2154             gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2155         }
2156     }
2157 }
2158 }else{
2159     if exec {
2160         syscall.Exec(fullpath,argv,os.Environ())
2161     }else{
2162         //pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
2163         proc, _ := os.StartProcess(fullpath,argv,&gshPA);
2164         pstat, _ := proc.Wait();
2165         pid := pstat.Pid();
2166         //fmt.Printf("[%d]\n",pid); // '%s' 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         }
2173         if gsh.BackGround {
2174             fmt.Fprintf(stderr,"--Ip- in Background pid[%d](%v)\n",pid,len(argv),argv)
2175             //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2176             gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
2177         }else{
2178             /*
2179             rusage := rUsage {}
2180             syscall.Wait4(pid,nil,0,&rusage);
2181             gsh.LastRusage = rusage
2182             gsh.CmdCurrent.Rusagev[1] = rusage
2183             */
2184             /*
2185             gsh.LastRusage = *pstat.SysUsage().(*rUsage);
2186             gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*rUsage);
2187             */
2188             aSetRusage(&gsh.LastRusage,pstat);
2189             gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2190         }
2191     }
2192 }
2193     return false,false
2194 }
2195
2196 // <a name="builtin">Builtin Commands</a>
2197 func (gshCtx *GshContext) sleep(argv []string) {
2198     if len(argv) < 2 {
2199         fmt.Printf("Sleep 100ms, 100us, 100ns, ...'\n")
2200         return
2201     }
2202     duration := argv[1];
2203     d, err := time.ParseDuration(duration)
2204     if err != nil {
2205         d, err = time.ParseDuration(duration+"s")
2206         if err != nil {
2207             fmt.Printf("duration ? %s (%s)\n",duration,err)
2208             return
2209         }
2210     }
2211     //fmt.Printf("Sleep %v\n",duration)
2212     time.Sleep(d)
2213     if 0 < len(argv[2:]) {
2214         gshCtx.gshellv(argv[2:])
2215     }
2216 }
2217 func (gshCtx *GshContext)repeat(argv []string) {
2218     if len(argv) < 2 {
2219         return
2220     }
2221     start0 := time.Now()
2222     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2223         if 0 < len(argv[2:]) {
2224             //start := time.Now()
2225             gshCtx.gshellv(argv[2:])
2226             end := time.Now()
2227             elps := end.Sub(start0);
2228             if( 1000000000 < elps ){
2229                 fmt.Printf("(repeat%d %v)\n",ri,elps);
2230             }
2231         }
2232     }
2233 }
2234
2235 func (gshCtx *GshContext)gen(argv []string) {
2236     gshPA := gshCtx.gshPA
2237     if len(argv) < 2 {
2238         fmt.Printf("Usage: %s H\n",argv[0])
2239         return
2240     }
2241     // should be repeated by "repeat" command
2242     count, _ := strconv.Atoi(argv[1])
2243     //fd := gshPA.Files[1] // Stdout
2244     //file := os.NewFile(fd,"internalStdout")
2245     file := gshPA.Files[1]; // Stdout
2246     fmt.Printf("--I-- Gen. Count=%d to [%d]\n",count,file.Fd())
2247     //buf := []byte{}
2248     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2249     for gi := 0; gi < count; gi++ {
2250         file.WriteString(outdata)
2251     }
2252     //file.WriteString("\n")
2253     fmt.Printf("\n(%d B)\n",count*len(outdata));
2254     //file.Close()
2255 }
2256
2257 // <a name="rexec">Remote Execution</a> // 2020-0820
2258 func Elapsed(from time.Time)(string){
2259     elps := time.Now().Sub(from)
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.%03dus]",elps/1000,(elps%1000))
2267         }
2268     }
2269 }

```

```

2268 }
2269 //func abftime(nanos int64)(string){
2270 func abftime(nanos time.Duration)(string){
2271     if 100000000 < nanos {
2272         return fmt.Sprintf("%d.%02ds", nanos/100000000, (nanos%100000000)/1000000)
2273     }
2274     else
2275     if 1000000 < nanos {
2276         return fmt.Sprintf("%d.%03dms", nanos/1000000, (nanos%1000000)/1000)
2277     }
2278     else{
2279         return fmt.Sprintf("%d.%03dus", nanos/1000, (nanos%1000))
2280     }
2281 }
2282 func absize(size int64)(string){
2283     fsize := float64(size)
2284     if 1024*1024*1024 < size {
2285         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
2286     }
2287     else
2288     if 1024*1024 < size {
2289         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
2290     }
2291     else{
2292         return fmt.Sprintf("%.3fKiB", fsize/1024)
2293     }
2294 }
2295 func absize(size int64)(string){
2296     fsize := float64(size)
2297     if 1024*1024*1024 < size {
2298         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
2299     }
2300     else
2301     if 1024*1024 < size {
2302         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
2303     }
2304     else{
2305         return fmt.Sprintf("%.3fKiB", fsize/1024)
2306     }
2307 }
2308 func abpspeed(totalB int64, ns int64)(string){
2309     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2310     if 1000 <= MBs {
2311         return fmt.Sprintf("%.3fGB/s", MBs/1000)
2312     }
2313     if 1 <= MBs {
2314         return fmt.Sprintf("%.3fMB/s", MBs)
2315     }
2316     else{
2317         return fmt.Sprintf("%.3fKB/s", MBs*1000)
2318     }
2319 }
2320 func abpspeed(totalB int64, ns time.Duration)(string){
2321     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2322     if 1000 <= MBs {
2323         return fmt.Sprintf("%.3fGBps", MBs/1000)
2324     }
2325     if 1 <= MBs {
2326         return fmt.Sprintf("%.3fMBps", MBs)
2327     }
2328     else{
2329         return fmt.Sprintf("%.3fKBps", MBs*1000)
2330     }
2331 }
2332 }
2333 func fileRelay(what string, in*os.File, out*os.File, size int64, bsiz int)(wcount int64){
2334     Start := time.Now()
2335     buff := make([]byte, bsiz)
2336     var total int64 = 0
2337     var rem int64 = size
2338     nio := 0
2339     Prev := time.Now()
2340     var PrevSize int64 = 0
2341     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) START\n",
2342         what, absize(total), size, nio)
2343     for i:= 0; ; i++ {
2344         var len = bsiz
2345         if int(rem) < len {
2346             len = int(rem)
2347         }
2348         Now := time.Now()
2349         Elps := Now.Sub(Prev);
2350         if 1000000000 <= Now.Sub(Prev) {
2351             fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %s\n",
2352                 what, absize(total), size, nio,
2353                 abpspeed((total-PrevSize), Elps))
2354             Prev = Now;
2355             PrevSize = total
2356         }
2357         rlen = len
2358         if in != nil {
2359             // should watch the disconnection of out
2360             rcc, err := in.Read(buff[0:rlen])
2361             if err != nil {
2362                 fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<v\n",
2363                     what, rcc, err, in.Name())
2364                 break
2365             }
2366             rlen = rcc
2367             if string(buff[0:10]) == "(SoftEOF " {
2368                 var ecc int64 = 0
2369                 fmt.Fprintf(string(buff), "((SoftEOF %v), %ecc)
2370                 fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
2371                     what, ecc, total)
2372                 if ecc == total {
2373                     break
2374                 }
2375             }
2376         }
2377         wlen := rlen
2378         if out != nil {
2379             wcc, err := out.Write(buff[0:rlen])
2380             if err != nil {
2381                 fmt.Printf(Elapsed(Start)+"--En-- X: %s write(%v,%v)>v\n",
2382                     what, wcc, err, out.Name())
2383                 break
2384             }
2385             wlen = wcc
2386         }
2387         if wlen < rlen {
2388             fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2389                 what, wlen, rlen)
2390             break;
2391         }
2392         nio += 1
2393         total += int64(rlen)
2394         rem -= int64(rlen)
2395         if rem <= 0 {
2396             break
2397         }
2398     }
2399     Done := time.Now()
2400     Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2401     TotalMB := float64(total)/1000000 //MB
2402     MBps := TotalMB / Elps
2403     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %v %.3fMB/s\n",
2404         what, total, size, nio, absize(total), MBps)
2405     return total
2406 }
2407 func tcpPush(clnt *os.File){
2408     // shrink socket buffer and recover
2409     usleep(100);
2410 }
2411 func (gsh*GshContext).RexecServer(argv[]string){
2412     debug := true
2413     Start0 := time.Now()
2414     Start := Start0
2415     // if local == ":" { local = "0.0.0.0:9999" }
2416     local := "0.0.0.0:9999"
2417     if 0 < len(argv) {
2418         if argv[0] == "-s" {
2419             debug = false
2420         }
2421         argv = argv[1:]
2422     }
2423     if 0 < len(argv) {
2424         argv = argv[1:]
2425     }
2426     port, err := net.ResolveTCPAddr("tcp", local);
2427     if err != nil {
2428         fmt.Printf("--En- S: Address error: %s (%s)\n", local, err)
2429         return
2430     }
2431     fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n", local);
2432     soconn, err := net.ListenTCP("tcp", port)
2433     if err != nil {

```

```

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

```



```

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
2609 req := ""
2610 res := make([]byte, LINESIZE)
2611 count, err := serv.Read(res)
2612 if err != nil {
2613     fmt.Printf("--En- S: (%d,%v) %v", count, err, string(res))
2614 }
2615 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res)) }
2616
2617 if argv[0] == "GET" {
2618     savPA := gsh.gshPA
2619     var bsize int = 64*1024
2620     req = fmt.Sprintf("%v\n", strings.Join(argv, " "))
2621     fmt.Printf(Elapsed(Start)+"--In- C: %v", req)
2622     fmt.Fprintf(serv, req)
2623     count, err = serv.Read(res)
2624     if err != nil {
2625         jelse{
2626             var dszize int64 = 0
2627             var out *os.File = nil
2628             var out_tobeclosed *os.File = nil
2629             var fname string = ""
2630             var rcode int = 0
2631             var pid int = -1
2632             fmt.Sscanf(string(res), "%d %d", &rcode, &dszize)
2633             fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res[0:count]))
2634             if 3 <= len(argv) {
2635                 fname = argv[2]
2636                 if strBegins(fname, "{") {
2637                     xin, xout, err := gsh.Popen(fname, "w")
2638                     if err {
2639                         jelse{
2640                             xin.Close()
2641                             defer xout.Close()
2642                             out = xout
2643                             out_tobeclosed = xout
2644                             pid = 0 // should be its pid
2645                         }
2646                     }
2647                     // should write to temporary file
2648                     // should suppress ^C on tty
2649                     xout, err := os.OpenFile(fname, os.O_CREATE|os.O_RDWR|os.O_TRUNC, 0600)
2650                     if err != nil {
2651                         fmt.Printf("--En- %v\n", err)
2652                     }
2653                     out = xout
2654                     //fmt.Printf("--In-- %d > %s\n", out.Fd(), fname)
2655                 }
2656             }
2657             in, _ := serv.File()
2658             fileRelay("RecvGET", in, out, dszize, bsize)
2659             if 0 <= pid {
2660                 gsh.gshPA = savPA // recovery of Fd(), and more?
2661                 fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n", fname)
2662                 out_tobeclosed.Close()
2663                 //syscall.Wait4(pid, nil, 0, nil) //##
2664             }
2665         }
2666     }
2667 }
2668 if argv[0] == "PUT" {
2669     remote, _ := serv.File()
2670     var local *os.File = nil
2671     var dszize int64 = 32*1024*1024
2672     var bsize int = 64*1024
2673     var ofile string = ""
2674     //fmt.Printf("--In- Rex %v\n", argv)
2675     if 1 <= len(argv) {
2676         fname := argv[1]
2677         if strBegins(fname, "-z") {
2678             fmt.Sscanf(fname[2:], "%d", &dszize)
2679         }
2680         jelse{
2681             if strBegins(fname, "{") {
2682                 xin, xout, err := gsh.Popen(fname, "r")
2683                 if err {
2684                     jelse{
2685                         xout.Close()
2686                         defer xin.Close()
2687                         //in = xin
2688                         local = xin
2689                         fmt.Printf("--In- [%d] < Upload output of %v\n",
2690                             local.Fd(), fname)
2691                         ofile = "-from."+fname
2692                         dszize = MaxStreamSize
2693                     }
2694                 }
2695             }
2696             xlocal, err := os.Open(fname)
2697             if err != nil {
2698                 fmt.Printf("--En- (%s)\n", err)
2699                 local = nil
2700             }
2701             jelse{
2702                 local = xlocal
2703                 fi, _ := local.Stat()
2704                 dszize = fi.Size()
2705                 defer local.Close()
2706                 //fmt.Printf("--In- Rex in(%v / %v)\n", ofile, dszize)
2707             }
2708             ofile = fname
2709             fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2710                 fname, dszize, local, err)
2711         }
2712     }
2713     if 2 <= len(argv) && argv[2] != "" {
2714         ofile = argv[2]
2715         //fmt.Printf("(%d)%v B.ofile=%v\n", len(argv), argv, ofile)
2716     }
2717     //fmt.Printf(Elapsed(Start)+"--In- Rex out(%v)\n", ofile)
2718     fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n", dszize, bsize)
2719     req = fmt.Sprintf("PUT %v %v\n", dszize, ofile)
2720     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v", req) }
2721     fmt.Fprintf(serv, req)
2722     count, err = serv.Read(res)
2723     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res[0:count])) }
2724     fileRelay("SendPUT", local, remote, dszize, bsize)
2725 }
2726 jelse{
2727     req = fmt.Sprintf("%v\n", strings.Join(argv, " "))
2728     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v", req) }
2729     fmt.Fprintf(serv, req)
2730     //fmt.Printf("--In- sending RexRequest(%v)\n", len(req))
2731 }
2732 //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2733 count, err = serv.Read(res)
2734 res := ""
2735 if count == 0 {
2736     res = "(nil)\r\n"
2737 }
2738 jelse{
2739     res = string(res[0:count])
2740 }
2741 if err != nil {
2742     fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v", count, err, res)
2743 }
2744 jelse{
2745     fmt.Printf(Elapsed(Start)+"--In- S: %v", res)
2746 }
2747 serv.Close()
2748 //conn.Close()
2749
2750 var stat string
2751 var rcode int
2752 fmt.Sscanf(res, "%d %s", &rcode, &stat)
2753 //fmt.Printf("--D- Client: %v (%v)", rcode, stat)
2754 return rcode, res
2755 }
2756
2757 // <a name="remote-sh">Remote Shell</a>
2758 // gop file .: { host:ip:port | dir } // -p | -no-p
2759 func (gsh*GshContext)FileCopy(argv[]string){
2760     var host = ""

```

```

2754 var port = ""
2755 var upload = false
2756 var download = false
2757 var xargv = []string{"rex-gcp"}
2758 var srcv = []string{}
2759 var dstv = []string{}
2760 argv = argv[1:]
2761
2762 for _, v := range argv {
2763     /*
2764     if v[0] == '-' { // might be a pseudo file (generated date)
2765         continue
2766     }
2767     */
2768     obj := strings.Split(v, ":")
2769     //fmt.Printf("%d %v %v\n", len(obj), v, obj)
2770     if 1 < len(obj) {
2771         host = obj[0]
2772         file = ""
2773         if 0 < len(host) {
2774             gsh.LastServer.host = host
2775         }else{
2776             host = gsh.LastServer.host
2777             port = gsh.LastServer.port
2778         }
2779         if 2 < len(obj) {
2780             port = obj[1]
2781             if 0 < len(port) {
2782                 gsh.LastServer.port = port
2783             }else{
2784                 port = gsh.LastServer.port
2785             }
2786             file = obj[2]
2787         }else{
2788             file = obj[1]
2789         }
2790         if len(srcv) == 0 {
2791             download = true
2792             srcv = append(srcv, file)
2793             continue
2794         }
2795         upload = true
2796         dstv = append(dstv, file)
2797         continue
2798     }
2799     /*
2800     idx := strings.Index(v, ":")
2801     if 0 <= idx {
2802         remote = v[0:idx]
2803         if len(srcv) == 0 {
2804             download = true
2805             srcv = append(srcv, v[idx+1:])
2806             continue
2807         }
2808         upload = true
2809         dstv = append(dstv, v[idx+1:])
2810         continue
2811     }
2812     */
2813     if download {
2814         dstv = append(dstv, v)
2815     }else{
2816         srcv = append(srcv, v)
2817     }
2818 }
2819 hostport := "" + host + ":" + port
2820 if upload {
2821     if host != "" { xargv = append(xargv, hostport) }
2822     xargv = append(xargv, "PUT")
2823     xargv = append(xargv, srcv[0]...)
2824     xargv = append(xargv, dstv[0]...)
2825     //fmt.Printf("--I-- FileCopy PUT gsh://%s/%s < %v // %v\n", hostport, dstv, srcv, xargv)
2826     fmt.Printf("--I-- FileCopy PUT gsh://%s/%s < %v\n", hostport, dstv, srcv)
2827     gsh.RexecClient(xargv)
2828 }else
2829 if download {
2830     if host != "" { xargv = append(xargv, hostport) }
2831     xargv = append(xargv, "GET")
2832     xargv = append(xargv, srcv[0]...)
2833     xargv = append(xargv, dstv[0]...)
2834     //fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v // %v\n", hostport, srcv, dstv, xargv)
2835     fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v\n", hostport, srcv, dstv)
2836     gsh.RexecClient(xargv)
2837 }else{
2838 }
2839 }
2840
2841 // target
2842 func (gsh*GshContext)Trelpath(rloc string)(string){
2843     cwd, _ := os.Getwd()
2844     os.Chdir(gsh.RWD)
2845     os.Chdir(rloc)
2846     twd, _ := os.Getwd()
2847     os.Chdir(cwd)
2848 }
2849 tpath := twd + "/" + rloc
2850 return tpath
2851 }
2852 // join to remote GShell - [user@host:port] or cd host:[port]:path
2853 func (gsh*GshContext)RJoin(argv[]string){
2854     if len(argv) <= 1 {
2855         fmt.Printf("--I-- current server = %v\n", gsh.RSERV)
2856         return
2857     }
2858     serv := argv[1]
2859     servv := strings.Split(serv, ":")
2860     if 1 <= len(servv) {
2861         if servv[0] == "lo" {
2862             servv[0] = "localhost"
2863         }
2864     }
2865     switch len(servv) {
2866     case 1:
2867         //if strings.Index(serv, ":") < 0 {
2868             serv = servv[0] + ":" + fmt.Sprintf("%d", GSH_PORT)
2869         //}
2870     case 2: // host:port
2871         serv = strings.Join(servv, ":")
2872     }
2873     xargv := []string{"rex-join", "@"+serv, "HELLO"}
2874     rcode, stat := gsh.RexecClient(xargv)
2875     if (rcode / 100) == 2 {
2876         fmt.Printf("--I-- OK Joined (%v) %v\n", rcode, stat)
2877         gsh.RSERV = serv
2878     }else{
2879         fmt.Printf("--I-- NG, could not joined (%v) %v\n", rcode, stat)
2880     }
2881 }
2882 func (gsh*GshContext)Rexec(argv[]string){
2883     if len(argv) <= 1 {
2884         fmt.Printf("--I-- rexec command [ | {file} | {command} ]\n", gsh.RSERV)
2885         return
2886     }
2887     /*
2888     nargv := gshScanArg(strings.Join(argv, " "), 0)
2889     fmt.Printf("--D-- nargc=%d %v\n", len(nargv), nargv)
2890     if nargv[1][0] != '{' {
2891         nargv[1] = "{" + nargv[1] + "}"
2892         fmt.Printf("--D-- nargc=%d %v\n", len(nargv), nargv)
2893     }
2894     argv = nargv
2895     */
2896     nargv := []string{}
2897     nargv = append(nargv, "{"+strings.Join(argv[1:], " ")+"}")
2898     fmt.Printf("--D-- nargc=%d %v\n", len(nargv), nargv)
2899     argv = nargv
2900 }
2901 xargv := []string{"rex-rxec", "@"+gsh.RSERV, "GET"}
2902 xargv = append(xargv, argv...)
2903 xargv = append(xargv, "/dev/tty")
2904 rcode, stat := gsh.RexecClient(xargv)
2905 if (rcode / 100) == 2 {
2906     fmt.Printf("--I-- OK Rexec (%v) %v\n", rcode, stat)
2907 }else{
2908     fmt.Printf("--I-- NG Rexec (%v) %v\n", rcode, stat)
2909 }
2910 }
2911 }
2912 func (gsh*GshContext)Rchdir(argv[]string){
2913     if len(argv) <= 1 {
2914         return
2915     }

```

```

2916     cwd, _ := os.Getwd()
2917     os.Chdir(gsh.RWD)
2918     os.Chdir(argv[1])
2919     twd, _ := os.Getwd()
2920     gsh.RWD = cwd
2921     fmt.Printf("--I-- JWD=%v\n", twd)
2922     os.Chdir(cwd)
2923 }
2924 func (gsh*GshContext)Rpwd(argv[]string){
2925     fmt.Printf("%v\n", gsh.RWD)
2926 }
2927 func (gsh*GshContext)Rls(argv[]string){
2928     cwd, _ := os.Getwd()
2929     os.Chdir(gsh.RWD)
2930     argv[0] = "-ls"
2931     gsh.XFind(argv)
2932     os.Chdir(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
2959 // <a name="network">network</a>
2960 // -s, -sl, -so // bi-directional, source, sync (maybe socket)
2961 func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
2962     gshPA := gshCtx.gshPA
2963     if len(argv) < 2 {
2964         fmt.Printf("Usage: -s [host]:[port].[udp]\n")
2965         return
2966     }
2967     remote := argv[1]
2968     if remote == ":" { remote = "0.0.0.0:9999" }
2969
2970     if inTCP { // TCP
2971         dport, err := net.ResolveTCPAddr("tcp", remote);
2972         if err != nil {
2973             fmt.Printf("Address error: %s (%s)\n", remote, err)
2974             return
2975         }
2976         conn, err := net.DialTCP("tcp", nil, dport)
2977         if err != nil {
2978             fmt.Printf("Connection error: %s (%s)\n", remote, err)
2979             return
2980         }
2981         file, _ := conn.File();
2982         //fd := file.Fd()
2983         //fmt.Printf("Socket: connected to %s, socket[%d]\n", remote, fd)
2984         fmt.Printf("Socket: connected to %s, socket[%d]\n", remote, file.Fd())
2985
2986         savfd := gshPA.Files[1]
2987         //gshPA.Files[1] = fd;
2988         gshPA.Files[1] = file;
2989         gshCtx.gshellv(argv[2:])
2990         gshPA.Files[1] = savfd
2991         file.Close()
2992         conn.Close()
2993     }else{
2994         //dport, err := net.ResolveUDPAddr("udp4", remote);
2995         dport, err := net.ResolveUDPAddr("udp", remote);
2996         if err != nil {
2997             fmt.Printf("Address error: %s (%s)\n", remote, err)
2998             return
2999         }
3000         //conn, err := net.DialUDP("udp4", nil, dport)
3001         conn, err := net.DialUDP("udp", nil, dport)
3002         if err != nil {
3003             fmt.Printf("Connection error: %s (%s)\n", remote, err)
3004             return
3005         }
3006         file, _ := conn.File();
3007         //fd := file.Fd()
3008
3009         ar := conn.RemoteAddr()
3010         //ar := conn.LocalAddr()
3011         fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
3012             //remote, ar.String(), fd)
3013             remote, ar.String(), file.Fd())
3014
3015         savfd := gshPA.Files[1]
3016         //gshPA.Files[1] = fd;
3017         gshPA.Files[1] = file;
3018         gshCtx.gshellv(argv[2:])
3019         gshPA.Files[1] = savfd
3020         file.Close()
3021         conn.Close()
3022     }
3023 }
3024 func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
3025     gshPA := gshCtx.gshPA
3026     if len(argv) < 2 {
3027         fmt.Printf("Usage: -ac [host]:[port].[udp]\n")
3028         return
3029     }
3030     local := argv[1]
3031     if local == ":" { local = "0.0.0.0:9999" }
3032     if inTCP { // TCP
3033         port, err := net.ResolveTCPAddr("tcp", local);
3034         if err != nil {
3035             fmt.Printf("Address error: %s (%s)\n", local, err)
3036             return
3037         }
3038         //fmt.Printf("Listen at %s...\n", local);
3039         sconn, err := net.ListenTCP("tcp", port)
3040         if err != nil {
3041             fmt.Printf("Listen error: %s (%s)\n", local, err)
3042             return
3043         }
3044         //fmt.Printf("Accepting at %s...\n", local);
3045         acconn := sconn.AcceptTCP()
3046         if err != nil {
3047             fmt.Printf("Accept error: %s (%s)\n", local, err)
3048             return
3049         }
3050         file, _ := acconn.File()
3051         //fd := file.Fd()
3052         //fmt.Printf("Accepted TCP at %s [%d]\n", local, fd)
3053         fmt.Printf("Accepted TCP at %s [%d]\n", local, file.Fd())
3054
3055         savfd := gshPA.Files[0]
3056         //gshPA.Files[0] = fd;
3057         gshPA.Files[0] = file;
3058         gshCtx.gshellv(argv[2:])
3059         gshPA.Files[0] = savfd
3060
3061         sconn.Close();
3062         acconn.Close();
3063         file.Close();
3064     }else{
3065         //port, err := net.ResolveUDPAddr("udp4", local);
3066         port, err := net.ResolveUDPAddr("udp", local);
3067         if err != nil {
3068             fmt.Printf("Address error: %s (%s)\n", local, err)
3069             return
3070         }
3071         //fmt.Printf("Listen UDP at %s...\n", local);
3072         //uconn, err := net.ListenUDP("udp4", port)
3073         uconn, err := net.ListenUDP("udp", port)
3074         if err != nil {
3075             fmt.Printf("Listen error: %s (%s)\n", local, err)
3076             return
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.Printf("Accepted at %s [%d] <- %s\n",local,fd,"")
3087
3088 savfd := gshPA.Files[0]
3089 //gshPA.Files[0] = fd;
3090 gshPA.Files[0] = file;
3091 savenv := gshPA.Env
3092 gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
3093 gshCtx.gshShellv(argv[2:]);
3094 gshPA.Env = savenv
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 isin("-a",argv):
3109 gshCtx.ShowChdirHistory(argv)
3110 case isin("-ls",argv):
3111 showFileInfo(cwd,argv)
3112 default:
3113 fmt.Printf("%s\n",cwd)
3114 case isin("-v",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\"",cwd)
3120 fmt.Printf(" HOST=\"%s\"",host)
3121 fmt.Printf(" DATE=\"%s\"",date)
3122 fmt.Printf(" TIME=\"%s\"",t.String())
3123 fmt.Printf(" PID=\"%d\"",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 // openfile-history, sort by -a -m -c
3133 // sort by elapsed time by -t -s
3134 // search by "more" like interface
3135 // edit history
3136 // sort history, and wc or uniq
3137 // CPU and other resource consumptions
3138 // limit showing range (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],"0") {
3143 atWorkDirX, _ = strconv.Atoi(argv[1][1:]);
3144 }
3145 //fmt.Printf("--D-- showHistory(%v)\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.Printf("%t-%d ",i)
3159 }
3160 if isin("-v",argv){
3161 fmt.Println(v) // should be with it date
3162 }else{
3163 if isin("-l",argv) || isin("-l0",argv) {
3164 elps := v.EndAt.Sub(v.StartAt);
3165 start := v.StartAt.Format(time.Stamp)
3166 fmt.Printf("%td ",v.WorkDirX)
3167 fmt.Printf("[%tv] %11v/t ",start,elps)
3168 }
3169 if isin("-l",argv) && !isin("-l0",argv){
3170 fmt.Printf("%tv",Rusagef("%t %u/t/ %s",argv,v.Rusagev))
3171 }
3172 if isin("-at",argv) { // isin("-ls",argv){
3173 dhi := v.WorkDirX // workdir history index
3174 fmt.Printf("%td %s/t",dhi,v.WorkDir)
3175 // show the FileInfo of the output command??
3176 }
3177 fmt.Printf("%s",v.CmdLine)
3178 fmt.Printf("\n")
3179 }
3180 }
3181 }
3182 // !n - 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.Printf("--E-- (%s : range)\n",hix)
3188 return "", false, true
3189 }
3190 if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3191 fmt.Printf("--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)ShowChdirHistory(i int, v GChdirHistory, argv []string){
3213 fmt.Printf("%t-%d ",v.CmdIndex) // the first command at this WorkDir
3214 fmt.Printf("%td ",i)
3215 fmt.Printf("[%tv] %v.MovedAt.Format(time.Stamp))
3216 showFileInfo(v.Dir,argv)
3217 }
3218 func (gsh*GshContext)ShowChdirHistory(argv []string){
3219 for i, v := range gsh.ChdirHistory {
3220 gsh.ShowChdirHistory(i,v,argv)
3221 }
3222 }
3223 func skipOpts(argv []string)(int){
3224 for i,v := range argv {
3225 if strBegins(v,"-") {
3226 }else{
3227 return i
3228 }
3229 }
3230 return -1
3231 }
3232 func (gshCtx*GshContext)xChdir(argv []string){
3233 chdir := gshCtx.ChdirHistory
3234 if isin("-a", argv) || isin("-at",argv) || isin("-s",argv) {
3235 gshCtx.ShowChdirHistory(argv)
3236 return
3237 }
3238 pwd, _ := os.Getwd()
3239 dir := ""

```

```

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[i+1]
3248     }
3249 }
3250 if strBegins(dir, "0") {
3251     if dir == "0" { // obsolete
3252         dir = gshCtx.StartDir
3253     }else
3254     if dir == "0:" {
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 != pwd {
3279             hist1 := GChdirHistory { }
3280             hist1.Dir = cwd
3281             hist1.MovedAt = time.Now()
3282             hist1.CmdIndex = len(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 }
3293 if isin("-ls", argv){
3294     cwd, _ := os.Getwd()
3295     showFileInfo(cwd, argv);
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
3324 // <a name="rusage">Resource Usage</a>
3325 func aRusagef(fmtspec string, argv []string, ru [2]aRusage)(string){
3326     // ru[0] self , ru[1] children
3327     ut := TimeValAdd(ru[0].Utime, ru[1].Utime)
3328     st := TimeValAdd(ru[0].Stime, ru[1].Stime)
3329     //uu := (int64(ut.Sec)*1000000 + int64(st.Sec)) * 1000
3330     //su := (int64(st.Sec)*1000000 + int64(st.Sec)) * 1000
3331     uu := ut // in nano sec
3332     su := st; // in nano sec
3333     tu := uu + su
3334     ret := fmt.Sprintf("%v/sum", abftime(tu))
3335     ret += fmt.Sprintf(", %v/usr", abftime(uu))
3336     ret += fmt.Sprintf(", %v/sys", abftime(su))
3337     return ret
3338 }
3339 func Rusagef(fmtspec string, argv []string, ru [2]aRusage)(string){
3340     ut := TimeValAdd(ru[0].Utime, ru[1].Utime)
3341     st := TimeValAdd(ru[0].Stime, ru[1].Stime)
3342     //fmt.Printf("%d.%06ds/u ", ut.Sec, ut.Usec) //ru[1].Utime.Sec, ru[1].Utime.Usec)
3343     //fmt.Printf("%d.%06ds/s ", st.Sec, st.Usec) //ru[1].Stime.Sec, ru[1].Stime.Usec)
3344     fmt.Printf("%d.%06ds/u ", ut/1000000000, (ut/1000)%1000000);
3345     fmt.Printf("%d.%06ds/s ", st/1000000000, (st/1000)%1000000);
3346     return ""
3347 }
3348
3349 func Getrusagev(){[2]aRusage{
3350     var ruv = [2]aRusage{
3351         aGetrusage(aRUSAGE_SELF, &ruv[0])
3352         aGetrusage(aRUSAGE_CHILDREN, &ruv[1])
3353     }
3354     return ruv
3355 }
3356 func (gshCtx *GshContext)xTime(argv []string)(bool){
3357     if 2 <= len(argv){
3358         gshCtx.LastRusage = aRusage{
3359             rusagev1 := Getrusagev()
3360             fin := gshCtx.gshellv(argv[1:])
3361             rusagev2 := Getrusagev()
3362             showRusage(argv[1], argv, &gshCtx.LastRusage)
3363             rusagev := RusageSub(rusagev2, rusagev1)
3364             showRusage("self", argv, &rusagev[0])
3365             showRusage("chld", argv, &rusagev[1])
3366             return fin
3367         }else{
3368             rusage := aRusage {
3369                 aGetrusage(aRUSAGE_SELF, &rusage)
3370                 showRusage("self", argv, &rusage)
3371                 aGetrusage(aRUSAGE_CHILDREN, &rusage)
3372                 showRusage("chld", argv, &rusage)
3373                 return false
3374             }
3375         }
3376     }
3377     func (gshCtx *GshContext)xJobs(argv []string){
3378         fmt.Printf("%d Jobs\n", len(gshCtx.BackGroundJobs))
3379         for ji, pid := range gshCtx.BackGroundJobs {
3380             //stat := syscall.Wait4(pid, nil, syscall.WNOHANG, &rusage);
3381             //wpid, err := syscall.Wait4(pid, nil, syscall.WNOHANG, &rusage);
3382             wpid := pid.Pid();
3383             err := errors.New("stab_NoError");
3384             if err != nil {
3385                 fmt.Printf("--E-- %%%d [%d] (%v)\n", ji, wpid, err)
3386             }else{
3387                 fmt.Printf("%%d [%d]\n", ji, wpid)
3388                 showRusage("chld", argv, &rusage)
3389             }
3390         }
3391     }
3392 }
3393 }
3394 func (gsh *GshContext)inBackground(argv []string)(bool){
3395     if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n", argv) }
3396     gsh.BackGround = true // set background option
3397     xfin := false
3398     xfin = gsh.gshellv(argv)
3399     gsh.BackGround = false
3400     return xfin
3401 }

```

```

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

```

```

3544 // scanf [format] [input]
3545 func scanf(str string)(strv []string){
3546     strv = strings.Split(str, " ")
3547     return strv
3548 }
3549 func scanUntil(src,end string)(rstr string, leng int){
3550     idx := strings.Index(src,end)
3551     if 0 <= idx {
3552         rstr = src[0:idx]
3553         return rstr,idx+leng(end)
3554     }
3555     return src,0
3556 }
3557 // -bn -- display base-name part only // can be in some fmt, for sed rewriting
3558 func (gsh*GshContext)printlnVal(fmts string, vstr string, optv []string){
3559     //vint,err := strconv.Atoi(vstr)
3560     var ival int64 = 0
3561     n := 0
3562     err := error(nil)
3563     if strBegins(vstr," ") {
3564         vx,_ := strconv.Atoi(vstr[1:])
3565         if vx < len(gsh.iValues) {
3566             vstr = gsh.iValues[vx]
3567         }else{
3568             }
3569         }
3570     // should use Eval()
3571     if strBegins(vstr,"0x") {
3572         n,err = fmt.Sscanf(vstr[2:], "%x", &ival)
3573     }else{
3574         n,err = fmt.Sscanf(vstr, "%d", &ival)
3575     }
3576     //fmt.Printf("--D-- n=%d err=%v) (%s)=%v\n",n,err,vstr, ival)
3577     if n == 1 && err == nil {
3578         //fmt.Printf("--D-- formatn(%v) ival(%v)\n",fmts,ival)
3579         fmt.Printf("%s"+fmts,ival)
3580     }else{
3581         if isin("-bn",optv){
3582             fmt.Printf("%s"+fmts,filepath.Base(vstr))
3583         }else{
3584             fmt.Printf("%s"+fmts,vstr)
3585         }
3586     }
3587 }
3588 }
3589 func (gsh*GshContext)printlnf(fmts,div string,argv []string,optv []string,list []string){
3590     //fmt.Printf("%d",len(list))
3591     //curfmt := "v"
3592     outlen := 0
3593     curfmt := gsh.iFormat
3594     if 0 < len(fmts) {
3595         for xi := 0; xi < len(fmts); xi++ {
3596             fch := fmts[xi]
3597             if fch == '$' {
3598                 if xi+1 < len(fmts) {
3599                     curfmt = string(fmts[xi+1])
3600                 }
3601                 xi += 1
3602                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3603                     vals,leng := scanUntil(fmts[xi+2:],")")
3604                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
3605                     gsh.printlnf(curfmt,vals,optv)
3606                     xi += 2+leng-1
3607                     outlen += 1
3608                 }
3609                 continue
3610             }
3611             if fch == '.' {
3612                 hl,leng := scanInt(fmts[xi+1:])
3613                 if 0 < leng {
3614                     if hl < len(gsh.iValues) {
3615                         gsh.printlnf(curfmt,gsh.iValues[hl],optv)
3616                         outlen += 1 // should be the real length
3617                     }else{
3618                         fmt.Printf("(out-range)")
3619                     }
3620                     xi += leng
3621                     continue;
3622                 }
3623             }
3624             fmt.Printf("%c",fch)
3625             outlen += 1
3626         }
3627     }else{
3628         //fmt.Printf("--D-- print (%s)\n")
3629         for i,v := range list {
3630             if 0 < i {
3631                 fmt.Printf(div)
3632             }
3633             gsh.printlnf(curfmt,v,optv)
3634             outlen += 1
3635         }
3636     }
3637     if 0 < outlen {
3638         fmt.Printf("\n")
3639     }
3640 }
3641 func (gsh*GshContext)Scanv(argv []string){
3642     //fmt.Printf("--D-- Scnav(%v)\n",argv)
3643     if len(argv) == 1 {
3644         return
3645     }
3646     argv = argv[1:]
3647     fmts := ""
3648     if strBegins(argv[0],"-F") {
3649         fmts = argv[0]
3650         gsh.iDelimiter = fmts
3651         argv = argv[1:]
3652     }
3653     input := strings.Join(argv, " ")
3654     if fmts == "" { // simple decomposition
3655         v := scanf(input)
3656         gsh.iValues = v
3657         //fmt.Printf("%v\n",strings.Join(v," "))
3658     }else{
3659         v := make([]string,8)
3660         n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3661         //fmt.Printf("--D-- Sscanf -> (%v) n=%d err=(%v)\n",v,n,err)
3662         gsh.iValues = v
3663     }
3664 }
3665 func (gsh*GshContext)Printv(argv []string){
3666     if false { //@@@
3667         fmt.Printf("%v\n",strings.Join(argv[1:], " "))
3668         return
3669     }
3670     //fmt.Printf("--D-- Printv(%v)\n",argv)
3671     //fmt.Printf("%v\n",strings.Join(gsh.iValues," "))
3672     div := gsh.iDelimiter
3673     fmts := ""
3674     argv = argv[1:]
3675     if 0 < len(argv) {
3676         if strBegins(argv[0],"-F") {
3677             div = argv[0][2:]
3678             argv = argv[1:]
3679         }
3680     }
3681     optv := []string{}
3682     for _,v := range argv {
3683         if strBegins(v,"-"){
3684             optv = append(optv,v)
3685             argv = argv[1:]
3686         }else{
3687             break;
3688         }
3689     }
3690     if 0 < len(argv) {
3691         fmts = strings.Join(argv, " ")
3692     }
3693     gsh.printlnf(fmts,div,argv,optv,gsh.iValues)
3694 }
3695 func (gsh*GshContext)Basename(argv []string){
3696     for i,v := range gsh.iValues {
3697         gsh.iValues[i] = filepath.Base(v)
3698     }
3699 }
3700 func (gsh*GshContext)Sortv(argv []string){
3701     sv := gsh.iValues
3702     sort.Slice(sv, func(i,j int) bool {

```

```

3726     return sv[i] < sv[j]
3727 }
3728 }
3729 func (gsh*GshContext)Shiftv(argv[]string){
3730     vi := len(gsh.iValues)
3731     if 0 < vi {
3732         if !isin("-",argv) {
3733             top := gsh.iValues[0]
3734             gsh.iValues = append(gsh.iValues[1:],top)
3735         }else{
3736             gsh.iValues = gsh.iValues[1:]
3737         }
3738     }
3739 }
3740
3741 func (gsh*GshContext)Eng(argv[]string){
3742 }
3743 func (gsh*GshContext)Deq(argv[]string){
3744 }
3745 func (gsh*GshContext)Push(argv[]string){
3746     gsh.iValStack = append(gsh.iValStack,argv[1:])
3747     fmt.Printf("depth=%d\n",len(gsh.iValStack))
3748 }
3749 func (gsh*GshContext)Dump(argv[]string){
3750     for i,v := range gsh.iValStack {
3751         fmt.Printf("%d %v\n",i,v)
3752     }
3753 }
3754 func (gsh*GshContext)Pop(argv[]string){
3755     depth := len(gsh.iValStack)
3756     if 0 < depth {
3757         v := gsh.iValStack[depth-1]
3758         if !isin("-",argv){
3759             gsh.iValues = append(gsh.iValues,v...)
3760         }else{
3761             gsh.iValues = v
3762         }
3763         gsh.iValStack = gsh.iValStack[0:depth-1]
3764         fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3765     }else{
3766         fmt.Printf("depth=%d\n",depth)
3767     }
3768 }
3769
3770 // <a name="interpreter">Command Interpreter</a>
3771 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3772     fin = false
3773
3774     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3775     if len(argv) <= 0 {
3776         return false
3777     }
3778     xargv := []string{}
3779     for ai := 0; ai < len(argv); ai++ {
3780         xargv = append(xargv,strings.TrimSpace(argv[ai]),false)
3781     }
3782     argv = xargv
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)\n",len(argv),argv) }
3791     switch { // https://tour.golang.org/flowcontrol/11
3792     case cmd == "":
3793         gshCtx.xPwd([]string{}); // empty command
3794     case cmd == "-x":
3795         gshCtx.CmdTrace = ! gshCtx.CmdTrace
3796     case cmd == "-xt":
3797         gshCtx.CmdTime = ! gshCtx.CmdTime
3798     case cmd == "-ot":
3799         gshCtx.sconnect(true, argv)
3800     case cmd == "-ou":
3801         gshCtx.sconnect(false, argv)
3802     case cmd == "-it":
3803         gshCtx.saccept(true, argv)
3804     case cmd == "-iu":
3805         gshCtx.saccept(false, argv)
3806     case cmd == "-i" || cmd == "-c" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3807         gshCtx.redirect(argv)
3808     case cmd == "|":
3809         gshCtx.fromPipe(argv)
3810     case cmd == "args":
3811         gshCtx.Args(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 == "-ksum":
3822         gshCtx.xFind(argv)
3823     case cmd == "-sum":
3824         gshCtx.xFind(argv)
3825     case cmd == "-sumtest":
3826         str := ""
3827         if 1 < len(argv) { str = argv[1] }
3828         crc := strCRC32(str,uint64(len(str)))
3829         fmt.Printf(stderr,"%v %v\n",crc,len(str))
3830     case cmd == "close":
3831         gshCtx.xClose(argv)
3832     case cmd == "gcp":
3833         gshCtx.FileCopy(argv)
3834     case cmd == "dec" || cmd == "decode":
3835         gshCtx.Dec(argv)
3836     case cmd == "#define":
3837         case cmd == "dic" || cmd == "d":
3838             xDic(argv)
3839     case cmd == "dump":
3840         gshCtx.Dump(argv)
3841     case cmd == "echo" || cmd == "e":
3842         echo(argv,true)
3843     case cmd == "enc" || cmd == "encode":
3844         gshCtx.Enc(argv)
3845     case cmd == "env":
3846         env(argv)
3847     case cmd == "eval":
3848         xEval(argv[1:],true)
3849     case cmd == "ev" || 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 == "fdls":
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 == "gset":
3872         gshCtx.Deq(argv)
3873     case cmd == "genq":
3874         gshCtx.End(argv)
3875     case cmd == "gpop":
3876         gshCtx.Pop(argv)
3877     case cmd == "gpush":
3878         gshCtx.Push(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.SplitLine(argv)
3885     case cmd == "-ls":
3886         gshCtx.xFind(argv)
3887     case cmd == "nop":

```



```

3888 // do nothing
3889 case cmd == "pipe":
3890     gshCtx.xOpen(argv)
3891 case cmd == "plug" || 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.Printv(argv)
3896 case cmd == "ps":
3897     gshCtx.xPs(argv)
3898 case cmd == "pstitle":
3899     // to be gsh.title
3900 case cmd == "rexec" || cmd == "rexd":
3901     gshCtx.RexecServer(argv)
3902 case cmd == "rexec" || 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.xHttpServer(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.Rexec(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 == "where":
3949     // data file or so?
3950 case cmd == "which":
3951     which("PATH",argv);
3952 case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
3953     go gj_server(argv[1]);
3954 case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
3955     go gj_server(argv[1]);
3956 case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
3957     go gj_client(argv[1]);
3958 case cmd == "gj":
3959     jsend(argv);
3960 case cmd == "jsend":
3961     jsend(argv);
3962 default:
3963     if gshCtx.whichPlugin(cmd,[]string{"-"}) != 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.gshellv(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("--T-- " + 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\\w",
3997     // fi.Mode(),fi.Mode()&os.ModeDevice)
3998     if (fi.Mode() & os.ModeDevice) != 0 {
3999         stat := aStat_t{};
4000         err := aStat(0,&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.Printf("--I-- GSH_HOME=%s\\n",gshCtx.GshHomeDir)
4014     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
4015         fmt.Sprintf("%02d",gshCtx.TerminalId)
4016     //strconv.Itoa(gshCtx.TerminalId)
4017     //fmt.Printf("--I-- ttyfile=%s\\n",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         else
4039         if( with_xgetline && gshCtx.GetLine != "" ){
4040             //var xhix int64 = int64(hix); // cast
4041             newenv := os.Environ()
4042             newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
4043             tty := gshCtx.ttyline()
4044             tty.WriteString(prevline)
4045             Pa := os.ProcAttr {
4046                 "", // start dir
4047                 newenv, //os.Environ(),
4048                 []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},

```

```

4050     nil,
4051 }
4052 //fmt.Printf("--I-- getline=%s // %s\n", gsh_getline[0], gshCtx.GetLine)
4053 proc, err := os.StartProcess(gsh_getline[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.Printf("--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 } else
4076 /*
4077 {
4078     // if isatty {
4079     //     fmt.Printf("%d", hix)
4080     //     fmt.Print(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/exec/">os/exec</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 StrBuff []byte
4116 var NULL_FP *os.File = nil
4117 var NULLSP = 0
4118 //var LINESIZE = 1024
4119
4120 func system(cmdstr string) (int) {
4121     //PA := syscall.ProcAttr {
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.Printf("--Es-- system(%v)\n(%v)\n", cmdstr, err);
4134         return -1;
4135     }
4136     pstat, _ := proc.Wait();
4137     pid := pstat.Pid();
4138     if (err != nil) {
4139         fmt.Printf("--E-- pid=%v syscall(%v) err(%v)\n", pid, cmdstr, err)
4140     }
4141     //syscall.Wait4(pid, nil, 0, nil)
4142     //fmt.Printf("====E== 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 out bytes.Buffer
4151 cmd.Stdout = &out
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.Printf("ERR:%s\n", serr.String())
4158 } else {
4159     fmt.Printf("%s", out.String())
4160 }
4161 */
4162 return 0
4163 }
4164
4165 func atoi(str string) (ret int) {
4166     ret, err := fmt.Sscanf(str, "%d", &ret)
4167     if err == nil {
4168         return ret
4169     } else {
4170         // should set errno
4171         return 0
4172     }
4173 }
4174
4175 func getenv(name string) (string) {
4176     val, got := os.LookupEnv(name)
4177     if got {
4178         return val
4179     } else {
4180         return "?"
4181     }
4182 }
4183
4184 func strcpy(dst StrBuff, src string) {
4185     var i int
4186     srcb := []byte(src)
4187     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4188         dst[i] = srcb[i]
4189     }
4190     dst[i] = 0
4191 }
4192
4193 func xstrcpy(dst StrBuff, src StrBuff) {
4194     dst = src
4195 }
4196
4197 func strcat(dst StrBuff, src StrBuff) {
4198     dst = append(dst, src...)
4199 }
4200
4201 func strdup(str StrBuff) (string) {
4202     return string(str[0:strlen(str)])
4203 }
4204
4205 func strlen(str string) (int) {
4206     return len(str)
4207 }
4208
4209 func strlen(str StrBuff) (int) {
4210     var i int
4211     for i = 0; i < len(str) && str[i] != 0; i++ {
4212     }
4213     return i
4214 }
4215
4216 func sizeof(data StrBuff) (int) {
4217     return len(data)
4218 }
4219
4220 func isatty(fd int) (ret int) {
4221     return 1
4222 }

```

```

4212 }
4213
4214 func fopen(file string,mode string)(fp*os.File){
4215     if mode == "r" {
4216         fp,err := os.Open(file)
4217         if( err != nil ){
4218             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
4219             return NULL_FP;
4220         }
4221         return fp;
4222     }else{
4223         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4224         if( err != nil ){
4225             return NULL_FP;
4226         }
4227         return fp;
4228     }
4229 }
4230 func fclose(fp*os.File){
4231     fp.Close()
4232 }
4233 func fflush(fp *os.File)(int){
4234     return 0
4235 }
4236 func fgetc(fp*os.File)(int){
4237     var buf [1]byte
4238     err := fp.Read(buf[0:1])
4239     if( err != nil ){
4240         return EOF;
4241     }else{
4242         return int(buf[0])
4243     }
4244 }
4245 func sfgets(str*string, size int, fp*os.File)(int){
4246     buf := make(StrBuff,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 fputc(ch int , fp*os.File)(int){
4283     var buf [1]byte
4284     buf[0] = byte(ch)
4285     fp.Write(buf[0:1])
4286     return 0
4287 }
4288 func fputs(buf StrBuff, fp*os.File)(int){
4289     fp.Write(buf)
4290     return 0
4291 }
4292 func xfputss(str string, fp*os.File)(int){
4293     return fputs([]byte(str),fp)
4294 }
4295 func sscanf(str StrBuff,fmts string, params ...interface{})(int){
4296     fmt.Sscanf(string(str[0:strlen(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/0.0.2 ----- MyIME
4306 var MYIMEVER = "MyIME/0.0.2";
4307 type RomKana struct {
4308     dic string // dictionary ID
4309     pat string // input pattern
4310     out string // output pattern
4311     hit int64 // count of hit and used
4312 }
4313 var dicents = 0
4314 var romkana [1024]RomKana
4315 var Romkan []RomKana
4316
4317 func isinDic(str string)(int){
4318     for i,v := range Romkan {
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_ENA = "en"
4330     DIC_COM_DIS = "di"
4331 }
4332 func helpDic(argv []string){
4333     out := stderr
4334     cmd := ""
4335     if 0 < len(argv) { cmd = argv[0] }
4336     fprintf(out,"--- %v Usage\n",cmd)
4337     fprintf(out,"... Commands\n")
4338     fprintf(out,"... %v %3v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
4339     fprintf(out,"... %v %3v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
4340     fprintf(out,"... %v %3v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
4341     fprintf(out,"... %v %3v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
4342     fprintf(out,"... %v %3v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
4343     fprintf(out,"... Keys ... %v\n", "ESC 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,"... \\j -- On/Off translation mode\n",)
4347     fprintf(out,"... \\l -- Force Lower Case\n",)
4348     fprintf(out,"... \\u -- Force Upper Case (software CapsLock)\n",)
4349     fprintf(out,"... \\v -- Show translation actions\n",)
4350     fprintf(out,"... \\x -- Replace the last input character with it Hexa-Decimal\n",)
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     opt := ""
4366     arg := ""
4367
4368     if 0 < len(argv) {
4369         arg1 := argv[0]
4370         if arg1[0] == '-' {
4371             switch arg1 {
4372                 default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4373             }
4374         }
4375     }

```

```

4374     case "-b": builtin = true
4375     case "-d": debug = true
4376     case "-s": silent = true
4377     case "-v": info = true
4378     }
4379     opt = arg1
4380     argv = argv[1:]
4381 }
4382 }
4383
4384 dicName := ""
4385 dicURL := ""
4386 if 0 < len(argv) {
4387     arg = argv[0]
4388     dicName = arg
4389     argv = argv[1:]
4390 }
4391 if 0 < len(argv) {
4392     dicURL = argv[0]
4393     argv = argv[1:]
4394 }
4395 if false {
4396     fprintf(stderr, "--Dd-- com(%v) opt(%v) arg(%v)\n", cmd, opt, arg)
4397 }
4398 if cmd == DIC_COM_LOAD {
4399     //dictype := ""
4400     dicbody := ""
4401     if !builtin && dicName != "" && dicURL == "" {
4402         f, err := os.Open(dicName)
4403         if err == nil {
4404             dicURL = dicName
4405         } else {
4406             f, err = os.Open(dicName+".html")
4407             if err == nil {
4408                 dicURL = dicName+".html"
4409             } else {
4410                 f, err = os.Open("gshdic-"+dicName+".html")
4411                 if err == nil {
4412                     dicURL = "gshdic-"+dicName+".html"
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 "sumomo":
4439             dicName = "SumomoDic"
4440             dicURL = "SumomoDic"
4441         case "sijimi":
4442             dicName = "SijimiDic"
4443             dicURL = "SijimiDic"
4444         case "jkl":
4445             dicName = "JKLJaDic"
4446             dicURL = "JKLJaDic"
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, "type: %v\n", dicv[0])
4455             fprintf(stderr, "body: %v\n", dicv[1])
4456             fprintf(stderr, "\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     entv := 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 entv {
4476         var pat string
4477         var out string
4478         fmt.Scanf(v, "%s %s", &pat, &out)
4479         if len(pat) <= 0 {
4480             } else {
4481                 if 0 <= isinDic(pat) {
4482                     dup += 1
4483                     continue
4484                 }
4485                 romkana[dictents] = RomKana{dicName, pat, out, 0}
4486                 dictents += 1
4487                 added += 1
4488                 Romkan = append(Romkan, RomKana{dicName, pat, out, 0})
4489                 if debug {
4490                     fmt.Printf("%3v: [%2v]%-8v [%2v] %v\n",
4491                         i, len(pat), pat, len(out), out)
4492                 }
4493             }
4494         }
4495     }
4496     if !silent {
4497         url := dicURL
4498         if strBegins(url, "data:") {
4499             url = "builtin"
4500         }
4501         fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4502             dicName, added, dup, len(Romkan), url);
4503     }
4504     // should sort by pattern length for complete match, for performance
4505     if debug {
4506         arg = "" // search pattern
4507         dump = true
4508     }
4509     if cmd == DIC_COM_DUMP || dump {
4510         fprintf(stderr, "--Id-- %v dump... %v entries:\n", dicName, len(Romkan));
4511         var match = 0
4512         for i := 0; i < len(Romkan); i++ {
4513             dic := Romkan[i].dic
4514             pat := Romkan[i].pat
4515             out := Romkan[i].out
4516             if arg == "" || 0 <= strings.Index(pat, arg) || 0 <= strings.Index(out, arg) {
4517                 fmt.Printf("\t\t\t%v [%2v]%-8v [%2v] %v\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(Romkan));
4523     }
4524 }
4525 func loadDefaultDic(int) {
4526     if 0 < len(Romkan) {
4527         return
4528     }
4529     //fprintf(stderr, "\n")
4530     xDic[0] = strings{"dic", DIC_COM_LOAD};
4531 }
4532 var info = false
4533 if info {
4534     fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\n")
4535     fprintf(stderr, "--Id-- enter \"dic\" command for help.\n")

```

```

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(dic,"r");
4545 if( rk == NULL_FP ){
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,"%s %i\n",&pat,&out);
4562 romkana[di].pat = pat;
4563 romkana[di].out = out;
4564 //fprintf(stderr,"--Dd- %10s %s\n",pat,out)
4565 }
4566 dicents += di
4567 if( false ){
4568 fprintf(stderr,"--%s-- loaded romkana.txt [%d]\n",MYIMEVER,di);
4569 for di = 0; di < dicents; di++ {
4570 fprintf(stderr,
4571 "%s %s\n",romkana[di].pat,romkana[di].out);
4572 }
4573 }
4574 }
4575 fclose(rk);
4576 //romkana[dicents].pat = "//ddump"
4577 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4578 */
4579 return 0;
4580 }
4581 }
4582 func matchlen(stri string, pati string)(int){
4583 if strBegins(stri,pati) {
4584 return len(pati)
4585 }else{
4586 return 0
4587 }
4588 }
4589 func convs(src string)(string){
4590 var si int;
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:], "kx/") {
4598 // %k/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 sval := fmt.Sprintf("%x",iv)
4605 bval := []byte(sval)
4606 dstb = append(dstb,bval...)
4607 si = si+3+ix+1
4608 continue
4609 }
4610 }
4611 if strBegins(src[si:], "d/") {
4612 // %d/integer // s/a/b/
4613 ix := strings.Index(src[si+3:],"/")
4614 if 0 < ix {
4615 var iv int = 0
4616 fmt.Sscanf(src[si+3:si+3+ix],"%v",&iv)
4617 sval := fmt.Sprintf("%d",iv)
4618 bval := []byte(sval)
4619 dstb = append(dstb,bval...)
4620 si = si+3+ix+1
4621 continue
4622 }
4623 }
4624 if strBegins(src[si:], "t") {
4625 now := time.Now()
4626 if true {
4627 date := now.Format(time.Stamp)
4628 dstb = append(dstb,[]byte(date)...)
4629 si = si+3
4630 }
4631 continue
4632 }
4633 var maxlen int = 0;
4634 var len int;
4635 mi = -1;
4636 for di = 0; di < dicents; 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 xfputs(dst,stderr);
4657 return 0;
4658 }
4659 }
4660 //----- LINEEDIT
4661 // "?" at the top of the line means searching history
4662
4663 // should be compatile 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 = 251 // 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_TOPW = 239 // b
4678 GO_ENDW = 238 // e
4679 GO_NEXTW = 237 // w
4680
4681 GO_FORWCH = 229 // f
4682 GO_PAIRCH = 228 // %
4683
4684 GO_DEL = 219 // d
4685
4686 HI_SRCH_FW = 209 // /
4687 HI_SRCH_BK = 208 // ?
4688 HI_SRCH_NW = 207 // n
4689 HI_SRCH_RBK = 206 // N
4690 )
4691
4692 // should return number of octets ready to be read immediately
4693 //fprintf(stderr,"\n--Select(%v %v)\n",err,r.Bits[0])
4694
4695
4696 var EventRecvFd = -1 // file descriptor
4697 var EventSendFd = -1

```

```

4698 const EventFdOffset = 100000
4699 const NormalFdOffset = 100
4700
4701 /* 2020-1021 replaced poll() with channel/select
4702 func putKeyinEvent(event int, evarg int){
4703     if true {
4704         if EventRecvFd < 0 {
4705             var pv = [int{-1,-1}]
4706             syscall.Pipe(pv)
4707             EventRecvFd = pv[0]
4708             EventSendFd = pv[1]
4709             //fmt.Printf("--De-- 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?s=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("--De-- EventSock created[%v,%v]\n",
4720                     EventRecvFd,EventSendFd,err)
4721             }
4722         }
4723     }
4724     var buf = []byte{ byte(event)}
4725     n,err := syscall.Write(EventSendFd,buf)
4726     if err != nil {
4727         fmt.Printf("--De-- putEvent[%v](%v %v)\n",EventSendFd,event,n,err)
4728     }
4729 }
4730 */
4731 func ungets(str string){
4732     for _,ch := range str {
4733         putKeyinEvent(int(ch),0)
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             //fmt.Printf(stderr, "--Dr-- tempo=[%v]\n",a[2:],tempo);
4746         }else
4747         if strBegins(a,"r") { // repeat
4748             fmt.Sscanf(a[1:], "%v", &repeat)
4749         }else
4750         if strBegins(a,"l") {
4751             fmt.Sscanf(a[1:], "%d", &hix)
4752         }else{
4753             fmt.Sscanf(a, "%d", &hix)
4754         }
4755     }
4756     if hix == 0 || len(argv) <= 1 {
4757         hix = len(gsh.CommandHistory)-1
4758     }
4759     fmt.Printf("--Ir-- Replay(1%v x%v r%v)\n",hix,xtempo,repeat)
4760     //dumpEvents(hix)
4761     //gsh.xScanReplay(hix,false,repeat,tempo,argv)
4762     go gsh.xScanReplay(hix,true,repeat,tempo,argv)
4763     runtime.Gosched(); // wait xScanReplay is launched
4764     //fmt.Printf("--Ir-- Replay set\n");
4765 }
4766
4767 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4768 // 2020-0827 GShell-0.2.3
4769 /*
4770 func FpollIn1(fp *os.File,usec int)(uintptr){
4771     nfd := 1
4772     rdv := syscall.FdSet {
4773         fd1 := fp.Fd()
4774         bank1 := fd1/32
4775         mask1 := int32(1 << fd1)
4776         rdv.Bits[bank1] = mask1
4777     }
4778     fd2 := -1
4779     bank2 := -1
4780     var mask2 int32 = 0
4781     if 0 <= EventRecvFd {
4782         fd2 = EventRecvFd
4783         nfd = fd2 + 1
4784         bank2 = fd2/32
4785         mask2 = int32(1 << fd2)
4786         rdv.Bits[bank2] |= mask2
4787         //fmt.Printf("--De-- EventPoll mask added [%d][%v]\n",fd2,bank2,mask2)
4788     }
4789     tout := syscall.NsecToTimeval(int64(usec*1000))
4790     //n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
4791     err := syscall.Select(nfd,&rdv,nil,nil,&tout)
4792     if err != nil {
4793         //fmt.Printf("--De-- select() err(%v)\n",err)
4794     }
4795     if err == nil {
4796         if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4797             if false {
4798                 fmt.Printf("--De-- got Event\n")
4799             }
4800             return uintptr(EventFdOffset + fd2)
4801         }else
4802         if (rdv.Bits[bank1] & mask1) != 0 {
4803             return uintptr(NormalFdOffset + fd1)
4804         }else{
4805             return 1
4806         }
4807     }else{
4808         return 0
4809     }
4810 }
4811 */
4812 func fgetTimeout1(fp *os.File,usec int)(int){
4813     READ1:
4814     //readyFd := FpollIn1(fp,usec)
4815     readyFd := CpollIn1(fp,usec)
4816     if readyFd < 100 {
4817         return EV_TIMEOUT
4818     }
4819     var buf [1]byte
4820     if EventFdOffset <= readyFd {
4821         fd := int(readyFd-EventFdOffset)
4822         _,err := syscall.Read(fd,buf[0:1])
4823         if( err != nil ){
4824             return EOF;
4825         }else{
4826             if buf[0] == EV_MODE {
4827                 recvKeyEvent(fd)
4828                 goto READ1
4829             }
4830             return int(buf[0])
4831         }
4832     }
4833     _,err := fp.Read(buf[0:1])
4834     if( err != nil ){
4835         return EOF;
4836     }else{
4837         return int(buf[0])
4838     }
4839 }
4840 */
4841 func visibleChar(ch int)(string){
4842     switch {
4843     case '!': return "!"
4844     case '<': return "<"
4845     case '>': return ">"
4846     case '\n': return "\n"
4847     case '\r': return "\r"
4848     }
4849     return string(ch)
4850 }

```

```

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 /*
4878 func recvKeyEvent(fd int){
4879     var buf = make([]byte,1)
4880     _,_ = syscall.Read(fd,buf[0:1])
4881     if( buf[0] != 0 ){
4882         romkanmode = true
4883     }else{
4884         romkanmode = false
4885     }
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.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
4913                     putKeyEvent(e.event,0)
4914                     if e.event == EV_MODE { // event with arg
4915                         putKeyEvent(int(e.evarg),0)
4916                     }
4917                 }else{
4918                     //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
4919                 }
4920             }else{
4921                 fmt.Printf("%7.3fms %#-3v %-3v [%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.Printf("--replay %v / %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 || i.CmdIndex == hix {
4948             fmt.Printf("%#-3v %-3v [%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 /*
4957 func fgetTimeout(fp *os.File,usec int){int){
4958     ch := fgetTimeout(fp,usec)
4959     if ch != EV_TIMEOUT {
4960         now := time.Now()
4961         if 0 < len(Events) {
4962             last := Events[len(Events)-1]
4963             dura := int64(now.Sub(last.when))
4964             Events = append(Events,Event(last.when,EV_IDLE,dura,last.CmdIndex))
4965         }
4966         Events = append(Events,Event(time.Now(),ch,0,CmdIndex))
4967     }
4968     return ch
4969 }
4970 */
4971
4972 // 2020-1021 replaced poll() with channel/select
4973 var Kbd = make(chan int);
4974 var Kbinet = false;
4975 var evQ = make(chan int);
4976 /*
4977 func keyInput(kbd chan int, fp *os.File){
4978     for {
4979         ch := C.get(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://stackoverflow.com/questions/14094190/function-similar-to-getchar
4989 func keyInput(kbd chan int, tty *os.File){
4990     tmode := C.setTermRaw();
4991     defer func(){ C.setTermMode(tmode); }();
4992     if( !OnWindows ){
4993         system("/bin/stty -echo -icanon");
4994         defer func(){ system("/bin/stty echo 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 ){
5009         system("/bin/stty echo sane"); }
5010 }
5011 func fgetTimeout(fp *os.File,usec int){int){
5012     if( !Kbinet ){
5013         Kbinet = true;
5014         go keyInput(Kbd,fp);
5015     }
5016     for {
5017         select {
5018         case <- time.After(time.Duration(usec*1000)):
5019             //fmt.Printf("--Timeout %v us\n",usec);
5020             return EV_TIMEOUT;
5021         case ch := <- Kbd:
5022             //fmt.Printf("--KBD[%X]\n",ch);

```

```

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, EV_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 }
5042 func putKeyInEvent(event int, evarg int){
5043     evQ <- event;
5044 }
5045 func recvKeyEvent(){
5046     ch := <- evQ;
5047     if( ch != 0 ){
5048         romkanmode = true
5049     }else{
5050         romkanmode = false
5051     }
5052 }
5053 }
5054 var AtConsoleLineTop = true
5055 var TtyMaxCol = 72 // to be obtained by ioctl?
5056 var EscTimeout = (100*1000)
5057 var {
5058     MODE_VicMode bool // vi compatible command mode
5059     MODE_ShowMode bool
5060     romkanmode bool // shown translation mode, the mode to be retained
5061     MODE_Recursive bool // recursive translation
5062     MODE_CapsLock bool // software CapsLock
5063     MODE_LowerLock bool // force lower-case character lock
5064     MODE_ViInsert int // visible insert mode, should be like "I" icon in X Window
5065     MODE_ViTrace bool // output newline before translation
5066 }
5067 type IInput struct {
5068     lno int
5069     lastlno int
5070     pch []int // input queue
5071     prompt string
5072     line string
5073     right string
5074     inJmode bool
5075     pinJmode bool
5076     waitingMeta string // waiting meta character
5077     LastCmd string
5078 }
5079 func (iin*IInput)Getc(timeoutUs int)(int){
5080     ch1 := EOF
5081     ch2 := EOF
5082     ch3 := EOF
5083     if( 0 < len(iin.pch) ){ // deQ
5084         ch1 = iin.pch[0]
5085         iin.pch = iin.pch[1:]
5086     }else{
5087         ch1 = fgetcTimeout(stdin, timeoutUs);
5088     }
5089     if( ch1 == 033 ){ // escape sequence
5090         ch2 = fgetcTimeout(stdin, EscTimeout);
5091         if( ch2 == EV_TIMEOUT ){
5092             }else{
5093                 ch3 = fgetcTimeout(stdin, EscTimeout);
5094                 if( ch3 == EV_TIMEOUT ){
5095                     iin.pch = append(iin.pch, ch2) // enQ
5096                 }else{
5097                     switch( ch2 ){
5098                         default:
5099                             iin.pch = append(iin.pch, ch2) // enQ
5100                             iin.pch = append(iin.pch, ch3) // enQ
5101                     case '[':
5102                         switch( ch3 ){
5103                             case 'A': ch1 = GO_UP; // ^
5104                             case 'B': ch1 = GO_DOWN; // v
5105                             case 'C': ch1 = GO_RIGHT; // >
5106                             case 'D': ch1 = GO_LEFT; // <
5107                             case '3':
5108                                 ch4 := fgetcTimeout(stdin, EscTimeout);
5109                                 if( ch4 == '-' ){
5110                                     //fprintf(stderr, "%i%02X %02X %02X %02X\n", ch1, ch2, ch3, ch4);
5111                                     ch1 = DEL_RIGHT
5112                                 }
5113                             case '\\':
5114                                 //ch4 := fgetcTimeout(stdin, EscTimeout);
5115                                 //fprintf(stderr, "%i%02X %02X %02X %02X\n", ch1, ch2, ch3, ch4);
5116                                 switch( ch3 ){
5117                                     case '-': ch1 = DEL_RIGHT
5118                                 }
5119                             }
5120                         }
5121                     }
5122                 }
5123             }
5124         return ch1
5125     }
5126 }
5127 func (iin*IInput)clearline(){
5128     var i int
5129     fprintf(stderr, "\r");
5130     // should be ANSI ESC sequence
5131     for i = 0; 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, iin.lno, iin.line, iin.right)
5138 }
5139 func redraw(iin *IInput, lno int, line string, right string){
5140     inMeta := false
5141     showMeta := "" // visible Meta mode on the cursor position
5142     showlno := fmt.Sprintf("%d", lno)
5143     InsertMark := "" // in visible insert mode
5144     if MODE_VicMode {
5145     }else
5146     if 0 < len(iin.right) {
5147         InsertMark = " "
5148     }
5149     if( 0 < len(iin.waitingMeta) ){
5150         inMeta = true
5151         if iin.waitingMeta[0] != 033 {
5152             showMeta = iin.waitingMeta
5153         }
5154     }
5155     if( romkanmode ){
5156         //romkanmark = " *";
5157     }else{
5158         //romkanmark = "";
5159     }
5160     if MODE_ShowMode {
5161         romkan := "-"
5162         inmeta := "-"
5163         inveri := ""
5164         if MODE_CapsLock {
5165             inmeta = "A"
5166         }
5167         if MODE_LowerLock {
5168             inmeta = "a"
5169         }
5170         if MODE_ViTrace {
5171             inveri = "v"
5172         }
5173         if MODE_VicMode {
5174             inveri = ":"
5175         }
5176         if romkanmode {
5177             romkan = "\343\201\202"
5178         }
5179         if MODE_CapsLock {
5180             inmeta = "R"
5181         }
5182     }else{
5183         inmeta = "r"
5184     }

```



```

5184     }
5185     }
5186     if inMeta {
5187         inMeta = ""
5188     }
5189     showMode = "["+romkan+inmeta+inveri+"]";
5190 }
5191 Pre := "\r" + showMode + showLino
5192 Output := ""
5193 Left := ""
5194 Right := ""
5195 if romkanmode {
5196     Left = convs(line)
5197     Right = InsertMark+convs(right)
5198 }else{
5199     Left = line
5200     Right = InsertMark+right
5201 }
5202 Output = Pre+Left
5203 if MODE_ViTrace {
5204     Output += iin.LastCmd
5205 }
5206 Output += showMeta+Right
5207 for len(Output) < ZtyMaxCol { // to the max. position that may be dirty
5208     Output += " "
5209     // should be ANSI ESC sequence
5210     // not necessary just after newline
5211 }
5212 Output += Pre+Left+showMeta // to set the cursor to the current input position
5213 fprintf(stderr, "%s", Output)
5214 }
5215 if MODE_ViTrace {
5216     if 0 < len(iin.LastCmd) {
5217         iin.LastCmd = ""
5218         fprintf(stderr, "\r\n")
5219     }
5220 }
5221 AtConsoleLineTop = false
5222 //fmt.Printf("(Redraw(%v))\n", len(line), len(right));
5223 }
5224 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5225 func delHeadChar(str string)(rline string, head string){
5226     clen := utf8.DecodeRune([]byte(str))
5227     head = string(str[:clen])
5228     return str[clen:], head
5229 }
5230 func delTailChar(str string)(rline string, last string){
5231     var i = 0
5232     var clen = 0
5233     for {
5234         clen := utf8.DecodeRune([]byte(str)[i:])
5235         if clen == 0 { break }
5236         clen = clen
5237         i += clen
5238     }
5239     last = str[len(str)-clen:]
5240     return str[0:len(str)-clen], last
5241 }
5242 }
5243 // 3> for output and history
5244 // 4> for keylog?
5245 // <a name="getline">Command Line Editor</a>
5246 func xgetline(lno int, prevline string, gsh*GshContext)(string){
5247     var iin Input
5248     iin.lastlno = lno
5249     iin.lno = lno
5250 }
5251 CmdIndex = len(gsh.CommandHistory)
5252 if !isatty(0) == 0 {
5253     if !fgets(&iin.line, LINESIZE, stdin) == NULL ){
5254         iin.line = "exit\n";
5255     }else{
5256     }
5257     return iin.line
5258 }
5259 if true {
5260     //var pts string;
5261     //pts = ptname(0);
5262     //pts = ttyname(0);
5263     //fprintf(stderr, "--pts[0] = %s\n, pts?pts:??");
5264 }
5265 if false {
5266     fprintf(stderr, "l ");
5267     fflush(stderr);
5268     fgets(&iin.line, LINESIZE, stdin);
5269     return iin.line
5270 }
5271 if !onWindows { system("/bin/stty -echo -icanon"); }
5272 xline := iin.xgetline(prevline, gsh)
5273 if !onWindows { system("/bin/stty echo sane"); }
5274 return xline
5275 }
5276 func (iin*Input)Translate(cmdch int){
5277     romkanmode = !romkanmode;
5278     if MODE_ViTrace {
5279         fprintf(stderr, "%v\r\n", string(cmdch));
5280     }else
5281     if cmdch == 'j' {
5282         fprintf(stderr, "J\r\n");
5283         iin.inMode = true
5284     }
5285     iin.Redraw();
5286     loadDefaultDic(cmdch);
5287     iin.Redraw();
5288 }
5289 func (iin*Input)Replace(cmdch int){
5290     iin.LastCmd = fmt.Sprintf("%v", string(cmdch))
5291     iin.Redraw();
5292     loadDefaultDic(cmdch);
5293     dst := convs(iin.line+iin.right);
5294     iin.line = dst
5295     iin.right = ""
5296     if cmdch == 'I' {
5297         fprintf(stderr, "I\r\n");
5298         iin.inMode = true
5299     }
5300     iin.Redraw();
5301 }
5302 // aa 12 alal
5303 func isAlpha(ch rune)(bool){
5304     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5305         return true
5306     }
5307     return false
5308 }
5309 func isAlnum(ch rune)(bool){
5310     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5311         return true
5312     }
5313     if '0' <= ch && ch <= '9' {
5314         return true
5315     }
5316     return false
5317 }
5318 }
5319 // 0.2.8 2020-0901 created
5320 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5321 func (iin*Input)GotoTOPW(){
5322     str := iin.line
5323     i := len(str)
5324     if i <= 0 {
5325         return
5326     }
5327     //i0 := i
5328     i -= 1
5329     lastSize := 0
5330     var lastRune rune
5331     var found = -1
5332     for 0 < i { // skip preamble spaces
5333         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5334         if !isAlnum(lastRune) { // character, type, or string to be searched
5335             i -= lastSize
5336             continue
5337         }
5338         break
5339     }
5340     for 0 < i {
5341         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5342         if lastSize <= 0 { continue } // not the character top
5343         if !isAlnum(lastRune) { // character, type, or string to be searched
5344             found = i
5345             break

```

```

5346     }
5347     i -= lastSize
5348 }
5349 if found < 0 && i == 0 {
5350     found = 0
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     }
5358     iin.right = str[i:] + iin.right
5359     if 0 < i {
5360         iin.line = str[0:i]
5361     }else{
5362         iin.line = ""
5363     }
5364 }
5365 //fmt.Printf("\n(%d,%d,%d){%s}\n",i0,i,found,iin.line,iin.right)
5366 //fmt.Printf("") // set debug messae at the end of line
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     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
5379     if !isAlnum(lastRune) {
5380         r,z := utf8.DecodeRuneInString(str[lastSize:])
5381         if 0 < z && !isAlnum(r) {
5382             inWord = true
5383         }
5384     }
5385     for i < len(str) {
5386         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5387         if lastSize <= 0 { break } // broken data?
5388         if !isAlnum(lastRune) { // character, type, or string to be searched
5389             break
5390         }
5391         lastW = i // the last alnum if in alnum word
5392         i += lastSize
5393     }
5394     if inWord {
5395         goto DISP
5396     }
5397     for i < len(str) {
5398         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5399         if lastSize <= 0 { break } // broken data?
5400         if !isAlnum(lastRune) { // character, type, or string to be searched
5401             break
5402         }
5403         i += lastSize
5404     }
5405     for i < len(str) {
5406         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5407         if lastSize <= 0 { break } // broken data?
5408         if !isAlnum(lastRune) { // character, type, or string to be searched
5409             break
5410         }
5411         lastW = i
5412         i += lastSize
5413     }
5414     DISP:
5415     if 0 < lastW {
5416         iin.line = iin.line + str[0:lastW]
5417         iin.right = str[lastW:]
5418     }
5419     //fmt.Printf("\n(%d){%s}\n",i,iin.line,iin.right)
5420     //fmt.Printf("") // set debug messae at the end of line
5421 }
5422 // 0.2.8 2020-0901 created
5423 func (iin*Input)GotoNEXTW(){
5424     str := iin.right
5425     if len(str) <= 0 {
5426         return
5427     }
5428     lastSize := 0
5429     var lastRune rune
5430     var found = -1
5431     i := 1
5432     for i < len(str) {
5433         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5434         if lastSize <= 0 { break } // broken data?
5435         if !isAlnum(lastRune) { // character, type, or string to be searched
5436             found = i
5437             break
5438         }
5439         i += lastSize
5440     }
5441     if 0 < found {
5442         if !isAlnum(lastRune) { // or non-kana character
5443         }else{ // when positioning to the top o the word
5444             found += lastSize
5445         }
5446         iin.line = iin.line + str[0:found]
5447         if 0 < found {
5448             iin.right = str[found:]
5449         }else{
5450             iin.right = ""
5451         }
5452     }
5453     //fmt.Printf("\n(%d){%s}\n",i,iin.line,iin.right)
5454     //fmt.Printf("") // set debug messae at the end of line
5455 }
5456 // 0.2.8 2020-0902 created
5457 func (iin*Input)GotoPAIRCH(){
5458     str := iin.right
5459     if len(str) <= 0 {
5460         return
5461     }
5462     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5463     if lastSize <= 0 {
5464         return
5465     }
5466     forw := false
5467     back := false
5468     pair := ""
5469     switch string(lastRune){
5470     case "(": pair = ")"; forw = true
5471     case ")": pair = "("; back = true
5472     case "(": pair = ")"; forw = true
5473     case ")": pair = "("; back = true
5474     case "[": pair = "]"; forw = true
5475     case "]": pair = "["; back = true
5476     case "<": pair = ">"; forw = true
5477     case ">": pair = "<"; back = true
5478     case "\'": pair = "\'"; // context depednet, can be f" or back-double quote
5479     case "f": pair = "f"; // context depednet, can be f' or back-quote
5480     // case Japanese Kakkos
5481     }
5482     if forw {
5483         iin.SearchForward(pair)
5484     }
5485     if back {
5486         iin.SearchBackward(pair)
5487     }
5488 }
5489 // 0.2.8 2020-0902 created
5490 func (iin*Input)SearchForward(pat string)(bool){
5491     right := iin.right
5492     found := -1
5493     i := 0
5494     if strBegins(right,pat) {
5495         _,z := utf8.DecodeRuneInString(right[i:])
5496         if 0 < z {
5497             i += z
5498         }
5499     }
5500     for i < len(right) {
5501         if strBegins(right[i:],pat) {
5502             found = i
5503             break
5504         }
5505     }
5506     _,z := utf8.DecodeRuneInString(right[i:])
5507     if z <= 0 { break }

```

```

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 (iin*Input)SearchBackward(pat string)(bool){
5520     line := iin.line
5521     found := -1
5522     i := len(line)-1
5523     for i = 1; 0 <= i; i-- {
5524         z := utf8.DecodeRuneInString(line[i:])
5525         if z <= 0 {
5526             continue
5527         }
5528         //fprintf(stderr,"-- %v %v\n",pat,line[i:])
5529         if strBegins(line[i:],pat) {
5530             found = i
5531             break
5532         }
5533     }
5534     //fprintf(stderr,"--%d\n",found)
5535     if 0 <= found {
5536         iin.right = line[found:] + iin.right
5537         iin.line = line[0: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                 //fprintf(stderr,"\n--De-- found %v [%v]%v\n",i,pat,v.CmdLine)
5550                 return true,v.CmdLine
5551             }
5552         }
5553     }else{
5554         hlen := len(gsh.CommandHistory)
5555         for i := hlen-1; 0 < i; i-- {
5556             v := gsh.CommandHistory[i]
5557             if 0 <= strings.Index(v.CmdLine,pat) {
5558                 //fprintf(stderr,"\n--De-- found %v [%v]%v\n",i,pat,v.CmdLine)
5559                 return true,v.CmdLine
5560             }
5561         }
5562     }
5563     //fprintf(stderr,"\n--De-- not-found(%v)\n",pat)
5564     return false,"(Not Found in History)"
5565 }
5566 // 0.2.8 2020-0902 created
5567 func (iin*Input)GotoFORWSTR(pat string,gsh*GshContext){
5568     found := false
5569     if 0 < len(iin.right) {
5570         found = iin.SearchForward(pat)
5571     }
5572     if !found {
5573         found,line := gsh.SearchHistory(pat,true)
5574         if found {
5575             iin.line = line
5576             iin.right = ""
5577         }
5578     }
5579 }
5580 func (iin*Input)GotoBACKSTR(pat string,gsh*GshContext){
5581     found := false
5582     if 0 < len(iin.line) {
5583         found = iin.SearchBackward(pat)
5584     }
5585     if !found {
5586         found,line := gsh.SearchHistory(pat,false)
5587         if found {
5588             iin.line = line
5589             iin.right = ""
5590         }
5591     }
5592 }
5593 func (iin*Input)getString(prompt string)(string){ // should be editable
5594     iin.clearline();
5595     fprintf(stderr,"\rtv",prompt)
5596     str := ""
5597     for {
5598         ch := iin.Getc(10*1000*1000)
5599         if ch == '\n' || ch == '\r' {
5600             break
5601         }
5602         sch := string(ch)
5603         str += sch
5604         fprintf(stderr,"%s",sch)
5605     }
5606     return str
5607 }
5608 // search pattern must be an array and selectable with "W"/P
5609 var SearchPat = ""
5610 var SearchForw = true
5611 var SearchForw = true
5612 func (iin*Input)xgetline(prevline string,gsh*GshContext)(string){
5613     var ch int;
5614     MODE_ShowMode = false
5615     MODE_VicMode = false
5616     iin.Redraw();
5617     first := true
5618     for cix := 0; ; cix++ {
5619         iin.pinJmode = iin.inJmode
5620         iin.inJmode = false
5621         ch = iin.Getc(1000*1000)
5622         if ch != EV_TIMEOUT && first {
5623             first = false
5624             mode := 0
5625             if romkanmode {
5626                 mode = 1
5627             }
5628             now := time.Now()
5629             Events = append(Events,Event(now,EV_MODE,int64(mode),Cmdindex))
5630         }
5631         if ch == 033 {
5632             MODE_ShowMode = true
5633             MODE_VicMode = !MODE_VicMode
5634             iin.Redraw();
5635             continue
5636         }
5637         if MODE_VicMode {
5638             switch ch {
5639                 case '0': ch = GO_TOPL
5640                 case '5': ch = GO_ENDL
5641                 case 'b': ch = GO_TOPW
5642                 case 'e': ch = GO_ENDW
5643                 case 'w': ch = GO_NEXTW
5644                 case 's': ch = GO_PAIRCH
5645                 case 'j': ch = GO_DOWN
5646                 case 'k': ch = GO_UP
5647                 case 'h': ch = GO_LEFT
5648                 case 'l': ch = GO_RIGHT
5649                 case 'x': ch = DEL_RIGHT
5650                 case 'a': MODE_VicMode = !MODE_VicMode
5651                     ch = GO_RIGHT
5652                 case 'i': MODE_VicMode = !MODE_VicMode
5653                     iin.Redraw();
5654                     continue
5655                 case '-':
5656                     right,head := delHeadChar(iin.right)
5657                     if len([]byte(head)) == 1 {
5658                         ch = int(head[0])
5659                         if ('a' <= ch && ch <= 'z') {
5660                             ch = ch + 'A-' + 'a'
5661                         }else
5662                         if ('A' <= ch && ch <= 'Z') {
5663                             ch = ch + 'a-' + 'A'
5664                         }
5665                     }
5666                 }
5667             }
5668         }
5669     }

```

```

5670     }
5671     iin.right = string(ch) + right
5672     }
5673     iin.Redraw();
5674     continue
5675 case 'f': // GO_FORWCH
5676     iin.Redraw();
5677     ch = iin.Getc(3*1000*1000)
5678     if ch == EV_TIMEOUT {
5679         iin.Redraw();
5680         continue
5681     }
5682     SearchPat = string(ch)
5683     SearchForw = true
5684     iin.GotoFORWSTR(SearchPat,gsh)
5685     iin.Redraw();
5686     continue
5687 case '/':
5688     SearchPat = iin.getstring("/") // should be editable
5689     SearchForw = true
5690     iin.GotoFORWSTR(SearchPat,gsh)
5691     iin.Redraw();
5692     continue
5693 case '?':
5694     SearchPat = iin.getstring("?") // should be editable
5695     SearchForw = false
5696     iin.GotoBACKSTR(SearchPat,gsh)
5697     iin.Redraw();
5698     continue
5699 case 'n':
5700     if SearchForw {
5701         iin.GotoFORWSTR(SearchPat,gsh)
5702     }else{
5703         iin.GotoBACKSTR(SearchPat,gsh)
5704     }
5705     iin.Redraw();
5706     continue
5707 case 'w':
5708     if !SearchForw {
5709         iin.GotoFORWSTR(SearchPat,gsh)
5710     }else{
5711         iin.GotoBACKSTR(SearchPat,gsh)
5712     }
5713     iin.Redraw();
5714     continue
5715     }
5716 }
5717 switch ch {
5718 case GO_TOPW:
5719     iin.GotoTOPW()
5720     iin.Redraw();
5721     continue
5722 case GO_ENDW:
5723     iin.GotoENDW()
5724     iin.Redraw();
5725     continue
5726 case GO_NEXTW:
5727     // to next space then
5728     iin.GotoNEXTW()
5729     iin.Redraw();
5730     continue
5731 case GO_PAIRCH:
5732     iin.GotoPAIRCH()
5733     iin.Redraw();
5734     continue
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     cmdch := ch
5747     if( ch == EV_TIMEOUT ){
5748         if metach == 033 {
5749             continue
5750         }
5751         ch = metach
5752     }else
5753     /*
5754     if( ch == 'm' || ch == 'M' ){
5755         mch := fgetcTimeout(stdin,1000*1000)
5756         if mch == 'r' {
5757             romkanmode = true
5758         }else{
5759             romkanmode = false
5760         }
5761         continue
5762     }else
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\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\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\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                 if( 'A' <= ch && ch <= 'Z' ){
5812                     ch = ch + 'a'-'A'
5813                 }
5814                 iin.line = xline + string(ch)
5815             }
5816         }
5817         if MODE_ViTrace {
5818             fprintf(stderr,"%v\n",string(cmdch));
5819         }
5820         iin.Redraw();
5821         continue
5822     }else{
5823         iin.pch = append(iin.pch,ch) // push
5824         ch = '\\\
5825     }
5826 }
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     case '\t':
5840         iin.Replace('j');
5841         continue
5842     case 'X'-0x40:
5843         iin.Replace('j');
5844         continue
5845     case EV_TIMEOUT:
5846         iin.Redraw();
5847         if iin.pinMode {
5848             fprintf(stderr, "\\J\\r\\n")
5849             iin.inmode = true
5850         }
5851         continue
5852     case GO_UP:
5853         if iin.lno == 1 {
5854             continue
5855         }
5856         cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
5857         if ok {
5858             iin.line = cmd
5859             iin.right = ""
5860             iin.lno = iin.lno - 1
5861         }
5862         iin.Redraw();
5863         continue
5864     case GO_DOWN:
5865         cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
5866         if ok {
5867             iin.line = cmd
5868             iin.right = ""
5869             iin.lno = iin.lno + 1
5870         }else{
5871             iin.line = ""
5872             iin.right = ""
5873             if iin.lno == iin.lastlno-1 {
5874                 iin.lno = iin.lno + 1
5875             }
5876         }
5877         iin.Redraw();
5878         continue
5879     case GO_LEFT:
5880         if 0 < len(iin.line) {
5881             xline,tail := delTailChar(iin.line)
5882             iin.line = xline
5883             iin.right = tail + iin.right
5884         }
5885         iin.Redraw();
5886         continue;
5887     case GO_RIGHT:
5888         if( 0 < len(iin.right) && iin.right[0] != 0 ){
5889             xright,head := delHeadChar(iin.right)
5890             iin.right = xright
5891             iin.line += head
5892         }
5893         iin.Redraw();
5894         continue;
5895     case SDF:
5896         goto EXIT;
5897     case 'R'-0x40: // replace
5898         dat := convs(iin.line+iin.right);
5899         iin.line = dat
5900         iin.right = ""
5901         iin.Redraw();
5902         continue;
5903     case 'T'-0x40: // just show the result
5904         readDic();
5905         romkanmode = !romkanmode;
5906         iin.Redraw();
5907         continue;
5908     case 'L'-0x40:
5909         iin.Redraw();
5910         continue
5911     case 'K'-0x40:
5912         iin.right = ""
5913         iin.Redraw();
5914         continue
5915     case 'E'-0x40:
5916         iin.line += iin.right
5917         iin.right = ""
5918         iin.Redraw();
5919         continue
5920     case 'A'-0x40:
5921         iin.right = iin.line + iin.right
5922         iin.line = ""
5923         iin.Redraw();
5924         continue
5925     case 'U'-0x40:
5926         iin.line = ""
5927         iin.right = ""
5928         iin.clearline();
5929         iin.Redraw();
5930         continue;
5931     case DEL_RIGHT:
5932         if( 0 < len(iin.right) ){
5933             iin.right,_ = delHeadChar(iin.right)
5934             iin.Redraw();
5935         }
5936         continue;
5937     case 0x7F: // BS? not DEL
5938         if( 0 < len(iin.line) ){
5939             iin.line,_ = delTailChar(iin.line)
5940             iin.Redraw();
5941         }
5942         /*
5943         else
5944         if( 0 < len(iin.right) ){
5945             iin.right,_ = delHeadChar(iin.right)
5946             iin.Redraw();
5947         }
5948         */
5949         continue;
5950     case 'H'-0x40:
5951         if( 0 < len(iin.line) ){
5952             iin.line,_ = delTailChar(iin.line)
5953             iin.Redraw();
5954         }
5955         continue;
5956     }
5957     }
5958     if( OnWindows && ch == '\\n' ){
5959         continue;
5960     }
5961     if( ch == '\\n' || ch == '\\r' ){
5962         iin.line += iin.right;
5963         iin.right = ""
5964         iin.Redraw();
5965         //fputc(ch,stderr);
5966         fprintf(stderr, "\\r\\n"); // NL on Unix, CR on Windows
5967         AtConsoleLineTop = true
5968         break;
5969     }
5970     if MODE_CapsLock {
5971         if 'a' <= ch && ch <= 'z' {
5972             ch = cht'A'-'a'
5973         }
5974     }
5975     if MODE_LowerLock {
5976         if 'A' <= ch && ch <= 'Z' {
5977             ch = cht'a'-'A'
5978         }
5979     }
5980     iin.line += string(ch);
5981     iin.Redraw();
5982 }
5983 EXIT:
5984 return iin.line + iin.right;
5985 }
5986 func getline_main(){
5987     line := xgetline(0,"",nil)
5988     fprintf(stderr,"%s\n",line);
5989     /*
5990     dp = strpbk(line,"\\r\\n");
5991     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( romkmode ){
6002         var buf [8*1024]byte;
6003         convs(line, buf);
6004         stropy(line, buf);
6005     }
6006     write(3, line, strlen(line));
6007     ftruncate(3, lseek(3,0,SEEK_CUR));
6008     //fprintf(stderr, "outsize=%d\n", (int)lseek(3,0,SEEK_END));
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)gshSetupHomedir()(bool) {
6028     homedir, found := userHomeDir()
6029     if !found {
6030         fmt.Printf("--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         err3 := 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.Printf("--I-- Created %s\n", gshhome)
6043     }
6044     gshCtx.GshHomeDir = gshhome
6045     return false
6046 }
6047 func setupGshContext()(GshContext, bool){
6048     //gshPA := syscall.ProcAttr {
6049     gshPA := os.ProcAttr {
6050         "", // 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     }
6056     cwd, _ := os.Getwd()
6057     gshCtx := GshContext {
6058         cwd, // StartDir
6059         "", // GetLine
6060         []GchdirHistory { {cwd, time.Now(), 0} }, // ChdirHistory
6061         gshPA,
6062         []GCommandHistory {}, // something for invocation?
6063         GCommandHistory {}, // CmdCurrent
6064         false,
6065         []os.ProcessState {}, //[]int {},
6066         aRusage {},
6067         "", // GshHomeDir
6068         ttyid(),
6069         false,
6070         false,
6071         []PluginInfo {},
6072         []string {},
6073         "",
6074         "",
6075         ValueStack {},
6076         GServer{"", ""}, // LastServer
6077         "", // RSRV
6078         cwd, // RWD
6079         CheckSum {},
6080     }
6081     err := gshCtx.gshSetupHomedir()
6082     return gshCtx, err
6083 }
6084 func (gsh *GshContext)gshellh(gline string)(bool){
6085     ghist := gsh.CmdCurrent
6086     ghist.WorkDir, _ = os.Getwd()
6087     ghist.WorkDirX = len(gsh.ChdirHistory)-1
6088     //fmt.Printf("--D-- ChdirHistory(%d)\n", len(gsh.ChdirHistory))
6089     ghist.StartAt = time.Now()
6090     rusagev1 := Getrusagev()
6091     gsh.CmdCurrent.FoundFile = []string{}
6092     fin := gsh.gshellh(gline)
6093     rusagev2 := Getrusagev()
6094     ghist.Rusagev = RusageSubv(rusagev2, 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 // <a name="main">Main loop/</a>
6110 func script(gshCtxGiven *GshContext) (_ GshContext) {
6111     gshCtxBuf, err0 := setupGshContext()
6112     if err0 {
6113         return gshCtxBuf;
6114     }
6115     gshCtx := *gshCtxBuf
6116     //fmt.Printf("--I-- GSH_HOME=%s\n", gshCtx.GshHomeDir)
6117     //resmap()
6118     /*
6119     if false {
6120         gsh_getlinev, with_exgetline :=
6121             which("PATH", []string{"which", "gsh-getline", "-s"})
6122         if with_exgetline {
6123             gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
6124             gshCtx.GetLine = toFullpath(gsh_getlinev[0])
6125         }else{
6126             fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6127         }
6128     }
6129     */
6130     ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
6131     gshCtx.CommandHistory = append(gshCtx.CommandHistory, ghist0)
6132     prevline := ""
6133     skipping := false
6134     for hix := len(gshCtx.CommandHistory); ; {
6135         gline := gshCtx.getline(hix, skipping, prevline)
6136         if skipping {
6137             if strings.Index(gline, "fi") == 0 {
6138                 fmt.Printf("fi\n");
6139                 skipping = false;
6140             }else{
6141                 //fmt.Printf("%s\n", gline);
6142             }
6143             continue
6144         }
6145         if strings.Index(gline, "if") == 0 {
6146             //fmt.Printf("--D-- if start: %s\n", gline);
6147             skipping = true;
6148             continue
6149         }
6150         if false {

```



```

6318 //</span>
6319
6320 var $ijimiDic = //<span id="gsh-ijimi-dic">
6321 "data:text/dic;base64,+
6322 "PC11dG92b2hcnm1dD0VVRkLFRpIjPge8dGV4dGFyZWE9Y29scz04MCRyY3ZzPTQwPgovL3JlI+
6323 "cglH2h1bGxke0NRVXZ2ZGljdGltbWVFeVx2Zm9YXHRtAGlwa1pXHV1LzXhMjAyM0w0NDw+
6324 "CnNpcE0BlvzackJ44GXCmpC0BmAp4QnJgB8RmEJ44GqCmp1Ce0Bm0C0p4eXU344KF+
6325 "CnU344G0Cm5pCe0B0pBwbnjz2MYXU344G2Cm5Uc0e0CkVpUbnjg44Y3ZpCe0B0p40nJ+
6326 "gSRz2Z14CnBpC0C0AnjgTXE1444C0CnuyW5Wc64qpsAmY1C0wGqg4hskJ+
6327 "5LqMctvEanLg1sKa29xwEAlvprb3gJ5YCLcm5hbmfqDXvuaXgJzIKBmFuYwP1dW5peHgJ+
6328 "77yX7YScm5hbmfqDXvuaXg77YX7Y7YScu84gWngE86HgZn2IKa291dW5uCeWai+WlHpg0+
6329 "aWhcm5xc0BoE0B1+OC1q0aWhcmEJ5YqbcM0aWt.hcmE5J5pCjwvGv4dGFyZWE+Cg+
6330 //</span>
6331
6332 var JA_JKldic = //<span id="gsh-ja-jk1-dic">
6333 "data:text/dic;base64,+
6334 "Ly92X2ZkCUI5SU1FamRyZ2pib3Z2WpKQWpKS0w0mJyMG0wODE5KShLW4pL1Nh4G94SVRT+
6335 "CmtqamprbGtqa2tsa2ps10S41ueVjApqamtgamwJ44GCCmtqbAnjgYQKa2tqbAnjgYKamtq+
6336 "amwJ44G1Cmtqa2trbAnjgY0Ka2pza2wJ44GLCmpraamtRbAnjgY0Ka2tramJ44GCPmrapms+
6337 "Ce0B0pqpamprbAnjgY0Kamtq2psCe0B0pqpamtga2wJ44GCCmpraamtqbAnjgY0Ka2pqa+
6338 "Ce0B0pqpamprbAnjgY0Kamtq2psCe0B0pqpamprbAnjgY0Ka2pqa2wJ44GCCmpraamtqbAnjgYK+
6339 "a2qa2tsce0B0paprantsCe0B0pqa2pbrbAnjgAsKa2tra2wJ44GScmpga2psCe0B0pqa2p+
6340 "bnjgAsKamtqa2pJ44G0Cmpga2pqbAnjgIKampra2wJ44G1Cm5sCe0B0pqa2tsce0B0p+
6341 "a2tqbAnjgY0Ka2ts2psCe0B0pqbAnjg0aKamtq2psCe0C0pqa2tqa2wJ44GCCmpraamtq+
6342 "44KECmpra2pqbAnjg0tKampsCe0C1Apra2tsce0C1Qpamtq2psCe0C1Qpqa2pqa2wJ44KLCmp+
6343 "amwJ44KRCmtqa2psCe0C1Qpqa2psce0C1wpramtramJ44KRCmtqamprbAnjgY0Ka2pqa+
6344 "44KECmpra2pqbAnjgY0Ka2pqa2psce0C0Apra2wJ44KRCmpraamtqbAnjgY0Ka2pqa+
6345 "gIER";
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
6362 <a href="https://developer.mozilla.org/ja/docs/Web">MDN web docs</a>
6363 <a href="https://developer.mozilla.org/en-US/docs/Web/HTML/Element">HTML</a>
6364 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS">CSS</a> : <span class="wrap">
6365 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/Selectors">Selectors</a>
6366 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/Background-repeat">repeat</a></span>
6367 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
6368 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
6369 <a href="https://mdn-web-dns.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">The Go Programming Language</a>
6373 <a href="https://golang.org/pkg">Packages</a>
6374 <a href="https://golang.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 -->
6380 </div></details>
6381 */
6382 /*
6383 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>
6384
6385 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
6386 <details id="gsh-whole-view"><summary>Whole file</summary>
6387 <a name="whole-view-src"></a>
6388 <span id="src-frame"></span><!-- a window to show source code -->
6389 </details>
6390
6391 <details id="gsh-style-frame" onclick="fill_CSSView();"><summary>CSS part</summary>
6392 <a name="style-src-view"></a>
6393 <span id="gsh-style-view"></span>
6394 </details>
6395
6396 <details id="gsh-script-frame" onclick="fill_JavaScriptView();"><summary>JavaScript part</summary>
6397 <a name="script-src-view"></a>
6398 <span id="gsh-script-view"></span>
6399 </details>
6400
6401 <details id="gsh-data-frame" onclick="fill_DataView();"><summary>Built-in data part</summary>
6402 <a name="gsh-data-frame"></a>
6403 <span id="gsh-data-view"></span>
6404 </details>
6405
6406 </div></details>
6407 */
6408 /*
6409 <div id="GshFooter0"></div>
6410 <!-- 2020-09-17 SatokITS, visible script { -- >
6411 <details><summary>GScript</summary>
6412 <style>gscript { font-family:Georgia; }</style>
6413 <pre id="gscript_1" class="gscript" function gjtest1(){ alert('Hello GScript!'); }
6414 gjtest1()
6415 </pre>
6416 <script>
6417 gjs = document.getElementById('gscript_1');
6418 //eval(gjs.innerHTML);
6419 //gjs.outerHTML = ""
6420 </script>
6421 </details><!-- ----- END-OF-VISIBLE-PART ----- } -->
6422
6423 <!--
6424 // 2020-0906 added,
6425 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
6426 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6427 -->
6428 <span id="GshGrid">{"_"}</small><Hit j k l h</small></span>
6429
6430 <span id="GStat"><hr>
6431 </span>
6432 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6433 <span id="GTop"></span>
6434 <div id="GShellPlane" onclick="showGShellPlane();"></div>
6435 <div id="RawTextViewer"></div>
6436 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6437
6438 <style id="GshStyleDef">
6439 #LineNumbered table,tr,td {
6440 margin:0;
6441 padding:4px;
6442 spacing:0;
6443 border:12px;
6444 }
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
6454 textarea.LineNumbered {
6455 font-size:12px;
6456 font-family:monospace,Courier New;
6457 padding:4px;
6458 wrap:off;
6459 }
6460
6461 #RawTextViewer{
6462 z-index:0;
6463 position:fixed; top:0px; left:0px;
6464 width:100%; xxxheight:50px; xheight:0px;
6465 overflow:auto;
6466 color:#fff; background-color:rgba(128,128,256,0.2);
6467 font-size:12px;
6468 spellcheck:false;
6469 }
6470
6471 #RawTextViewerClose{
6472 z-index:0;
6473 position:fixed; top:100px; left:100px;
6474 color:#fff; background-color:rgba(128,128,256,0.2);
6475 font-size:20px; font-family:Georgia;
6476 white-space:pre;
6477 }
6478
6479 #xxxGShellPlane{
6480 z-index:0;
6481 position:fixed; top:0px; left:0px;
6482 width:100%; height:50px;
6483 overflow:auto;
6484 }

```



```

4480 color:#fff; background-color:rgba(128,128,256,0.3);
4481 font-size:12px;
4482 }
4483 #xxxGTop{
4484 z-index:9;
4485 opacity:1.0;
4486 position:fixed; top:0px; left:0px;
4487 width:120px; height:120px;
4488 color:#fff; background-color:rgba(32,32,160,0.15);
4489 color:#fff; font-size:12px;
4490 }
4491 #xxxGPos{
4492 z-index:12;
4493 position:fixed; top:0px; left:0px;
4494 opacity:1.0;
4495 width:60px; height:30px;
4496 color:#fff; background-color:rgba(0,0,0,0.2);
4497 color:#fff; font-size:12px;
4498 }
4499 #GMenu{
4500 z-index:10000000;
4501 position:fixed; top:250px; left:0px;
4502 opacity:1.0;
4503 width:100px; height:100px;
4504 color:#fff;
4505 color:#fff; background-color:rgba(0,0,0,0.0);
4506 color:#fff; font-size:16px; font-family:Georgia;
4507 background-repeat:no-repeat;
4508 }
4509 #xxxGStat{
4510 z-index:8;
4511 xopacity:0.0;
4512 position:fixed; top:20px; left:0px;
4513 width:60px; height:30px;
4514 width:100%; height:90px;
4515 color:#fff; background-color:rgba(0,0,128,0.04);
4516 font-size:20px; font-family:Georgia;
4517 }
4518 #GLog{
4519 z-index:10;
4520 position:fixed; top:50px; left:0px;
4521 opacity:1.0;
4522 width:640px; height:60px;
4523 color:#fff; background-color:rgba(0,0,128,0.10);
4524 font-size:12px;
4525 }
4526 #GshGrid {
4527 z-index:11;
4528 xopacity:0.0;
4529 position:fixed; top:0px; left:0px;
4530 width:320px; height:30px;
4531 color:#fff; font-size:16px;
4532 }
4533 xbody {display:none;}
4534 .gsh-link{color:green;}
4535 #gsh {border-width:1px;margin:0;padding:0;}
4536 #gsh {font-family:monospace,Courier New;color:#fff;font-size:8px;}
4537 #gsh header {height:100px;}
4538 #gsh header {height:100px;background-image:url(GShell-Logo00.png);}
4539 #GshMenu {font-size:14pt;color:#c44;}
4540 .GshMenu {font-size:14pt;color:#c44;}
4541 .GshMenu {
4542 font-size:14pt;color:#2a2;padding:4px; text-align:right;
4543 }
4544 .GshMenu: hover{
4545 font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
4546 }
4547 #GshFooter {height:100px;background-size:80px;background-repeat:no-repeat;}
4548 #gsh note {color:#000;font-size:10pt;}
4549 #gsh h2 {color:#2a2;font-family:Georgia;font-size:18pt;}
4550 #gsh h3 {color:#2a2;font-family:Georgia;font-size:16pt;}
4551 #gsh details {color:#888;background-color:#fff;font-family:monospace;}
4552 #gsh summary {font-size:16pt;color:#fff;background-color:#8af;xxheight:30px;}
4553 #gsh summary {font-size:16pt;color:#fff;
4554 padding:2pt;
4555 line-height:1.0;
4556 vertical-align:middle;
4557 xxx-background-color:#8af;
4558 background-color:#6881AD;xxx-PBlue;
4559 xxxheight:30px;
4560 }
4561 #gsh pre {font-size:11pt;color:#223;background-color:#ffff;}
4562 #gsh a {color:#24a;}
4563 #gsh a name {color:#24a;font-size:16pt;}
4564 #gsh .gsh-src {white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
4565 #gsh .gsh-src {background-color:#ffff;color:#223;}
4566 #gsh-src {spellcheck:false}
4567 #SrcTextarea {white-space:pre;font-family:Courier New;font-size:10pt;}
4568 #SrcTextarea {background-color:#ffff;color:#223;}
4569 .gsh-code {white-space:pre;font-family:Courier New !important;}
4570 .gsh-code {color:#024;font-size:11pt; background-color:#fafaf;}
4571 .gsh-golang-a {display:none;}
4572 #gsh-winId {color:#000;font-size:14pt;}
4573 }
4574 .gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}
4575 .gsh-document {color:#000;background-color:#fff !important;}
4576 .gsh-document > h2 {color:#000;background-color:#fff !important;}
4577 .gsh-document details {color:#000;background-color:#fff;font-family:Georgia;}
4578 .gsh-document #h2 {font-size:18pt;color:#000;background-color:#fff;font-family:Georgia;}
4579 .gsh-document address {width:500pt;color:#000;background-color:#fff;font-family:Georgia;}
4580 }
4581 @media print {
4582 #gsh pre {font-size:11pt !important;}
4583 }
4584 </style>
4585
4586 <!--
4587 // Logo image should be drawn by JavaScript from a meta-font.
4588 // CSS seems not follow line-splitlited URL
4589 -->
4590 <script id="gsh-data">
4591 //GshellLogo="08-ITS-more.jp.png"
4592 GShellLogo="data:image/png;base64,\
4593 IVB0W0Q0oAAJAAANhBhAAQAAABAA(CYAAADv3f4AAAXNSR0IARs4cQA0A8hWlE1\
4594 TU0AqAAAgAAABAAEAUAUAABAAABAAAgEBAUAAABAAABAAAgEBAUAAABAAABAAIADpAAQAAAB\
4595 AAATyAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAA\
4596 AAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAAB\
4597 x++t7uK3iCggoJ7y6o8bWgMzAvn7u4+biSTR7ynQXQdQcPKG2JabWLD2MS1RkEtArNoCdu\
4598 4iUx7jrIvY25ODOGm2VgIBEISgCoIMMA+mu+vu //ZM9D1da6a2aUbu91GRq3vvd6/g\
4599 fnVdx8rB8SIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
4600 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
4601 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
4602 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
4603 2eX9Y9ftSkSdms1C2q8E7Yus8Iqa1kfnY9oHMwEPd4K4QF5Ue8E1x15aV10\
4604 npDLKXEZCF1RM53JSU9q8Scq061+2K3StuONy5eEGK7Qw7mOvKec2ToQ1Lzwo1jhfS\
4605 jBOVHcSLMRB3USKEJ8hFu7DadmF2+XU4WVFXbBpMeZUAIE/hcKogab6eEKG0Lnyk56PC\
4606 Hkx2V8KorKqh3gUeK1YdaofON56kdi6w5BwomOQlyPz109DlXmpFR/6pZP/Plyovf\
4607 N8mf+/_NjM9j9g9GT0vN8r2z2r11gn3130V7YsoWwZcVrFRK7YfOaZLRSB9Q\
4608 zWocCG6qhvRacJ/dktj3g7dXh4gR6aRS0zPyZbqz6SRAOZDqfK79SKTRXhu/e9FN\
4609 L6as88pU/PNlp1TLQKSC73dFXS20ur7iLwPec8QhbnCyhl1ryTOTVf5IvqB7Jx\
4610 +c8H15j3ryr139b4d0E2QxFTFAPEZU0+1cKryhSF0E0W0gAxB30XMOlY\
4611 rikd9gHEF52Vq14h8F9A6kjYfFbQbnzLj4zF1eandHCVtoe1VQOB/5C9Y9D1LuoHR\
4612 +zGhU9n5GqrmouWguk19Rj7Bd4Y6uQcQd05TUW63z3MHesay14V91abDkxzbG1CpFR\
4613 U6t0AC7F9Y9F8B8DHT0MbaE74Ent+eWz+Hw/Lz/QT6w0d67QJjpe/OA7c00BNCEMUZ\
4614 tTCu/cooZfIpeK21PFF7JUR8AhhW8x81qaoeBvrfUd0+V0e8yD8Lct19K8KFAK0C\
4615 bgGov0glzG04jP392xnHhdc+MvF3Tj7fnt2yC1YJBJXNUSKIKycklaxXRd1d6BceVn\
4616 aJovy/VBacHvEP46/2lnJt9jx17V53215MLvapIQGLNrw5pQdQXNT012Zb8nGcM2ZV\
4617 qdOFJ8dVv0z28Yid6F3JCS4jXK9h17e2Zmp3TBLJpK1CpV1Hk/DJF04Dw1\
4618 1ImhG51R8z2pXw/C4HQ8F8zctY78zqPFBhHald2x5GK0CpWUhpogocw/8e\
4619 7/2doA8ptag/mlKJ77U0Vg0YtX/Ex/PTEA/177Ku+cSoCXDvRohUF16V9H+ccvG1\
4620 pd/CI/42AK2IUPVTKL/sdJyE5FVHg728NzVUzv0D0Gy9GooouhnmLmEctX48Y2gh\
4621 f/8hXV4/43ryr139b4d0E2QxFTFAPEZU0+1cKryhSF0E0W0gAxB30XMOlY\
4622 hrauce6WdKcJUZK8EUXMae71zUqCu2n16vnl19/P1e+IOMqof+H13JL5F8d91pA\
4623 NNW4rPy9jXxPeDL/HXNz2gTsvesLD2v8WHR19mu5rvvZ9x054V/LfmqdB1P8G1FM2UW\
4624 gJry2wE8R2zIeivd+Y2N0N7YrNyhG08R6o7RmMk1gQCJNRW5FpTm+EtWd4S10V\
4625 W1Ww1fLc0P4g2dR176/zs3y1l8p18W14wQu17i1K2KecOpw+K3J0g98RMAZ\
4626 IO2yuvn092z2x0DRdQVf5QxXt50WjRKAAM46pvTRXAJLJW7Y9Y/+CeU8MK168/zPQA\
4627 mCufKa2dFw/yI8A5IwC3dK1h9yZuCVSg/KRhwHwRRANM0D8EX+HF12A9b2d172\
4628 2gh0v0CDHmE80700C0yA0mner5sXVct3328F20w0w0U7A+Ed7\
4629 yHbUgm1X7f7KIDw9YfN42f1uXNDVtEanL65YCR2EVLbZAW2px5hfmshz3EM50k1K\
4630 d3/ZnabE1XPW0Xg1Yc5eW5/z8Gy54A0gWkfnWbqptceV74FUBvov32gew8DLd7Maj\
4631 aup17LBMX+Yw/egGKOtKy2d+gF8b9VodX5B1ZTORWfJy0p06U0XGOTRq/guta3vB\
4632 Fgeuqz2dVw8Dz3zLdbE4G8Fg10D0h5DMhbh8Am9yJdK1FZEMD3on7Vt5cC\
4633 h/YrC1p7Wp/VvurDuc+we54ym+8z2K0yRSRPA4IKoGz118b6ytagcEpm9v/m09cUATz\
4634 Jow6VnPCXh2j+sNph8Clyja6c8RMyrGk1w4F15Iou1LLRW7fMNL8z3z+/GFWLUZ\
4635 Y57268zKfY0PcJ315D1jYdRz0p1/3pmK0/y99A0y0FTI7mVx/6C0Hgh11uh/\
4636 F9V0g703p7z4F8P9w4F7xzhLW1yW6d21L1e/4uLmPw8A5u0cd0FC\
4637 ZFA6c0gxzhT6G41NR5D0j9Xw1cy+rFbcuVanLkV0CefhU0C1RMV19K4vng8GfCz\
4638 NCG5ICnckfexb+mf1BwuxF0ZqT/19425Yz2PCzPQWhbG2h8aJo/yb0kdkpnanG\
4639 CKWf56/KC5Mh2dpm0JegYF0w1y8m8tGCKEctiYuc3Lk4VY7Fepu55g8FKdX\
4640 uF9MfWIG38qNtX76+3zEX0WzVeqSPzMc2aFYSV461+04RyVgicCzGz2pYPrVveJ\
4641 02u12JWZEO+I6K0dLNXf2U9x70/bqzct520Poi0+vdpyDjcdxR34U9XCeHr1oSktt3ug\

```



```

6966 "3QrEqFmc/errSxliYeff7zqf4AAVCAhKEYif+Lk6jhtYF80H74VUkEhfF0MEzOk/jAg19"
6967 "kaP2/2nth0N0tE5y1jpk0EYVvM4wEaE6LJoc7VUa3650quXz2Mw7d/YWYent2x0v0W"
6968 "A4/3B7FD/gnuPfltc0PZV44zdc1jGPKXdbhYkYxuzG8NdpLmplaa575yYeuc18h1kX"
6969 "TJneq7qj3XKQkvbf4Q0H6Lxv2afNFZ4cbe0855hgmDpkzocGka6i/4Qw8ratpp5"
6970 "MlXka170x3j8yevetA1855y3j/4e08jF66qY8p9hV6d14VW+e+/FSc09aChk;jd"
6971 "1XXte1jHXzLlRwud5HhSAbkCzEps/5LZxH7jx0X0DI3HW/XYRDRdMv429m8yCHRScLFFZ"
6972 "r50w670G/cq0TuxClM2Oyo/q7Ybf8tEq/Lvn2G80JWR/vDzh1D05GyTXJ9Mx0MKwAw5fRE"
6973 "uColFL8eBz2fEkmy21vtLwz813H85Hh1jYWEYLaStAGh/+LEjdn5S687Dg8zF0o+mp4W"
6974 "L4H8c0h8q3c76G0M94L2q8e8780XzRQ8VxmGd3791Evdur8Ug7h0UMVv+TSVQa3kQr"
6975 "6Ze0VCOBJpNBIUeW8q61nNL/152pJfIUTR3+6rp47dx2B9fMxnXLMaxoT2bJg/WgrpnIa"
6976 "25kLwSgVcf3vU0Vf1p3jLxcl169Ajk0fxQ06bnjYGFbNl+GH8ebD0/3mpac7ncDjKTKz"
6977 "fivCAL+tl3dr/Scl+5H4A2Zcmpp0JhXf4ec/nL4l3mP8Aa03jP8HCUJ0CC0FP3jgdel"
6978 "n8qVAZVAwVlc/578h0x0211r0q8Y8vMz8U3Xj7azn/Bawq1fjhjQ813h1hEpl"
6979 "WifaeafKx+noQdzMK3YU0u1JONL5PnfaJlhn398N04x049HGKnlV88QNApoM+IBdT"
6980 "4eg8ppjcx186XmZLL0z0uP8uLNPXmewFtry/gkpZCzRRKHzLpdY/6rqr58vMPrhMx"
6981 "0lK5xYx3c76G0M94L2q8e8780XzRQ8VxmGd3791Evdur8Ug7h0UMVv+TSVQa3kQr"
6982 "swBUOafcoZ2MIP84YXJ3uXuppl12w6JdrPhOrdEp47Qd5nXlcF9Q8Wtpe7ootley6NG"
6983 "V6A+5aJ3KXblfhdQUB4j425K46aKeaLkSX8KtR7hobswLLx1K33ga0VzA2zY8a5+mLE"
6984 "YoeJ3cVFRk0ogj191zmi+maoKDLGG9WDD/KFN8M12aK00M0Jlvhgagfd88H4V4EcyIQrW0"
6985 "v3DUDeW83W0r70pA8ahH6LkykdrYmp65L8hLwvg8NAY0e1TnG88WQeCkCFRq"
6986 "ffdvRTE33AtvnrKuzLy8tOWQKupJbyVhHb1+L4aa06jwhhlegq300PxfYfstZbqu"
6987 "kg2ifVyh1jQW8btdk3+dvVH3W+G5N8yLlVyaXp106vyde7zro85nfdW8gu8Ew4T10/zA"
6988 "Lysaf1E8mdc2vW0r70pA8ahH6LkykdrYmp65L8hLwvg8NAY0e1TnG88WQeCkCFRq"
6989 "Xc0ALx+NT0xNQITeVjLacy/H0WVxahfV7n4maoJ10qpfjJjH0vX2Y89zq8SewF044"
6990 "174dGvXWYXMR6LcARoq8M5pA0P9F5U1R36ned9DA2BWTCThy10Tn6MkxUIKpt+UtuImXzgg"
6991 "tcyvgEjJGXMRJR16uvMlYgaur003Aig0LXG3268K/CTC8m069CT5Fbu+LZay1kk+g8B"
6992 "fTz7e7u4L/vT18h8h8f46/8K1ajfDh1VWY6G8m0y8XW619jDf+roq/088ay"
6993 "Xwed/3LlX68PTRCPIV3UHCrtyps/n5/BCUFMf1Hb5/P/D94D125t1E3AAAAAE1P7K5u"
6994 "QmC";
6995 </script>
6996
6997 <div id="GJFactory_1" class="xxxGJFactory" style=""></div>
6998 <!--
6999 htEpa://developer.mozilla.org/en-US/docs/Web/CSS/line-height
7000 -->
7001 <style>
7002 .GJFactory{
7003   resize:both; overflow:scroll;
7004   position:static;
7005   border:1.2px dashed #282; xborder-radius:2px;
7006   margin:0px; padding:10px 1important;
7007   width:340px; height:340px;
7008   flex-wrap: wrap;
7009   color:#fff; background-color:rgba(0,0,0,0.0);
7010   line-height:0.0;
7011   xxxcolor:#22a 1important;
7012   text-shadow:2px 2px #ddf;
7013 }
7014 .GJFactory h1,h2,h3,h4 {
7015   xxxcolor:#22a 1important;
7016 }
7017 xxxinput {
7018   border:1px dashed #f0; border-radius:0px;
7019 }
7020 .GJWin:hover{
7021   color:#f8f 1important;
7022   background-color:rgba(32,32,160,0.8) 1important;
7023   line-height:0.0;
7024 }
7025 .GJWin:active{
7026   color:#df8 1important;
7027   background-color:rgba(224,32,32,0.8) 1important;
7028   line-height:0.0;
7029 }
7030 .GJWin:focus{
7031   color:#df8 1important;
7032   background-color:rgba(32,32,32,1.0) 1important;
7033   line-height:0.0;
7034 }
7035 .GJWin{
7036   z-index:10000;
7037   display:inline;
7038   position:relative;
7039   flex-wrap: wrap;
7040   top:0; left:0px;
7041   width:285px 1important; height:205px 1important;
7042   border:1px solid #eaa; border-radius:2px;
7043   margin:0px; padding:0px;
7044   font-size:8pt;
7045   line-height:0.0;
7046   color:#fff; background-color:rgba(0,64,0.1) 1important;
7047 }
7048 .GJTab{
7049   display:inline;
7050   position:relative;
7051   top:0px; left:0px;
7052   margin:0px; padding:2px;
7053   border:0px solid #000; border-radius:2px;
7054   width:90px; height:20px;
7055   font-family:Georgia;
7056   font-size:9pt;
7057   line-height:1.0;
7058   white-space:nowrap;
7059   color:#fff; background-color:rgba(0,0,64,0.7);
7060   text-align:center;
7061   vertical-align:middle;
7062 }
7063 .GJStat:focus{
7064   color:#f8f 1important;
7065   background-color:rgba(32,32,32,1.0) 1important;
7066   line-height:1.0;
7067 }
7068 .GJStat{
7069   display:inline;
7070   position:relative;
7071   top:0px; left:0px;
7072   margin:0px; padding:2px;
7073   border:0px solid #00f; border-radius:2px;
7074   width:16px; height:20px;
7075   font-family:monospace;
7076   font-size:9pt;
7077   line-height:1.0;
7078   color:#fff; background-color:rgba(0,0,64,0.2);
7079   text-align:center;
7080   vertical-align:middle;
7081 }
7082 .GJIcon{
7083   display:inline;
7084   position:relative;
7085   top:0px; left:1px;
7086   border:2px solid #4a;
7087   margin:0px; padding:1px;
7088   width:13.2; height:13.2px;
7089   border-radius:2px;
7090   font-family:Georgia;
7091   font-size:13.2px;
7092   line-height:1.0;
7093   white-space:nowrap;
7094   color:#fff; background-color:rgba(32,32,160,0.8);
7095   text-align:center;
7096   vertical-align:middle;
7097   text-shadow:0px 0px;
7098 }
7099 .GJText:focus{
7100   color:#fff 1important;
7101   background-color:rgba(32,32,160,0.8) 1important;
7102   line-height:1.0;
7103 }
7104 .GJText{
7105   display:inline;
7106   position:relative;
7107   top:0px; left:0px;
7108   border:0px solid #000; margin:0px; padding:0px;
7109   width:280px; height:160px;
7110   border:0px;
7111   font-family:Courier New,monospace 1important;
7112   font-size:8pt;
7113   line-height:1.0;
7114   white-space:pre;
7115   color:#fff; xbackground-color:rgba(0,0,64,0.5);
7116   background-color:rgba(32,32,128,0.8) 1important;
7117 }
7118 .GJMode{
7119   display:inline;
7120   position:relative;
7121   top:0px; left:0px;
7122   border:0px solid #000; border-radius:0px;
7123   margin:0px; padding:0px;
7124   width:280px; height:20px;
7125   font-size:9pt;
7126   line-height:1.0;
7127   white-space:nowrap;

```

```

7128     color:#fff; background-color:rgba(0,0,64,0.7);
7129     text-align:left;
7130     vertical-align:middle;
7131 }
7132 </style>
7133
7134 <script id="gsh-script">
7135 // 2020-0909 added, permanent local storage
7136 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7137 var MyHistory = "";
7138 Permanent = localStorage;
7139 MyHistory = Permanent.getItem('MyHistory');
7140 if( MyHistory == null ){ MyHistory = "" }
7141 d = new Date()
7142 MyHistory = d.getTime()/1000+" "+document.URL+"\n" + MyHistory
7143 Permanent.setItem('MyHistory',MyHistory)
7144 //Permanent.setItem('MyWindow',window)
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     gjf = GJFactory
7167     //gjf = document.getElementById('GJFactory')
7168     //alert('gjf'+gjf)
7169     if( gjf != null ){
7170         if( gjf.childNodes != null ){
7171             for( i = 0; i < gjf.childNodes.length; i++ ){
7172                 gjf.removeChild(gjf.childNodes[i])
7173             }
7174             gjf.innerHTML = ''
7175             gjf.style.width = 0
7176             gjf.style.height = 0
7177             gjf.removeAttribute('style')
7178             GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
7179             window.clearInterval(FProductInterval)
7180             return '-- Destroy: work product destroyed'
7181         }else{
7182             return '-- Destroy: work product not exist'
7183         }
7184     }
7185 }
7186
7187 var TransMode = false
7188 var onKeyControl = false
7189 var OnKeyShift = false
7190 var OnKeyHit = false
7191 var OnKeyJ = 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         OnKeyHit = false
7205     }else
7206     if( keycode == 'KeyJ' ){ OnKeyJ = false }else
7207     if( keycode == 'KeyK' ){ OnKeyK = false }else
7208     if( keycode == 'KeyL' ){ OnKeyL = false }else
7209     {
7210     }
7211     ev.preventDefault()
7212 }
7213 function and(a,b){ if(a){ if(b){ return true; } return false; } }
7214 function GJWin_OnKeyDown(ev){
7215     keycode = ev.code;
7216     mode = '';
7217     key = '';
7218     if( keycode == 'ControlLeft' ){
7219         onKeyControl = true
7220         ev.preventDefault()
7221         return;
7222     }else
7223     if( keycode == 'ShiftLeft' ){
7224         OnKeyShift = true
7225         ev.preventDefault()
7226         return;
7227     }else
7228     if( keycode == 'AltLeft' ){
7229         ev.preventDefault()
7230         OnKeyHit = true
7231         return;
7232     }else
7233     if( keycode == 'Backquote' ){
7234         TransMode = !TransMode
7235         ev.preventDefault()
7236     }else
7237     if( and(keycode == 'Space', OnKeyShift) ){
7238         TransMode = !TransMode
7239         ev.preventDefault()
7240     }else
7241     if( keycode == 'ShiftRight' ){
7242         TransMode = !TransMode
7243     }else
7244     if( keycode == 'Escape' ){
7245         TransMode = true
7246         ev.preventDefault()
7247     }else
7248     if( keycode == 'Enter' ){
7249         TransMode = false
7250         //ev.preventDefault()
7251     }
7252     if( keycode == 'KeyJ' ){ OnKeyJ = true }else
7253     if( keycode == 'KeyK' ){ OnKeyK = true }else
7254     if( keycode == 'KeyL' ){ OnKeyL = true }else
7255     {
7256     }
7257
7258     if( ev.altKey ){ key += 'Alt+' }
7259     if( onKeyControl ){ key += 'Ctrl+' }
7260     if( OnKeyShift ){ key += 'Shift+' }
7261     if( and(keycode == 'KeyJ', OnKeyJ) ){ key += 'J+' }
7262     if( and(keycode == 'KeyK', OnKeyK) ){ key += 'K+' }
7263     if( and(keycode == 'KeyL', OnKeyL) ){ key += 'L+' }
7264     key += keycode
7265
7266     if( TransMode ){
7267         //mode = "\343\201\202r"
7268         JaUtf8 = new Uint8Array([0343,0201,0202]);
7269         utf8dec = new TextDecoder();
7270         JA = utf8dec.decode(JaUtf8);
7271         mode = "[" + JA + "r";
7272     }else{
7273         mode = '[---]'
7274     }
7275 }
7276 ///// /gmode.innerHTML = "[---]"
7277 GJWin_Mode.innerHTML = mode + '+' + key
7278 //alert('Key:'+keycode)
7279 ev.stopPropagation()
7280 //ev.preventDefault()
7281 }
7282 function GJWin_OnScroll(ev){
7283     x = DragStartX = gsh.getBoudingClientRect().left.toFixed(0)
7284     y = DragStartY = gsh.getBoudingClientRect().top.toFixed(0)
7285     GJLog_append('OnScroll: x='+x+',y='+y)
7286 }
7287 document.addEventListener('scroll',GJWin_OnScroll)
7288 function GJWin_OnResize(ev){
7289     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 // maybe this is the grabbing 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 = '3000'
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 = '3000'
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 GJ_ConsoleID = 0
7339 var 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 txl = '#'+p.id+' pw='+pw+', ph='+ph+'\n';
7351
7352 w = txa.getBoundingClientRect().width;
7353 h = txa.getBoundingClientRect().height;
7354 //txa.value += 'w='+w+', h='+h+'\n';
7355 txl += 'w='+w+', h='+h+'\n';
7356
7357 //txa.value += '\n';
7358 //txa.value += DateShort() + '\n';
7359 txl += '\n';
7360 txl += DateShort() + '\n';
7361 tmlap1 = new Date();
7362
7363 txa.value += txl;
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("StatBarUpdate:
7377 + 'length' + txa.value.length + ' byte, '
7378 + 'time' + (tmlap4 - tmlap0) + 'ms {
7379 + 'tadd' + (tmlap2 - tmlap1) + ', '
7380 + 'hcal' + (tmlap3 - tmlap2) + ', '
7381 + 'scri' + (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',360+'px'); //GJFsize
7403 GJFactory.style.setProperty('height',320+'px')
7404 e = GJFactory;
7405 console.log("GJFa #'+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('dragend',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 = wid
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 = '<Gfont color=#f44>J</font>'
7449 gjicon.setAttribute('class', 'GJIcon')
7450 gjicon.setAttribute('spellcheck', 'false')
7451 gjwin.appendChild(gjicon)

```

```

7452
7453 gJtext = GJLog_Text = document.createElement('textarea')
7454 gJtext.addEventListener('keydown',GJWin_OnKeyDown)
7455 gJtext.addEventListener('keyup',GJWin_OnKeyUp)
7456 gJtext.addEventListener('resize',GJWin_OnResizeTextarea)
7457 gJtext.id = wid + "_Text"
7458 gJtext.setAttribute('class','GJText')
7459 gJtext.setAttribute('spellcheck','false')
7460 gJwin.appendChild(gJtext)
7461
7462
7463 // user's mode as of IME
7464 gJmode = GJWin_Mode = document.createElement('textarea')
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 = gJwin.getBoundingClientRect().left.toFixed(0)
7480 //y = gJwin.getBoundingClientRect().top.toFixed(0)
7481 //gJwin.style.position = 'static'
7482 //gJwin.style.left = 0
7483 //gJwin.style.top = 0
7484
7485 //update = '{'+wid+'.value=DateShort()}',
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 }
7494 function xxxGJF_StripClass(){
7495 GJLog_Min.style.removeProperty('width')
7496 GJLog_Tab.style.removeProperty('width')
7497 GJLog_Stat.style.removeProperty('width')
7498 GJLog_Text.style.removeProperty('width')
7499 return "Stripped classes"
7500 }
7501 function isElem(id){
7502 return document.getElementById(id) != null
7503 }
7504 function GJLog_append(...args){
7505 txt = GJLog_Text;
7506 if( txt == null ){
7507 return; // maybe GJLog element is removed
7508 }
7509 logs = args.join(' ');
7510 txt.value += logs + '\n';
7511 txt.scrollTop = txt.scrollHeight;
7512 //GJLog_Stat.value = DateShort()
7513 }
7514 //window.addEventListener('time',GJLog_StatUpdate)
7515 function test_GJ_Console(){
7516 window.setInterval(GJLog_StatUpdate,1000);
7517 GJ_NewConsole('GJ_Console')
7518 e = GJFactory;
7519 console.log('GJFO #'+'e.id+' from w='+e.style.width+', h='+e.style.height)
7520 e.style.width = 360; //GJFsize
7521 e.style.height = 320;
7522 console.log('GJFO #'+'e.id+' to w='+e.style.width+', h='+e.style.height)
7523 }
7524 // test_GJ_Console();
7525
7526 var StopConsoleLog = true
7527 // 2020-09-15 added,
7528 // log should be saved to permanent memory
7529 // const px = new Proxy(console.log,{ alert() })
7530 __console_log = console.log
7531 __console_info = console.info
7532 __console_warn = console.warn
7533 __console_error = console.error
7534 __console_exception = console.exception
7535 // should pop callstack info.
7536 console.exception = function(...args){
7537 __console_exception(...args)
7538 alert('-- got console.exception('+args+')')
7539 }
7540 console.error = function(...args){
7541 __console_error(...args)
7542 alert('-- got console.error('+args+')')
7543 }
7544 console.warn = function(...args){
7545 __console_warn(...args)
7546 alert('-- got console.warn('+args+')')
7547 }
7548 console.info = function(...args){
7549 alert('-- got console.info('+args+')')
7550 __console_info(...args)
7551 }
7552 console.xxxlog = function(...args) { // rewrite xxxlog to log to enable it
7553 __console_log(...args)
7554 if( StopConsoleLog ){
7555 return;
7556 }
7557 if( 0 <= args[0].indexOf('l') ){
7558 //alert('-- got console.log('+args+')')
7559 }
7560 GJLog_append(...args)
7561 }
7562
7563 //document.getElementById('GshFaviconURL').href = GShellFavicon
7564 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
7565 //document.getElementById('GshFaviconURL').href = ITSMoreQR
7566 //document.getElementById('GshFaviconURL').href = GShellLogo
7567
7568 // id of GShell HTML elements
7569 var E_BANNER = "GshBanner" // banner element in HTML
7570 var E_FOOTER = "GshFooter" // footer element in HTML
7571 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7572 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7573 var E_TODO = "gsh-todo" // TODO of GShell
7574 var E_DICT = "gsh-dict" // Dictionary of GShell
7575
7576 function bannerElem(){ return document.getElementById(E_BANNER); }
7577 function bannerStyleFunc(){ return bannerElem().style; }
7578 var bannerStyle = bannerStyleFunc()
7579 function GshSetImages(){
7580 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7581 bannerStyle.backgroundImage = "url("+GShellLogo+")";
7582 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7583 //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
7584 //GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7585 //showFooter();
7586 }
7587 function GshInsideIconSetup(){
7588 GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7589 GMenu.style.zIndex = 10000000;
7590 //GMenu.style.left = window.innerWidth - 100
7591 GMenu.style.left = 0;
7592 GMenu.style.top = window.innerHeight - 90; // - 200
7593 window.addEventListener('resize',GshInsideIconSetup);
7594 }
7595
7596 function footerElem(){ return document.getElementById(E_FOOTER); }
7597 function footerStyleFunc(){ return footerElem().style; }
7598 //footerElem().style.backgroundImage="url("+ITSMoreQR+")";
7599 //footerStyle().backgroundImage = "url("+ITSMoreQR+")";
7600
7601 function html_fold(e){
7602 if( e.innerHTML == "Fold" ){
7603 e.innerHTML = "Unfold"
7604 document.getElementById('gsh-menu-exit').innerHTML=""
7605 document.getElementById('GshStatement').open=false
7606 GshFeatures.open = false
7607 document.getElementById('html-src').open=false
7608 document.getElementById(E_GINDEX).open=false
7609 document.getElementById(E_GOCODE).open=false
7610 document.getElementById(E_TODO).open=false
7611 document.getElementById('References').open=false
7612 }else{
7613 e.innerHTML = "Fold"

```

```

7614     document.getElementById('GshStatement').open=true
7615     GshFeatures.open = true
7616     document.getElementById(E_GINDEX).open=true
7617     document.getElementById(E_GOCODE).open=true
7618     document.getElementById(E_TOOO).open=true
7619     document.getElementById('references').open=true
7620 }
7621 }
7622 function html_pure(e){
7623     if (e.innerHTML == "Pure" ){
7624         document.getElementById('gsh').style.display=true
7625         //document.style.display = false
7626         e.innerHTML = "Unpure"
7627     }else{
7628         document.getElementById('gsh').style.display=false
7629         //document.style.display = true
7630         e.innerHTML = "Pure"
7631     }
7632 }
7633 }
7634 var bannerIsStopping = false
7635 //NOTE: .com/JSPREF/prop_style_backgroundposition.asp
7636 function shiftBanner(){
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/jspref/met_win_setinterval.asp
7663 var banNshift = 0;
7664 function consoleLog(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( !bannerIsStopping ){
7671         bannerStyle.backgroundPosition = bpos;
7672         //GshBanner.style.setProperty('background-position',bpos,'important');
7673         banNshift ++ 1;
7674         consoleLog('shiftBanner <'+GshBanner.nodeName+'> '+banNshift
7675             + ' now'+(now*10)
7676             + ' stop'+bannerIsStopping
7677             + ' pos'+bpos
7678             + ' -> '+bannerStyle.backgroundPosition);
7679     }
7680 }
7681 function Banner_init(){
7682     console.log('-- Banner Shift init. ');
7683     window.setInterval(shiftBanner,10); // oninit.
7684 }
7685
7686 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open(</a>
7687 // from embed html to standalone page
7688 var MyChildren = 0
7689 function html_fork(){
7690     ResetFrom();
7691     ResetAffView();
7692     Reset_ShadingCanvas();
7693     GJFactory_Destroy()
7694     MyChildren = 1
7695     WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
7696     newwin = window.open("",WinId,"");
7697     src = 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 checksum of HTML
7722 CRC32UNIX = 0x04C1D97 // Unix cksum
7723 function byteCRC32add(bigcrc,octstr,octlen){
7724     var crc = new Uint32Array(1)
7725     crc[0] = bigcrc
7726
7727     let oi = 0
7728     for( ; oi < octlen; oi++ ){
7729         var oct = new Uint8Array(1)
7730         oct[0] = octstr[oi]
7731         for( bi = 0; bi < 8; bi++ ){
7732             //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\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]+" "+oi+"/"+octlen+"\n")
7742     return crc[0];
7743 }
7744 function strCRC32add(bigcrc,stri,striLen){
7745     var crc = new Uint32Array(1)
7746     crc[0] = bigcrc
7747     var code = new Uint8Array(striLen);
7748     for( i = 0; i < striLen; i++){
7749         code[i] = stri.charCodeAt(i) // not charAt() !!!!
7750         //console.log("=== "+code[i].toString(16)+" <== "+stri[i]+" \n")
7751     }
7752     crc[0] = byteCRC32add(crc,code,striLen)
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 Uint8Array(4)
7760     let li = 0
7761     for( ; li < 4; ){
7762         li += 1
7763         slen[li] = len
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(stri,len){
7774     var crc = new Uint32Array(1)
7775     crc[0] = 0

```



```

7776   crc[0] = strCRC32add(0,stri,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_VirtualDesktop = function(){
7786 DestroyNavButtons = function(){
7787
7788 function getSourceText(){
7789   if( DestroyFooter != null ) DestroyFooter();
7790   version = document.getElementById('GshVersion').innerHTML
7791   sfavico = document.getElementById('GshFavicoURL').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_VirtualDesktop();
7799   GshTopbar.innerHTML = "";
7800   DestroyIndexBar();
7801   DestroyNavButtons();
7802   ResetPerMon();
7803   ResetAffView();
7804   Reset_ShadingCanvas();
7805
7806   // these should be removed by CSS selector or class, after seaved to non-printed attribute
7807   GshBanner.removeAttribute('style');
7808   document.getElementById('GshMenuSign').removeAttribute("style");
7809   styleGStat = GMenu.getAttribute("style");
7810   GMenu.removeAttribute("style");
7811   styleGStat = GStat.getAttribute("style");
7812   GStat.removeAttribute("style");
7813   styleGTop = GTop.getAttribute("style");
7814   GTop.removeAttribute("style");
7815   styleGshGrid = GshGrid.getAttribute("style");
7816   GshGrid.removeAttribute("style");
7817   //styleGPos = GPos.getAttribute("style");
7818   //GPos.removeAttribute("style");
7819   //GPos.innerHTML = "";
7820   //styleGLog = GLog.getAttribute("style");
7821   //GLog.removeAttribute("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   GshFavicoURL.href = "";
7831   if( !elem('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   GshMenuSign.style=""
7839
7840   textarea = document.createElement("textarea")
7841   srchtml = document.getElementById("gsh").outerHTML;
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 = srchtml
7846   // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7847   var rawtext = textarea.value
7848   //textarea.destroy()
7849   //rawtext = gsh.textContent // this removes #include <FILENAME> too
7850   var orgtext = ""
7851   + "/*<<">html>\n" // lost preamble text
7852   + rawtext
7853   + "<<">html>\n" // lost trail text
7854   ;
7855
7856   tlen = orgtext.length
7857   //console.log("getSourceText: length="+tlen+"\n")
7858   document.getElementById('GshFavicoURL').href = sfavico;
7859
7860   document.getElementById('GshBanner').style.backgroundImage = sbanner;
7861   document.getElementById('GshBanner').style.backgroundPosition = spositi;
7862
7863   GStat.setAttribute("style",styleGStat)
7864   GMenu.setAttribute("style",styleGMenu)
7865   GTop.setAttribute("style",styleGTop)
7866   //GLog.setAttribute("style",styleGLog)
7867   //GPos.setAttribute("style",styleGPos)
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 function getDigest(){
7877   var text = ""
7878   text = getSourceText()
7879   var digest = ""
7880   tlen = text.length
7881   digest = strCRC32(text,tlen) + " " + tlen
7882   return { text, digest }
7883 }
7884 function html_digest(){
7885   version = document.getElementById('GshVersion').innerHTML
7886   let {text, digest} = getDigest()
7887   alert("cksum: " + digest + " " + version)
7888 }
7889 function charsIn(str1,char){
7890   ln = 0;
7891   for( i = 0; i < str1.length; i++ ){
7892     if( str1.charCodeAt(i) == char.charCodeAt(0) )
7893       ln++;
7894   }
7895   return ln;
7896 }
7897
7898 //class digestElement extends HTMLElement {
7899 //< script>customElements.define('digest',digestElement)< /script>
7900 function showDigest(e){
7901   result = "version=" + GshVersion.innerHTML + '\n'
7902   result += "lines=" + e.dataset.lines + '\n'
7903   result += "length" + e.dataset.length + '\n'
7904   result += "crc32u" + e.dataset.crc32u + '\n'
7905   result += "time=" + e.dataset.time + '\n';
7906
7907   alert(result)
7908 }
7909
7910 function html_sign(e){
7911   if( RawTextViewer.style.zIndex == 1000 ){
7912     hideRawTextViewer()
7913     return
7914   }
7915   GshTopbar.innerHTML = "";
7916   ResetPerMon();
7917   ResetAffView();
7918   Reset_ShadingCanvas();
7919   DestroyIndexBar();
7920   DestroyNavButtons();
7921   DestroyEventSharingCodeview();
7922   Destroy_VirtualDesktop();
7923   GJFactory_Destroy()
7924   if( DestroyGJLink != null ) DestroyGJLink();
7925   //gsh_digest_innerHTML = "";
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,'\n')
7932
7933   name = "gsh"
7934   sid = name + "-digest"
7935   d = new Date()
7936   signedDt = d.getTime()
7937

```

```

7938 sign = '/' + '*' + 'span\n'
7939 + ' id="' + sid + '"\n'
7940 + ' class="digest"\n'
7941 + ' data-target-id="' + name + '"\n'
7942 + ' data-crc32="' + 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 txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>'
7951 + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
7952 for( i = 1; i <= Lines; i++) {
7953   txthtml += i.toString() + '\n'
7954 }
7955 txthtml += ""
7956 + '<' + '/textareas>'
7957 + '<' + '/td><' + 'td>'
7958 + '<' + 'textarea cols=150 rows=' + Lines + ' spellcheck="false"'
7959 + ' class="LineNumbered">'
7960 + text + '<' + '/textareas>'
7961 + '<' + '/td><' + '/tr><' + '/table>'
7962
7963 for( i = 1; i <= 30; i++) {
7964   txthtml += '<br>\n'
7965 }
7966 RawTextViewer.innerHTML = txthtml
7967 RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
7968
7969 btn = e
7970 e.style.color = "rgba(128,128,255,0.9)";
7971 y = e.getBoundingClientRect().top.toFixed(0)
7972 //h = e.getBoundingClientRect().height.toFixed(0)
7973 RawTextViewer.style.top = Number(y) + 30
7974 RawTextViewer.style.left = 100;
7975 RawTextViewer.style.height = window.innerHeight - 20;
7976 //RawTextViewer.style.Opacity = 1.0;
7977 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7978 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
7979 RawTextViewer.style.zIndex = 1000;
7980 RawTextViewer.style.display = true;
7981
7982 if( RawTextViewerClose.style == null ) {
7983   RawTextViewerClose.style = "";
7984 }
7985 RawTextViewerClose.style.top = Number(y) + 10
7986 RawTextViewerClose.style.left = 100;
7987 RawTextViewerClose.style.zIndex = 1001;
7988
7989 ScrollToElement(CurElement,RawTextViewerClose)
7990 }
7991 function hideRawTextViewer(){
7992   RawTextViewer.style.left = 10000;
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.display = 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 frame_open(){
8022   GshTopbar.innerHTML = "";
8023   ResetPerfMon();
8024   ResetAffView();
8025   ResetShadingCanvas();
8026   DestroyIndexBar();
8027   DestroyNavButtons();
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( iselem("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   //GLog.innerHTML = "";
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   DestroyVirtualDesktop();
8060   DestroyEventSharingCodeview();
8061
8062   src = document.getElementById("gsh");
8063   srchtml = src.outerHTML
8064   srcframe = document.getElementById("src-frame");
8065   srcframe.innerHTML = ""
8066   + "<"+ "cite id=\"GENSRC\">\n"
8067   + "<"+ "style>\n"
8068   + "#GENSRC textarea{tab-size:4;}\n"
8069   + "#GENSRC textarea(-o-tab-size:4;)\n"
8070   + "#GENSRC textarea(-moz-tab-size:4;)\n"
8071   + "#GENSRC textarea(spellcheck:false;)\n"
8072   + "<"+ "style>\n"
8073   + "<"+ "textarea id=\"SrcTextarea\" cols=100 rows=20 class=\"gsh-code\" spellcheck=\"false\">\n"
8074   + "<"+ "html>\n" // lost preamble text
8075   + srchtml
8076   + "<"+ "html>\n" // lost trail 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\">"
8090   + part.innerHTML
8091   + "<"+ "textarea>"
8092 }
8093 function fill_JavascriptView(){
8094   jspart = document.getElementById('gsh-script')
8095   view = document.getElementById('gsh-script-view')
8096   view.innerHTML = ""
8097   + "<"+ "textarea cols=100 rows=20 class=\"gsh-code\">"
8098   + jspart.innerHTML
8099   + "<"+ "textarea>"

```

```

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 jumpto_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 jumpto_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 jumpto_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 jumpto_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.backgroundColor;
8141 footer = document.getElementById('GshFooter').style.backgroundColor;
8142 document.getElementById('GshBanner').style.backgroundColor = "";
8143 document.getElementById('GshBanner').style.backgroundPosition = "";
8144 document.getElementById('GshFooter').style.backgroundColor = "";
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(-moz-tab-size:4;)\n");
8155 srcwin.document.write("</style>\n");
8156 srcwin.document.write("<h2>\n");
8157 srcwin.document.write("<+>span onclick="window.close();\>Close</span> | \n");
8158 //srcwin.document.write("<+>span onclick="html_stop();\>Run</span>\n");
8159 srcwin.document.write("<h2>\n");
8160 srcwin.document.write("<+>textarea id="gsh-src-src" cols=100 rows=60>");
8161 srcwin.document.write("</+>html>\n");
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.backgroundColor = banner;
8168 document.getElementById('GshFooter').style.backgroundColor = 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();
8182 srcwin.focus();
8183 }
8184 GSH = document.getElementById("gsh")
8185
8186 //GSH.onclick = "alert('Ouch!')";
8187 //GSH.css = "(background-color:#eef;";
8188 //GSH.style = "background-color:#eef;";
8189 //GSH.style.display = false;
8190 //alert('Ouch0!')
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 prevSib(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
8240 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
8241 function showCurElemPosition(ev){
8242     // if( document.getElementById("GPos") == null ){
8243     //     return;
8244     // }
8245     // if( GPos == null ){
8246     //     return;
8247     // }
8248     e = CurElement
8249     y = e.getBoundingClientRect().top.toFixed(0)
8250     x = e.getBoundingClientRect().left.toFixed(0)
8251     h = ev + " "
8252     h += "y="+y+", " + "x="+x+" -- "
8253     h += "w=" + window.innerWidth + ", h=" + window.innerHeight + " -- "
8254     //GPos.test = h
8255     //GPos.innerHTML = h
8256     // GPos.innerHTML = h
8257 }
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     //GShellIconSetup();
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     //}
8308     GStat.style.width = window.innerWidth
8309     //if( document.getElementById("GPos") != null ){
8310     //GPos.style.width = window.innerWidth
8311     //GPos.style.top = window.innerHeight - 30; //GPos.style.height
8312     //}
8313     //if( document.getElementById("GLog") != null ){
8314     // GLog.style.width = window.innerWidth
8315     //GLog.innerHTML = ""
8316     //}
8317     //if( document.getElementById("GLog") != null ){
8318     //GLog.innerHTML = "Resize: w=" + window.innerWidth +
8319     //"/", h=" + window.innerHeight
8320     //}
8321     showCurElementPosition(ev)
8322 }
8323 function GShellResize(){
8324     GShellResizeX("RESIZE")
8325 }
8326 window.onresize = GShellResize
8327 var prevNode = null
8328 var LogHouseMoveOverElement = false;
8329 function GJSH_OnMouseMove(ev){
8330     if( LogHouseMoveOverElement == false ){
8331         return;
8332     }
8333     x = ev.clientX
8334     y = ev.clientY
8335     d = new Date()
8336     t = d.getTime() / 1000
8337     if( document.elementFromPoint(x,y)
8338     e = document.elementFromPoint(x,y)
8339     if( e != null ){
8340         if( e == prevNode ){
8341             //}
8342             //}
8343             //}
8344             //}
8345             //}
8346             //}
8347             //}
8348             //}
8349             //}
8350             //}
8351             //}
8352             //}
8353             //}
8354             //}
8355             //}
8356             //}
8357             //}
8358             //}
8359             //}
8360             //}
8361             //}
8362             //}
8363             //}
8364             //}
8365             //}
8366             //}
8367             //}
8368             //}
8369             //}
8370             //}
8371             //}
8372             //}
8373             //}
8374             //}
8375             //}
8376             //}
8377             //}
8378             //}
8379             //}
8380             //}
8381             //}
8382             //}
8383             //}
8384             //}
8385             //}
8386             //}
8387             //}
8388             //}
8389             //}
8390             //}
8391             //}
8392             //}
8393             //}
8394             //}
8395             //}
8396             //}
8397             //}
8398             //}
8399             //}
8400             //}
8401             //}
8402             //}
8403             //}
8404             //}
8405             //}
8406             //}
8407             //}
8408             //}
8409             //}
8410             //}
8411             //}
8412             //}
8413             //}
8414             //}
8415             //}
8416             //}
8417             //}
8418             //}
8419             //}
8420             //}
8421             //}
8422             //}
8423             //}

```

```

8424 }else
8425 if( keycode == "Digit2" ){ // unfold side-bar
8426   primary.style.width = "58%"
8427   secondary.style.width = "36%"
8428   secondary.style.opacity = 1
8429   GStat.innerHTML = "[Double Column View]"
8430 }else
8431 if( keycode == "KeyU" ){ // fold/unfold all
8432   html_fold(GshMenuFold);
8433   location.href = "#"+CurElement.id;
8434 }else
8435 if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8436   CurElement.open = !CurElement.open;
8437 }else
8438 if( keycode == "ArrowRight" ){ // unfold the element
8439   CurElement.open = true
8440 }else
8441 if( keycode == "ArrowLeft" ){ // unfold the element
8442   CurElement.open = false
8443 }else
8444 if( keycode == "KeyI" ){ // inspect the element
8445   e = CurElement
8446   //GLog.innerHTML +=
8447   GLog.append("Current Element: " + e + "<br>"
8448     + "name="+e.nodeName + ", "
8449     + "id="+e.id + ", "
8450     + "children="+e.childNodes.length + ", "
8451     + "parent="+e.parentNode.id + "<br>"
8452     + "text="+e.textContent)
8453   GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8454   return
8455 }else
8456 if( keycode == "KeyM" ){ // memory the position
8457   MemElement = CurElement
8458 }else
8459 if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8460   e = nextSib(CurElement)
8461   if( e != null ){
8462     setColor(CurElement,"SUMMARY","#fff")
8463     setColor(e,"SUMMARY","#8f8") // should be complement ?
8464     oe = CurElement
8465     CurElement = e
8466     //location.href = "#"+e.id;
8467     scrollToElement(oe,e)
8468   }
8469 }else
8470 if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8471   oe = CurElement
8472   e = prevSib(CurElement)
8473   if( e != null ){
8474     setColor(CurElement,"SUMMARY","#fff")
8475     setColor(e,"SUMMARY","#8f8") // should be complement ?
8476     CurElement = e
8477     //location.href = "#"+e.id;
8478     scrollToElement(oe,e)
8479   }else{
8480     e = document.getElementById("GshBanner")
8481     if( e != null ){
8482       setColor(CurElement,"SUMMARY","#fff")
8483       CurElement = e
8484       scrollToElement(oe,e)
8485     }else{
8486       e = document.getElementById("primary")
8487       if( e != null ){
8488         setColor(CurElement,"SUMMARY","#fff")
8489         CurElement = e
8490         scrollToElement(oe,e)
8491       }
8492     }
8493   }
8494 }else
8495 if( keycode == "KeyR" ){
8496   location.reload()
8497 }else
8498 if( keycode == "KeyJ" ){
8499   GshGrid.style.top = '120px';
8500   GshGrid.innerHTML = '{>_}{Down}';
8501 }else
8502 if( keycode == "KeyX" ){
8503   GshGrid.style.top = '0px';
8504   GshGrid.innerHTML = '{>_}{Up}';
8505 }else
8506 if( keycode == "KeyH" ){
8507   GshGrid.style.left = '0px';
8508   GshGrid.innerHTML = '{_}{Left}';
8509 }else
8510 if( keycode == "KeyL" ){
8511   //GLog.innerHTML +=
8512   GLog.append(
8513     "screen="+screen.width+'px'+<br>'+
8514     "window="+window.innerWidth+'px'+<br>'+
8515     )
8516   GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8517   GshGrid.innerHTML = '{@_}{Right}';
8518 }else
8519 if( keycode == "Keys" ){
8520   html_stop(GshMenuStop,true)
8521 }else
8522 if( keycode == "KeyP" ){
8523   html_fork()
8524 }else
8525 if( keycode == "KeyC" ){
8526   window.close()
8527 }else
8528 if( keycode == "KeyD" ){
8529   html_digest()
8530 }else
8531 if( keycode == "KeyV" ){
8532   e = document.getElementById('gsh-digest')
8533   if( e != null ){
8534     showDigest(e)
8535   }
8536 }
8537 }
8538 showCurElementPosition("[ "+key.code+" ] --");
8539 //if( document.getElementById("GPos") != null ){
8540 //  GPos.innerHTML += "[ "+key.code+" ] --"
8541 //}
8542 //GShellResizeX("[ "+key.code+" ] --");
8543 }
8544 var initGSKC = false;
8545 function GShell_initKeyCommands(){
8546   if( initGSKC ){ return; } initGSKC = true;
8547 }
8548 GShellResizeX("[INIT]");
8549 DisplaySize = "-- Display: '
8550 + 'screen="+screen.width+'px, '+window.innerWidth+'px';
8551 }
8552 let {text, digest} = getDigest()
8553 //GLog.innerHTML +=
8554 GLog.append(
8555   "-- GShell: ' + GshVersion.innerHTML + '\n' +
8556   "-- Digest: ' + digest + '\n' +
8557   DisplaySize
8558 )
8559 //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8560 }
8561 GShellResizeX(null);
8562 //GShell_initKeyCommands();
8563 }
8564 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8565 //Convert a string into an ArrayBuffer
8566 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8567 function str2ab(str) {
8568   const buf = new ArrayBuffer(str.length);
8569   const bufView = new Uint8Array(buf);
8570   for (let i = 0, strLen = str.length; i < strLen; i++) {
8571     bufView[i] = str.charCodeAtAt(i);
8572   }
8573   return buf;
8574 }
8575 function importPrivateKey(pem) {
8576   const binaryDerString = window.atob(pemContents);
8577   const binaryDer = str2ab(binaryDerString);
8578   return window.crypto.subtle.importKey(
8579     "pkcs8",
8580     binaryDer,
8581     {
8582       name: "RSA-PSS",
8583       modulusLength: 2048,
8584       publicExponent: new Uint8Array([1, 0, 1]),
8585       hash: "SHA-256",

```

```

8586     },
8587     true,
8588     ["sign"]
8589 );
8590 }
8591 //importPrivateKey(ppem)
8592
8593 //key = {}
8594 //buf = "abc"
8595 //enc = "xyzxxxxx"; //crypto.publicEncrypt(key,buf)
8596 //b64 = btoa(enc)
8597 //dec = atob(b64)
8598 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8599 </script>
8600 */
8601
8602 <!-- ===== Work { ===== -->
8603 </span id="SightGlass_WorkCodeSpan">
8604 */
8605 <details id="SightGlass"><summary>SightGlass</summary>
8606 <!-- ===== SightGlass // 2020-1023 SatoziTS { -->
8607 <h2>SightGlass</h2>
8608 <details><summary>ometers</summary>
8609 <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8610 <div id="SightGlass_l_BGScroffset" class="SightGlass_TextData">(0000, 0000) </div>
8611 <div id="SightGlass_l_GPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8612 <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8613 <div id="SightGlass_l_Wheel" class="SightGlass_TextData">Wheel</div>
8614 </details>
8615 <p>
8616 <div id="SightGlass_l_Window" class="SightGlass_Window" draggable="true"></div>
8617 </p>
8618 <style>
8619 .SightGlass_TextData {
8620   color:#000;
8621   font-size:10pt;
8622 }
8623 .SightGlass_Window {
8624   xzoom:2;
8625   resize:both;
8626   width:600px;
8627   height:320px;
8628   background-color:rgba(200,200,200,0.5);
8629   background-size:200%;
8630   background-size:1280px 720px;
8631   background-image:url(WD-WallPaper03.png);
8632 }
8633 xbody {
8634   scroll-behavior:smooth;
8635 }
8636 </style>
8637 <script>
8638 //
8639 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
8640 var SGscriniX = 0;
8641 var SGscriniY = 0;
8642 var SG_initX = 0;
8643 var SG_initY = 0;
8644 function SG_adjust(){
8645   xy = window.screenX + ' ' + window.screenY;
8646   wh = window.innerWidth + ' x ' + window.innerHeight;
8647   xywh = 'xy(' + xy + ') wh(' + wh + ')';
8648   if( SightGlass.open) SightGlass_l_BPosition.innerHTML = 'Browser: ' + xywh;
8649
8650   sg = SightGlass_l_Window;
8651   sgr = sg.getBoundingClientRect();
8652   xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
8653   wh = sgr.width + ' x ' + sgr.height;
8654   xywh = 'xy(' + xy + ') wh(' + wh + ')';
8655   if( SightGlass.open) SightGlass_l_GPosition.innerHTML = 'SGLass: ' + xywh;
8656
8657   //SightGlass_l_Window.style.backgroundColor = sgr.left+'px ' + sgr.top+'px';
8658   if( SG_initX == 0 ){
8659     SGscriniX = window.screenX;
8660     SGscriniY = window.screenY;
8661     //SightGlass_l_Window.style.backgroundColor = '200%';
8662     //SightGlass_l_Window.style.backgroundColor = 'url("WD-WallPaper03.png")';
8663     SG_initX = sgr.left;
8664     SG_initY = sgr.top;
8665   }
8666   dx = SG_initX - sgr.left;
8667   dy = SG_initY - sgr.top;
8668
8669   dsx = SGscriniX - window.screenX;
8670   dsy = SGscriniY - window.screenY;
8671   scroff = 'Screen: '+dx+'dx'+',sy='+dsy;
8672   if( SightGlass.open) SightGlass_l_BGScroffset.innerHTML = scroff;
8673   dx += dsx;
8674   dy += dsy;
8675
8676   bgpos = dx+'px ' + dy+'px';
8677   SightGlass_l_Window.style.backgroundColor = bgpos;
8678   if( SightGlass.open) SightGlass_l_BGPosition.innerHTML = 'BGround:'
8679     + SightGlass_l_Window.style.backgroundColor;
8680 }
8681 var wheels = 0;
8682 function SG_wheel(e){
8683   wheels += 1;
8684   SightGlass_l_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
8685   if( e.target.id == 'SightGlass_l_Window' ){
8686     e.preventDefault();
8687   }
8688 }
8689 function SG_adjust1(){
8690   SG_adjust();
8691   //sampling = 0;
8692   sampling = 20;
8693   //sampling = 200;
8694   window.setTimeout(SG_adjust1,sampling);
8695 }
8696 function SightGlass_Setup(){
8697   SG_adjust();
8698   window.addEventListener('resize',SG_adjust);
8699   window.addEventListener('mousemove',SG_adjust);
8700   window.addEventListener('scroll',SG_adjust);
8701   window.addEventListener('wheel',SG_wheel,{passive:false});
8702   //window.moveTo(0,0);
8703
8704   // if focused
8705   //window.setInterval(SG_adjust,1);
8706   window.setTimeout(SG_adjust,1);
8707 }
8708 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
8709 function Electron_Setup(){
8710   const { BrowserWindow } = require('electron');
8711   const currentWindow = BrowserWindow.getFocusedWindow();
8712   //currentWindow.on('move',function(){ SG_adjust(); });
8713   currentWindow.addEventListener('move',SG_adjust);
8714 }
8715 </script>
8716
8717 <input id="SightGlass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8718 <input id="SightGlass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8719 <input id="SightGlass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8720 <span id="SightGlass_WorkCodeView"></span>
8721 <script id="SightGlass_WorkScript">
8722 function SightGlass_openWorkCodeView(){
8723   function SightGlass_showWorkCode(){
8724     showHtmlCode(SightGlass_WorkCodeView,SightGlass_WorkCodeSpan);
8725   }
8726   SightGlass_WorkCodeViewOpen.addEventListener('click',SightGlass_showWorkCode);
8727 }
8728 SightGlass_openWorkCodeView(); // should be invoked by an event
8729 </script>
8730 </details>
8731 <!-- SightGlass_WorkCodeSpan } -->
8732 */ </span>
8733 <!-- ===== Work } ===== -->
8734
8735
8736
8737 */
8738 <!-- ===== GJConsole BEGIN { ===== -->
8739 <span id="gjc" data-title="GJConsole" data-author="satofits-more.jp">
8740 <details><summary>GJ Console</summary>
8741 <p>
8742 <span id="GJE_RootNode0"></span>
8743 <span id="GJCI_Container"></span>
8744 </p>
8745 <style id="GJConsoleStyle">
8746 .GJConsole {
8747   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
8763   if( key == "Escape" ){
8764     SuppressGJShell = false
8765     GshMenu.focus() // should be previous focus
8766   }
8767 }
8768 var GJC_SessionId
8769 function GJC_SetSessionId(){
8770   var xd = new Date()
8771   GJC_SessionId = xd.getTime() / 1000
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+val
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   return ret
8805 }
8806 // -- 2020-09-14 added TableEditor
8807 var GJE_CurElement = null; //GJE_RootNode
8808 GJE_NodesSaved = 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     GJC_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 = GJC_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     GJC_CommandMode = true
8828   }else
8829   if( key == "KeyA" ){
8830     tab.style.caretColor = "red"
8831     GJC_CommandMode = false
8832   }else
8833   if( key == "KeyI" ){
8834     tab.style.caretColor = "red"
8835     GJC_CommandMode = false
8836   }else
8837   if( key == "KeyO" ){
8838     tab.style.caretColor = "red"
8839     GJC_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 == "KeyY" ){
8848     console.log("COL-FORW")
8849   }else
8850   if( key == "KeyB" ){
8851     console.log("COL-BACK")
8852   }
8853   kev.stopPropagation()
8854   if( wasCmdMode ){
8855     kev.preventDefault()
8856   }
8857 }
8858
8859 function GJE_DragEvent(ev,elem){
8860   x = ev.clientX
8861   y = ev.clientY
8862   console.log("Dragged: "+this.nodeName+"#"+this.id+ " x="+x+ " y="+y)
8863 }
8864 // https://developer.mozilla.org/en-US/docs/Web/API/Event/DragEvent
8865 // https://www.w3.org/TR/uievents/#events-mouseevents
8866 function GJE_DropEvent(ev,elem){
8867   x = ev.clientX
8868   y = ev.clientY
8869   this.style.x = x
8870   this.style.y = y
8871   this.style.position = 'absolute' // 'fixed'
8872   this.parentNode = gsh // just for test
8873   console.log("Dropped: "+this.nodeName+"#"+this.id+ " x="+x+ " y="+y
8874     + " parent="+this.parentNode.id)
8875 }
8876 function GJE_SetTableStyle(ev){
8877   this.innerHTML = this.value; // sync. for external representation?
8878   if(false){
8879     stid = this.parentNode.id+this.id
8880     // and remove "span" at the end
8881     e = document.getElementById(stid)
8882     //alert('SetTableStyle #' +e.id+' \n'+this.value)
8883     if( e != null ){
8884       e.innerHTML = this.value
8885     }else{
8886       console.log('Style Not found: '+stid)
8887     }
8888     //alert('event StopPropagtion: '+ev)
8889   }
8890 }
8891 function setCSSofclass(cclass,cstyle){
8892   const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
8893   rlen = ss.cssRules.length;
8894   let tabrule = null;
8895   rulex = -1
8896
8897   // should skip white space at the top of cstyle
8898   sel = cstyle.charAt(0);
8899   selector = sel+cclass;
8900   console.log('-- search style rule for '+selector)
8901
8902   for(let i = 0; i < rlen; i++){
8903     cr = ss.cssRules[i];
8904     console.log('css rule ['+'i'+"/'+rlen+'] '+'cr.selectorText);
8905     if( cr.selectorText === selector ){ // css class selector
8906       tabrule = ss.cssRules[i];
8907       console.log('CSS rule found for:['+'i'+"/'+rlen+'] '+'selector);
8908       ss.deleteRule(i);
8909       //rlen = ss.cssRules.length;

```

```

8910     rules = i
8911     // should search and replace the property here
8912   }
8913 }
8914 // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
8915 if( tabrule == null ){
8916   console.log('CSS rule NOT found for:['+rlen+'] '+selector);
8917   ss.insertRule(cssstyle,rlen);
8918   ss.insertRule(cssstyle,0); // override by 0?
8919   console.log('CSS rule inserted:['+(rlen+1)+']\n'+cssstyle);
8920 }else{
8921   ss.insertRule(cssstyle,rlen);
8922   ss.insertRule(cssstyle,0);
8923   console.log('CSS rule replaced:['+(rlen+1)+']\n'+cssstyle);
8924 }
8925 }
8926 function GJE_SetStyle(te){
8927   console.log('Apply the style to:'+te.id+'\n');
8928   console.log('Apply the style to:'+te.parentNode.id+'\n');
8929   console.log('Apply the style to:'+te.parentNode.class+'\n');
8930   cclass = te.parentNode.class;
8931   setCSSofClass(cclass,te.value); // should get selector part from
8932   // selector { rules }
8933 }
8934 if(false){
8935   //console.log('Apply the style:')
8936   //stid = this.parentNode.id+this.id+"
8937   //stid = this.id+"style"
8938   css = te.value
8939   stid = te.parentNode.id+"style"
8940   e = document.getElementById(stid)
8941   if( e != null ){
8942     //console.log('Apply the style:'+e.id+'\n'+te.value);
8943     console.log('Apply the style:'+e.id+'\n'+css);
8944     // e.innerHTML = css; //te.value;
8945     //ncss = e.sheet;
8946     //ncss.insertRule(te.value,ncss.cssRules.length);
8947   }else{
8948     console.log('No element to Apply the style: '+stid)
8949   }
8950   tblid = te.parentNode.id+"table";
8951   e = document.getElementById(tblid);
8952   if( e != null ){
8953     //e.setAttribute('style',css);
8954     e.setProperty('style',css,'important');
8955   }
8956 }
8957 }
8958 function makeTable(argv){
8959   //tid = ""
8960   //cwe = GJE_CurElement
8961   cwe = GJCI_Container;
8962   //cwd = GJFactory;
8963   tid = 'table_' + GJE_TableNo
8964   nt = new Text('\n')
8965   cwe.appendChild(nt)
8966   ne = document.createElement('span'); // the container
8967   cwe.appendChild(ne)
8968   ne.id = tid + '-span'
8969   ne.setAttribute('contenteditable',true)
8970   htspan = document.createElement('span'); // html part
8971   //htspan.id = tid + '-html'
8972   //ne.innerHTML = '\n'
8973   nt = new Text('\n')
8974   ne.appendChild(nt)
8975   ne.appendChild(htspan)
8976   htspan.id = tid
8977   htspan.setAttribute('class',tid)
8978   ne.setAttribute('draggable','true')
8979   ne.addEventListener('drag',GJE_DragEvent);
8980   ne.addEventListener('dragend',GJE_DropEvent);
8981   var col = 3
8982   var row = 2
8983   if( argv[0] != null ){
8984     col = argv[0]
8985     argv.shift()
8986   }
8987   if( argv[0] != null ){
8988     row = argv[0]
8989     argv.shift()
8990   }
8991   //ne.setAttribute('class',tid)
8992   ht = "\n"
8993   //ht += '<table id="'+tid+'"' + ' class="'+tid+'"'
8994   ht += '<table id="'+tid+'"' + ' class="'+tid+'"'
8995   // + ' onkeydown="GJE_TableKeyCommand(event,this)"'
8996   // + ' ondrag="GJE_DragEvent(event,this)"\n'
8997   // + ' ondragend="GJE_DropEvent(event,this)"\n'
8998   // + ' draggable="true"\n'
8999   // + ' contenteditable="true"'
9000   // + '>\n'
9001   ht += '<tbody>\n';
9002   for( r = 0; r < row; r++){
9003     ht += '<tr>\n'
9004     for( c = 0; c < col; c++){
9005       ht += '<td>'
9006       ht += "ABCDEFGHJKLMNPQRSTUVWXYZ".charAt(c) + r
9007       ht += "</td>\n"
9008     }
9009     ht += "</tr>\n"
9010   }
9011   ht += '</tbody>\n';
9012   ht += '</table>\n';
9013   htspan.innerHTML = ht;
9014   nt = new Text('\n')
9015   ne.appendChild(nt)
9016   st = '#'+tid+' *{\n' // # for instane specific
9017   st += 'border:1px solid #aaa;\n'
9018   st += 'background-color:#efe;\n'
9019   st += 'color:#222;\n'
9020   st += 'font-size:14pt !important;\n'
9021   st += 'font-family:monospace,Courier New !important;\n'
9022   st += 'hit ESC to apply *+\n'
9023 }
9024 // wish script to be included
9025 //nj = document.createElement('script')
9026 //ne.appendChild(nj)
9027 //ne.innerHTML = 'function SetStyle(e){}'
9028 // selector seems lost in dynamic style appending
9029 if(false){
9030   ns = document.createElement('style')
9031   ne.appendChild(ns)
9032   ns.id = tid + 'style'
9033   ns.innerHTML = '\n'+st
9034   nt = new Text('\n')
9035   ne.appendChild(nt)
9036 }
9037 setCSSofClass(tid,st); // should be in JavaScript script?
9038 nx = document.createElement('textarea')
9039 ne.appendChild(nx)
9040 nx.id = tid + '-style_def'
9041 nx.setAttribute('class','GJ_styleEditor')
9042 nx.spellcheck = false
9043 nx.cols = 60
9044 nx.rows = 10
9045 nx.innerHTML = '\n'+st
9046 nx.addEventListener('change',GJE_SetTableStyle);
9047 nx.addEventListener('keydown',GJE_StyleKeyCommand);
9048 //nx.addEventListener('click',GJE_SetTableStyle);
9049 nt = new Text('\n')
9050 cwe.appendChild(nt)
9051 GJE_TableNo += 1
9052 return 'created TABLE id="'+tid+'"'
9053 }
9054 function GJE_NodeEdit(argv){
9055   cwe = GJE_CurElement
9056   cmd = argv[0]
9057   argv.shift()
9058   args = argv.join(' ')
9059   ret = ""
9060 }

```



```

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

```

```

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   html_fork()
9243 }else
9244 if( cmd == 'last' ){
9245   text.value += MyHistory
9246   //h = document.createElement("span")
9247   //h.innerHTML = MyHistory
9248   //text.value += h.innerHTML
9249   //tx = MyHistory.replace("\n","")
9250   //text.value += tx.replace("<+>br>","\n") + "xxxxc<+>br>yyyy"
9251 }else
9252 if( cmd == 'ns' ){
9253   text.value += GJC_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   bannersIsStopping = 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
9278 function GJC_Input(){
9279   if( this.value.endsWith("\n") ){ // remove NL added by textarea
9280     this.value = this.value.slice(0,this.value.length-1)
9281   }
9282 }
9283
9284 var GJC_Id = null
9285 function GJC_Resize(){
9286   GJC_Id.style.zindex = 20000
9287   //GJC_Id.style.width = window.innerWidth - 16
9288   GJC_Id.style.width = "100%";
9289   GJC_Id.style.height = 300;
9290   GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9291   GJC_Id.style.color = "rgba(255,255,1.0)"
9292 }
9293 function GJC_FocusIn(){
9294   this.spellcheck = false
9295   SuppressGJShell = true
9296   this.onkeydown = GJC_Keydown
9297   GJC_Resize()
9298 }
9299 function GJC_FocusOut(){
9300   SuppressGJShell = false
9301   this.removeEventListener('keydown',GJC_Keydown);
9302 }
9303 window.addEventListener('resize',GJC_Resize);
9304
9305 function GJC_OnStorage(e){
9306   //alert('Got Message')
9307   //GJC.value += "\n((ReceivedMessage))\n"
9308 }
9309 window.addEventListener('storage',GJC_OnStorage);
9310 //window.addEventListner('storage',()=>{alert('GotMessage')})
9311
9312 function GJC_Setup(gjcId){
9313   //gjcId.style.width = gsh.getBoudingClientRect().width
9314   gjcId.style.width = "100%";
9315   gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
9316   //gjcId.value += "Date: " + DateLong() + "\n"
9317   gjcId.value += PSI
9318   gjcId.onfocus = GJC_FocusIn
9319   gjcId.addEventListener('input',GJC_Input);
9320   gjcId.addEventListener('focusout',GJC_FocusOut);
9321   GJC_Id = gjcId
9322 }
9323 function GJC_Clear(id){
9324 }
9325 function GJConsole_initConsole(){
9326   if( document.getElementById("GJC_0") != null ){
9327     GJC_Setup(GJC_0)
9328   }else{
9329     GJCl_Container.innerHTML = '<
9330       + "textarea id="GJC_1" class="GJConsole"><'+"/textarea>";
9331     GJC_Setup(GJC_1)
9332     factory = document.createElement('span');
9333     gsh.appendChild(factory)
9334     GJE_RootNode = factory;
9335     GJE_CurElement = GJE_RootNode;
9336   }
9337 }
9338 var initGJCF = false;
9339 function GJConsole_initFactory(){
9340   if( !initGJCF ){ return; } initGJCF = true;
9341   GShell_initKeyCommands();
9342   GJConsole_initConsole();
9343 }
9344 //GJConsole_initFactory();
9345 // TODO: focus handling
9346 </script>
9347 <style>
9348 .GJ_StyleEditor {
9349   font-size:9pt !important;
9350   font-family:Courier New, monospace !important;
9351 }
9352 </style>
9353 </details>
9354 </span>
9355 <!-- ----- GJConsole END ) ----- -->
9356 */
9357
9358 /*
9359 <span id="BlinderText">
9360 <style id="BlinderTextStyle">
9361 #GJLinkView {
9362   xxposition:absolute; z-index:5000;
9363   position:relative;
9364   display:block;
9365   left:8px;
9366   color:#fff;
9367   width:800px; height:300px; resize:both;
9368   margin:0px; padding:4px;
9369   background-color:rgba(200,200,200,0.5) !important;
9370 }
9371 .MsggText {
9372   width:578px !important;
9373   resize:both !important;
9374   color:#000 !important;
9375 }
9376 .GJNote {
9377   font-family:Georgia !important;
9378   font-size:1pt !important;
9379   color:#22a !important;
9380 }
9381 .textField {
9382   display:inline;
9383   border:0.5px solid #444;
9384   border-radius:3px;
9385   color:#000; background-color:#fff;
9386   width:106pt; height:18pt;
9387   margin:2px;
9388   padding:4px;
9389   resize:none;
9390   vertical-align:middle;
9391   font-size:10pt; font-family:Courier New;
9392 }
9393 .textLabel {
9394   border:0px solid #000 !important;
9395

```

```

9396     background-color:rgba(0,0,0,0);
9397 }
9398 .textURL {
9399     width:300pt !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:11pt;
9411     line-height:1.1;
9412     height:18pt;
9413     width:50pt;
9414     padding:0px !important;
9415     text-align:center !important;
9416     border-color:#aaa !important;
9417     border-radius:5px;
9418     color:#fff; background-color:#fa4 !important;
9419     vertical-align:middle !important;
9420 }
9421 .SendButton {
9422     vertical-align:top;
9423 }
9424 .ws0_log {
9425     font-size:10pt;
9426     color:#000 !important;
9427     line-height:1.0;
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
9435 <!-- Form autofill test
9436 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
9437 <form method="POST" id="xxform" action="https://192.168.10.1/boafm/formLogin">
9438 dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
9439 -->
9440 <details><summary>Form Auto. Filling</summary>
9441 <style>
9442 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9443     display:inline !important; font-size:10pt !important; padding:1px !important;
9444 }
9445 </style>
9446 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9447 <form Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
9448 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
9449 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
9450 Sessionid:<input id="xxsid" class="xxinput" name="session_id" type="text" size="80">
9451 <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
9452 </span>
9453 </script>
9454
9455 function XXSetFormAction(){
9456     xxform.setAttribute('action',xxserv.value);
9457 }
9458 xxform.setAttribute('action',xxserv.value);
9459 xxserv.addEventListener('change',XXSetFormAction);
9460 //xxserv.value = location.href;
9461 </script>
9462 </details>
9463 */
9464
9465 /*
9466 <details id="BlinderTextClass"><summary>BlinderText</summary>
9467 <span class="gsh-src">
9468 <span id="BlinderTextScript">
9469 // https://w3c.github.io/uievents/#event-type-keydown
9470 //
9471 // 2020-09-21 class BlinderText - textarea element not to be readable
9472 //
9473 // BlinderText attributes
9474 // bl_plainText - null
9475 // bl_hideChecksum - [false]
9476 // bl_showLength - [false]
9477 // bl_visible - [false]
9478 // data-bl_config - []
9479 // - min. length
9480 // - max. length
9481 // - acceptable charset in generate text
9482 //
9483 function BlinderChecksum(text){
9484     plain = text.bl_plainText;
9485     return strCRC32(plain,plain.length).toFixed(0);
9486 }
9487 function BlinderKeydown(ev){
9488     pass = ev.target
9489     if( ev.code == 'Enter' ){
9490         ev.preventDefault();
9491     }
9492     ev.stopPropagation()
9493 }
9494 function BlinderKeyUp(ev){
9495     blind = ev.target
9496     if( ev.code == 'Backspace' ){
9497         blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
9498     }
9499     else if( and(ev.code == 'Key', ev.ctrlKey) ){
9500         blind.bl_visible = !blind.bl_visible;
9501     }
9502     else if( and(ev.code == 'KeyL', ev.ctrlKey) ){
9503         blind.bl_showLength = !blind.bl_showLength;
9504     }
9505     else if( and(ev.code == 'KeyU', ev.ctrlKey) ){
9506         blind.bl_plainText = "";
9507     }
9508     else if( and(ev.code == 'KeyR', ev.ctrlKey) ){
9509         checksum = BlinderChecksum(blind);
9510         blind.bl_plainText = checksum; //.toString(32);
9511     }
9512     else if( ev.code == 'Enter' ){
9513         ev.stopPropagation();
9514         ev.preventDefault();
9515         return;
9516     }
9517     else if( ev.key.length != 1 ){
9518         console.log('KeyUp: '+ev.code+'/'+ev.key);
9519         return;
9520     }
9521     else if(
9522         blind.bl_plainText += ev.key;
9523     )
9524     leng = blind.bl_plainText.length;
9525     //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
9526     checksum = BlinderChecksum(blind) & 10; // show last one digit only
9527
9528     visual = '';
9529     if( !blind.bl_hideChecksum || blind.bl_showLength ){
9530         visual += ' ';
9531     }
9532     if( !blind.bl_hideChecksum ){
9533         visual += '#' + checksum.toString(10);
9534     }
9535     if( blind.bl_showLength ){
9536         visual += '/' + leng;
9537     }
9538     if( !blind.bl_hideChecksum || blind.bl_showLength ){
9539         visual += ' ';
9540     }
9541     if( blind.bl_visible ){
9542         visual += blind.bl_plainText;
9543     }
9544     else if(
9545         visual += '*'.repeat(leng);
9546     )
9547     blind.value = visual;
9548 }
9549 function BlinderKeyUp(ev){
9550     BlinderKeyUp(ev);
9551     ev.stopPropagation();
9552 }
9553 // https://w3c.github.io/uievents/#keyboardevent
9554 // https://w3c.github.io/uievents/#uievent
9555 // https://dom.spec.whatwg.org/#event
9556 function BlinderTextEvent(){
9557     ev = event;
9558     blind = ev.target;

```

```

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 //< textarea hidden id="gj_pass1"
9572 // class="textField BlinderText"
9573 // placeholder="PassWord1"
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',phold);
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(txa);
9590 }
9591 function DestroyBlinderText(txa){
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('VisualText#'+txa.id+' cols="'+cols'');
9630     }else{
9631         //console.log('VisualText#'+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(txa){
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 js = document.getElementById('BlinderTextScript');
9653 eval(js.innerHTML);
9654 //js.outerHTML = ""
9655 </script>
9656
9657 </span><!-- end of class="gsh-src" -->
9658 </details>
9659 </span>
9660 */
9661 //
9662 //<script id="GJLinkScript">
9663 function gjkey_hash(text){
9664     return strCRC32(text,text.length) % 0x10000;
9665 }
9666 function gj_addlog(e,msg){
9667     now = (new Date()).getTime() / 1000).toFixed(3);
9668     tstp = '['+now+'] '
9669     e.value += tstp + msg;
9670     e.scrollTop = e.scrollHeight;
9671 }
9672 }
9673 function gj_addlog_cl(msg){
9674     ws0_log.value += '(console.log) ' + msg + '\n';
9675 }
9676 var GJ_Channel = null;
9677 var GJ_Log = null;
9678 var gjk; // the global variable
9679 function GJ_Join(){
9680     target = gj_Join;
9681     if( target.value == 'Leave' ){
9682         GJ_Channel.close();
9683         GJ_Channel = null;
9684         target.value = 'Join';
9685         return;
9686     }
9687 }
9688 var ws0;
9689 var ws0_log;
9690
9691 sav_console_log = console.error
9692 console.error = gj_addlog_cl
9693 ws0 = new WebSocket(gj_serv.innerHTML);
9694 console.error = sav_console_log
9695
9696 GJ_Channel = ws0;
9697 ws0_log = document.getElementById('ws0_log');
9698 GJ_Log = ws0_log;
9699
9700 now = (new Date()).getTime() / 1000).toFixed(3);
9701 const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
9702 cst = wsstats[ws0.readyState];
9703 gj_addlog(ws0_log,stat +ws0.readyState+'('+cst+') : GJ Linked\n');
9704
9705 ws0.addEventListener('error', function(event){
9706     gj_addlog(ws0_log,stat error : transport error?\n');
9707 });
9708 ws0.addEventListener('open', function(event){
9709     GJLinkView.style.zIndex = 10000;
9710     //console.log('#'+GJLinkView.id+'.zIndex'+GJLinkView.style.zIndex);
9711     date1 = new Date().getTime();
9712     date2 = (date1 / 1000).toFixed(3);
9713     seed = date1.toString(16);
9714
9715     // user name and key
9716     user = document.getElementById('gj_user').value;
9717     if( user.length == 0 ){
9718         gj_user.value = 'nemo';
9719         user = 'nemo';

```

```

9720     }
9721     key1 = document.getElementById('gj_ukey').hl_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').hl_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     tstamp = 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         ofrom = argv[0];
9759         argv.shift(); // original from
9760     }
9761     argv.shift(); // command
9762     from = argv[0];
9763     argv.shift(); // from/to
9764     cmd1 = argv[0];
9765     argv.shift(); // xxxx command
9766
9767     if( false ){
9768         gj_addlog(ws0_log,'--'
9769             + ' tstamp='+tstamp
9770             + ',gjcmd='+gjcmd
9771             + ',from='+from
9772             + ',cmd='+cmd1+' '
9773             + '+argv+'\n');
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     }
9800 });
9801 ws0.addEventListener('close', function(event){
9802     if( GJ_Channel == null ){
9803         gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
9804         return;
9805     }
9806     GJ_Channel.close();
9807     GJ_Channel = null;
9808     target.value = 'Join';
9809     gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
9810 });
9811
9812 function GJ_BcastMessageUserPass(user,chan,msgbody){
9813     now = (new Date()).getTime() / 1000.toFixed(3);
9814     msg = now+ ' BCAST '+user+' '+chan+' '+msgbody;
9815     if( false ){
9816         gj_addlog(GJ_Log,'send '+msg+'\n');
9817     }
9818     GJ_Channel.send(msg);
9819 }
9820 function GJ_BcastMessage(msgbody){
9821     if( GJ_Channel == null ){
9822         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
9823         return;
9824     }
9825     //target = event.target;
9826     user = document.getElementById('gj_user').value;
9827     chan = document.getElementById('gj_chan').value;
9828     GJ_BcastMessageUserPass(user,chan,msgbody);
9829 }
9830 function GJ_SendMessageUserPass(user,chan,msgbody){
9831     now = (new Date()).getTime() / 1000.toFixed(3);
9832     msg = now+ ' ISM '+user+' '+chan+' '+msgbody;
9833     gj_addlog(GJ_Log,'send '+msg+'\n');
9834     GJ_Channel.send(msg);
9835 }
9836 function GJ_SendMessage(msgbody){
9837     if( GJ_Channel == null ){
9838         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
9839         return;
9840     }
9841     //target = event.target;
9842     user = document.getElementById('gj_user').value;
9843     chan = document.getElementById('gj_chan').value;
9844     GJ_SendMessageUserPass(user,chan,msgbody);
9845 }
9846 function GJ_Send(){
9847     msgbody = gj_sendText.value;
9848     GJ_SendMessage(msgbody);
9849 }
9850
9851 </script>
9852 <!-- ----- GJLINK ----- -->
9853 <!--
9854 - User can subscribe to a channel
9855 - A channel will be broadcasted
9856 - A channel can be a pattern (regular expression)
9857 - User is like From:(me) and channel is like To: or Recipient:
9858 - like VMWARE
9859 - watch message with SENDME, WATCH, CATCH, HEAR, or so
9860 - routing with path expression or name pattern (with routing with DNS like system)
9861 -->
9862 */
9863
9864 <<span id="GJLinkGolang">
9865 // <details id="GJwebsocket"><summary>Golang / JavaScript Link</summary>
9866 // <span class="gsh-src"><!-- { -->
9867 // 2020-0920 created
9868 // <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
9869 // <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
9870 // INSTALL: go get golang.org/x/net/websocket
9871 // INSTALL: sudo (apt,yum) install git (if git is not installed yet)
9872 // import "golang.org/x/net/websocket"
9873 const gshws_origin = "http://localhost:9999"
9874 const gshws_server = "localhost:9999"
9875 const gshws_port = 9999
9876 const gshws_path = "gjlink1"
9877 const gshws_url = "ws://" +gshws_server+"/" +gshws_path
9878 const GSHWS_MSGSIZE = (8*1024)
9879 func fmtString(fmts string, params ...interface{})(string){
9880     return fmt.Sprintf(fmts,params...)
9881 }

```

```

9882 func GSHWS_MARK(what string)(string){
9883     now := time.Now()
9884     us := fmtstring("%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 gchh(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.Print(GSHWS_MARK(what))
9900     fmt.Printf(fmts+"\n",params...)
9901 }
9902
9903 var WSV = []*websocket.Conn{}
9904 func jsend(argv []string){
9905     if len(argv) <= 1 {
9906         fmt.Printf("--i] %v [-m command arguments\n",argv[0])
9907         return
9908     }
9909     argv = argv[1:]
9910     if( len(WSV) == 0 ){
9911         fmt.Printf("--E]-- No link now\n")
9912         return
9913     }
9914     if( 1 < len(WSV) ){
9915         fmt.Printf("--i]-- 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     }
9921     argv = argv[1:]
9922 }
9923 args := strings.Join(argv, " ")
9924
9925 now := time.Now()
9926 msec := now.UnixNano() / 1000000;
9927 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9928 msg := fmtstring("%v SEND gshell* %v",tstamp,args)
9929
9930 if( multicast ){
9931     for i,ws := range WSV {
9932         wn,werr := ws.Write([]byte(msg))
9933         if( werr != nil ){
9934             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9935         }
9936         glog("SQ",fmtstring("(%v) %v",wn,msg))
9937     }
9938 }else{
9939     i := 0
9940     ws := WSV[i]
9941     wn,werr := ws.Write([]byte(msg))
9942     if( werr != nil ){
9943         fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9944     }
9945     glog("SQ",fmtstring("(%v) %v",wn,msg))
9946 }
9947 }
9948 func ws_broadcast(msg string)(wn int,werr error){
9949     for i,ws := range WSV {
9950         wn,werr := ws.Write([]byte(msg))
9951         if( werr != nil ){
9952             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9953         }
9954         glog("SQ",fmtstring("(%v) %v",wn,msg))
9955     }
9956     return wn,werr;
9957 }
9958 func serv(ws *websocket.Conn) {
9959     WSV = append(WSV,ws)
9960     //fmt.Print("\n")
9961     glog("CO", "accepted connections"+fmtstring("%v",len(WSV)))
9962     //remoteAddr := ws.RemoteAddr
9963     //fmt.Printf("-- accepted %v\n",remoteAddr)
9964     //fmt.Printf("-- accepted %v\n",ws.Config())
9965     //fmt.Printf("-- accepted %v\n",ws.Config().Header)
9966     //fmt.Printf("-- accepted %v // %v\n",ws,servl)
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",fmtstring("(%v,%v)",rn,rerr))
9973             break
9974         }
9975         req := string(reqb[0:rn])
9976         glog("SQ",fmtstring("(%v) %v",rn,req))
9977
9978         margv := strings.Split(req, " ");
9979         margv = margv[1:]
9980         if( 0 < len(margv) ){
9981             if( margv[0] == "RESP" ){
9982                 // should forward to the destination
9983                 continue;
9984             }
9985         }
9986         now := time.Now()
9987         msec := now.UnixNano() / 1000000;
9988         tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9989         res := fmtstring("%v "+"CAST"+ " %v",tstamp,req)
9990
9991         wn := 0;
9992         werr := error(nil);
9993         if( 0 < len(margv) && margv[0] == "CAST" ){
9994             wn, werr = ws_broadcast(res);
9995         }else{
9996             wn, werr = ws.Write([]byte(res))
9997         }
9998         gchh("SR",werr)
9999         glog("SR",fmtstring("(%v) %v",wn,string(res)))
10000     }
10001     glog("SF","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     WSV = wsv
10012     //glog("CO","closed %v",ws)
10013     glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
10014     ws.Close()
10015 }
10016 // url := [scheme://]host[:port][/path]
10017 func decomp_URL(url string){
10018 }
10019 func full_wsvURL(){
10020 }
10021 func gj_server(argv []string) {
10022     gjserv := gshws_url
10023     gjport := gshws_server
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             schemev := strings.Split(serv, "://")
10033             gjscheme = schemev[0]
10034             serv = schemev[1]
10035         }
10036         if( 0 < strings.Index(serv, "/") ){
10037             pathv := strings.Split(serv, "/")
10038             serv = pathv[0]
10039             gjpath = pathv[1]
10040         }
10041         serv := 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.Sprintf(servv[1], "%d", sport)
10049     }
10050     //glog("LC", "hostport=%v (%v : %v)", servv, host, port)
10051     gjport = fmt.Sprintf("%v:%v", host, port)
10052     gjserv = gjscheme + "://" + gjport + "/" + gjpath
10053 }
10054 glog("LS", fmt.Sprintf("listening at %v", gjserv))
10055 http.Handle("/"+gjpath, websocket.Handler(servi))
10056 err := error(nil)
10057 if( gjscheme == "was" ){
10058     // 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", fmt.Sprintf("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 := fmt.Sprintf("Hello, GShell! (%v)", qi)
10074         wn, werr := ws.Write([]byte(req))
10075         glog("QM", fmt.Sprintf("%v %v", wn, req))
10076         gchk("QE", werr)
10077         rn, rerr := ws.Read(resb)
10078         gchk("RE", rerr)
10079         glog("RM", fmt.Sprintf("%v %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 id="GJLink_1">
10092 <div id="GJLink_ServerSet"></div>
10093 <div>
10094 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10095 <span id="GJLink_Account"></span>
10096 </div>
10097 <div>
10098 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10099 <span id="GJLink_SendArea"></span>
10100 </div>
10101 <div id="ws0_log_container"></div>
10102 </span>
10103 </span>
10104 <script>
10105 function setupGJLinkArea(){
10106     GJLink_ServerSet.innerHTML = '<'+span id="gj_serv_label"
10107     + " class="textField textLabel">Server:<'+/span>
10108     + '<'+span id="gj_serv" class="textField TEXTURL" contenteditable><'+/span>;
10109
10110     GJLink_Account.innerHTML = '<'+textarea id="gj_user" class="textField"><'+/textarea'
10111     + '<'+textarea id="gj_ukey" class="textField"><'+/textarea'
10112     + '<'+textarea id="gj_chan" class="textField"><'+/textarea'
10113     + '<'+textarea id="gj_ckey" class="textField"><'+/textarea';
10114
10115     GJLink_SendArea.innerHTML =
10116     '<'+textarea id="gj_sendText" class="textField MasgText" cols=60 rows=2<'+/textarea';
10117
10118     ws0_log_container.innerHTML = '<'+textarea id="ws0_log" class="ws0_log"
10119     + " cols=100 rows=10 spellcheck="false"><'+/textarea';
10120 }
10121
10122 function clearGJLinkArea(){
10123     GJLink_ServerSet.innerHTML = "";
10124     GJLink_Account.innerHTML = "";
10125     GJLink_SendArea.innerHTML = "";
10126     ws0_log_container.innerHTML = "";
10127 }
10128 </script>
10129 <script>
10130 function SetupGJLink(){
10131     setupGJLinkArea();
10132     SetupVisibleText(GJLink_1, gj_serv, 'GJLinksv');
10133     SetupVisibleText(GJLink_1, gj_user, 'UserName');
10134     SetupBlinderText(GJLink_1, gj_ukey, 'UserKey');
10135     SetupVisibleText(GJLink_1, gj_chan, 'ChannelName');
10136     SetupBlinderText(GJLink_1, gj_ckey, 'ChannelKey');
10137     SetupVisibleText(GJLink_1, gj_sendtext, 'Message');
10138     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
10139 }
10140
10141 function GJLink_init(){
10142     SetupGJLink();
10143 }
10144
10145 function iselem(eid){
10146     return document.getElementById(eid);
10147 }
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( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10158     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10159     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10160     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10161     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10162     if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
10163     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10164     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10165 }
10166 DestroyGJLink = DestroyGJLink1;
10167 </script>
10168 //</details>
10169 /*
10170 <style>
10171 .GJDigest {
10172     display:none;
10173 }
10174 </style>
10175 <script id="HtmlCodeview-script">
10176 function showNodeAsHtmlSourceX(otxa, code, prefix, postfix, sign){
10177     txa = document.createElement('textarea');
10178     txa.id = otxa.id;
10179     txa.setAttribute('class', 'HtmlCodeviewText');
10180     otxa.parentNode.replaceChild(txa, otxa);
10181     txa.setAttribute('spellcheck', 'false');
10182     //txa.value = code.innerHTML;
10183     //txa.innerHTML = code.innerHTML;
10184     txa.innerHTML = prefix + code.outerHTML + postfix;
10185     if( sign ){
10186         text = txa.value;
10187         tlen = txa.value.length;
10188         digest = strCRC32(text, tlen) + ' ' + tlen
10189         + ' ' + code.id + ' (' + DateShort() + ')';
10190         //alert('digest: ' + digest);
10191         console.log(digest);
10192         txa.innerHTML += '<'+span class="GJDigest">'+digest+'<'+/span>\n';
10193     }
10194     txa.style.display = "block";
10195     txa.style.width = "100%";
10196     txa.style.height = "300px";
10197 }
10198
10199 function showHtmlCodeX(otxa, code, prefix, postfix, sign){
10200     if( event.target.value == "ShowCode" ){
10201         showNodeAsHtmlSourceX(otxa, code, prefix, postfix, sign);
10202     }
10203 }

```

```

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 showHtmlCode(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:2px 10px;
10233         line-height:1.1 !important;
10234         border:2px inset #bbb !important;
10235         font-size:10pt !important;
10236         font-weight:normal !important;
10237         font-family:Georgia !important;
10238         border-radius:3px !important;
10239         color:#dd; background-color:#228 !important;
10240     }
10241</style>
10242</script>
10243</script>
10244/*
10245<details><summary>Live HTML Snapshot</summary>
10246<span id="LiveHTML">
10247<!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
10248<div class="GshMenu">
10249<span class="GshMenu" onclick="html_edit();">Edit</span>
10250<span class="GshMenu" onclick="html_save();">Save</span>
10251<span class="GshMenu" onclick="html_load();">Load</span>
10252<span class="GshMenu" onclick="html_ver0();">Vers</span>
10253</div>
10254<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()">
10255<span id="LiveHTML_CodeView"></span>
10256</div>
10257<script id="LiveHTMLScript">
10258function showLiveHTMLCode(){
10259     showHtmlCode(LiveHTML_CodeView,LiveHTML);
10260 }
10261
10262var _editable = false;
10263var savSuppressGJShell = false;
10264function ToggleEditMode(){
10265     _editable = !_editable;
10266     if( _editable ){
10267         savSuppressGJShell = SuppressGJShell;
10268         SuppressGJShell = true;
10269         gsh.setAttribute('contenteditable','true');
10270         GshMenuEdit.innerHTML = 'Lock';
10271         GshMenuEdit.style.color = 'rgba(255,0,0,1)';
10272         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10273     }else{
10274         SuppressGJShell = savSuppressGJShell;
10275         gsh.setAttribute('contenteditable','false');
10276         GshMenuEdit.innerHTML = 'Edit';
10277         GshMenuEdit.style.color = 'rgba(16,160,16,1)';
10278         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10279     }
10280 }
10281function html_edit(){
10282     ToggleEditMode();
10283 }
10284
10285// Live HTML (DOM) Snapshot onto browser's localStorage
10286// 2020-0923 SatoxITS
10287var htRoot = gsh // -- Element-ID, should be selectable
10288const snappedHTML = "SnappedHTML"; // Item-ID of the HTML data in localStogate
10289 // -- should be a [map] of URL
10290 // -- should be with CSSOM as inline script
10291const htVersionTag = 'VersionTag'; // VersionTag Element-ID in the HTML (in DOM)
10292function showVersion(note,w,v,u,t){
10293     w.alert(note+ ' ' + v + '\n'
10294         + '-- URL: ' + u + '\n'
10295         + '-- Time: ' + t + ' (' + DateLong0(t+1000) + ') '
10296     );
10297 }
10298function html_save(){
10299     u = document.URL;
10300     t = new Date().getTime() / 1000;
10301     v = '<'+span id="+htVersionTag+" data-url="+u+" data-time="+t+">';
10302     v += '<'+/span>\n';
10303     h += v + htRoot.outerHTML;
10304     localStorage.setItem(snappedHTML,h);
10305     showVersion("Saved",window,v,u,t);
10306 }
10307function html_load(){
10308     h = localStorage.getItem(snappedHTML);
10309     if( h == null ){
10310         alert("No snapshot taken yet");
10311         return;
10312     }
10313     w = window.open('','','');
10314     d = w.document;
10315     d.write(h);
10316     w.focus();
10317     html_ver1("Loaded",w,d);
10318 }
10319function html_ver1(note,w,d){
10320     if( (v = d.getElementById(htVersionTag)) != null ){
10321         h = v.outerHTML;
10322         u = v.getAttribute('data-url');
10323         t = v.getAttribute('data-time');
10324     }else{
10325         h = 'No version info. in the page';
10326         u = '';
10327         t = 0;
10328     }
10329     showVersion(note,w,v,u,t);
10330 }
10331function html_ver0(){
10332     html_ver1("Version",window,document);
10333 }
10334</script>
10335<!-- LiveHTML ) -->
10336</span>
10337</details>
10338*/
10339</script>
10340/*
10341<details><summary>Event sharing</summary>
10342<span id="EventSharingCodeSpan">
10343<!-- ----- Event sharing // 2020-0925 SatoxITS { -->
10344<div id="lftestTemplate" class="lftest" hidden="">
10345<style> .lftestbody{ color:#f22;font-family:Georgia;font-size:10pt;} </style>
10346<span id="frameBody" class="lftestbody" onclick="frameClick()"><script>
10347function docadd(txt){
10348     document.body.append(txt);
10349     window.scrollTo(0,100000);
10350 }
10351function frameClick(){
10352     xy = 'x'+event.x + ' y'+event.y+'';
10353     //docadd('Got Click on #' +event.target.id+' '+xy+ '\n');
10354     docadd('Got Click on #' +Pid.value+' '+xy+ '\n');
10355     window.scrollTo(0,100000);
10356     window.parent.postMessage('OnClick: '+xy, '*');
10357 }
10358function frameMouseMove(){
10359     if( false ){
10360         document.body.append('Mousemove on #' +event.target.id+' '
10361             + 'x'+event.x + ' y'+event.y + '\n');
10362         peerWin = window.frames.iframe1;
10363         document.body.append('Send to peer #' +peerWin+' ' + '\n');
10364         window.scrollTo(0,100000);
10365         peerWin.postMessage('Hi!', '*');
10366     }
10367 }

```



```

10360 }
10361 }
10370 function frameKeyDown(){
10371     msg = 'Got Keydown: #' + Fid.value + ', (' + event.code + ')';
10372     docadd(msg + '\n');
10373     window.parent.postMessage(msg, '*');
10374 }
10375 function frameOnMessage(){
10376     docadd('Message ' + event.data + '\n');
10377     window.scrollTo(0,100000);
10378 }
10379 if ( document.getElementById('Fid') ){
10380     frameBody.id = Fid.value;
10381     h = '';
10382     h += '<'+style+'>';
10383     h += 'font-size:10pt;white-space:pre-wrap;';
10384     h += 'font-family:Courier New;';
10385     h += '<'+style+'>';
10386     h += ' I am '+Fid.value+'\n';
10387     document.write(h);
10388     window.addEventListener('click',frameClick);
10389     window.addEventListener('keydown',frameKeyDown);
10390     window.addEventListener('message',frameOnMessage);
10391     window.addEventListener('mousemove',frameMouseMove);
10392     window.parent.postMessage('Hi parent, I am '+Fid.value, '*');
10393 }
10394</script></span></div>
10395
10396<style>.iframeTest{margin:3px;resize:both;width:370px;height:120px}</style>
10397<div>Inter-window communication</div>
10398<note>
10399frame0 >>> frame1 and frame2<br>
10400frame1 >>> frame0 and frame2<br>
10401frame2 >>> frame0 and frame1<br>
10402</note>
10403<div id="iframe-test">
10404<pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
10405<iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
10406<iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
10407<iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
10408</div>
10409
10410<script id="ifo-test-script">
10411 function InterFrameComm_init(){
10412     setupFrames0();
10413     setupFrames12();
10414 }
10415 function setFrameSrcdoc(dst,src){
10416     if ( true ){
10417         dst.contentWindow.document.write(src);
10418         // this makes browser wait close, and crash if accumulated !?
10419         // so it should be closed after write
10420         dst.contentWindow.document.close();
10421     }else{
10422         // to be erased before source dump
10423         // but should be set for live snapshot
10424         dst.srcdoc = src;
10425     }
10426 }
10427 function setupFrames0(){
10428     ibody = iframe0.contentWindow.document.body;
10429     iframe0.style.width = "755px";
10430     //iframe0.innerHTML = "Message exchange at iframes' host:\n";
10431     window.addEventListener('message',messageFromChild);
10432 }
10433 if0 = '';
10434 if0 += '<'+pre style="font-family:Courier New;";
10435 if0 += '<input id="Fid" value="iframe0">';
10436 if0 += iftestTemplate.innerHTML;
10437 setFrameSrcdoc(iframe0,if0);
10438 }
10439 function clickOnChild(){
10440     console.log('clickOn #' + this.id);
10441 }
10442 function moveOnChild(){
10443     console.log('moveOn #' + this.id);
10444 }
10445 iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10446 iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10447 }
10448 function setupFrames12(){
10449     if1 = '<input id="Fid" value="iframe1">';
10450     if1 += iftestTemplate.innerHTML;
10451     setFrameSrcdoc(iframe1,if1);
10452     //iframe1.name = 'iframe1'; // this seems break contentWindow
10453 }
10454 if2 = '<input id="Fid" value="iframe2">';
10455 if2 += iftestTemplate.innerHTML;
10456 setFrameSrcdoc(iframe2,if2);
10457 }
10458 iframe1.addEventListener('message',messageFromChild);
10459 //iframe1.addEventListener('mouseover',moveOnChild);
10460 iframe2.addEventListener('message',messageFromChild);
10461 //iframe2.addEventListener('mouseover',moveOnChild);
10462 iframe1.contentWindow.postMessage([parent] Hi iframe1 -- from parent, '*');
10463 //iframe1.contentWindow.postMessage('Your peer is '+iframe2.contentWindow, '*');
10464 iframe2.contentWindow.postMessage([parent] Hi iframe2 -- from parent, '*');
10465 //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow, '*');
10466 }
10467 function messageFromChild(){
10468     from = null;
10469     forw = null;
10470     if( event.source == iframe0.contentWindow ){
10471         from = 'iframe0';
10472         forw = 'iframe12';
10473     }else
10474     if( event.source == iframe1.contentWindow ){
10475         from = 'iframe1';
10476         forw = 'iframe2';
10477     }else
10478     if( event.source == iframe2.contentWindow ){
10479         from = 'iframe2';
10480         forw = 'iframe1';
10481     }else
10482     {
10483         iframeHost.innerHTML += 'Message [unknown] '
10484         + ' orig=' + event.origin
10485         + ' data=' + event.data
10486         //+ ' from=' + event.source
10487         ;
10488     }
10489     msglog1 = from + event.data + ' -- '
10490     + ' from=' + event.source
10491     + ' orig=' + event.origin
10492     + ' name=' + event.source.name
10493     //+ ' port=' + event.ports
10494     //+ ' evid=' + event.lastEventId
10495     + '\n'
10496     ;
10497     if( true ){
10498         if( forw == 'iframe1' || forw == 'iframe12' ){
10499             iframe1.contentWindow.postMessage(from+event.data);
10500         }
10501         if( forw == 'iframe2' || forw == 'iframe12' ){
10502             iframe2.contentWindow.postMessage(from+event.data);
10503         }
10504     }
10505     txtadd0(msglog1);
10506 }
10507 function txtadd0(txt){
10508     iframe0.contentWindow.document.body.append(txt);
10509     iframe0.contentWindow.scrollTo(0,100000);
10510 }
10511 }
10512 function es_showSelf(){
10513     iframe1.setAttribute('src',document.URL);
10514     iframe2.setAttribute('src',document.URL);
10515 }
10516</script>
10517
10518<input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_showSelf()">
10519<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
10520<span id="EventSharingCodeview"></span>
10521<script id="EventSharingScript">
10522function es_showHtmlCode(){
10523    showHtmlCode(EventSharingCodeview,EventSharingCodePan);
10524}
10525DestroyEventSharingCodeview = function(){
10526    //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
10527    EventSharingCodeview.innerHTML = "";
10528    iframe0.style = "";
10529    //iframe0.srcdoc = "erased";

```

```

10530 //iframe1.srdoc = "erased";
10531 //iframe2.srdoc = "erased";
10532
10533</script>
10534<!-- EventSharing ) -->
10535</span>
10536</details>
10537*/
10538
10539/*
10540<!-- ----- "GShell Inside" Notification ( -->
10541<script id="script-gshell-inside">
10542var notices = 0;
10543function noticeGShellInside(){
10544    ver = '';
10545    if (ver = document.getElementById('GshVersion')) {
10546        ver = ver.innerHTML;
10547    }
10548    console.log('GShell Inside (~-~)'+ver);
10549    notices += 1;
10550    if (2 <= notices) {
10551        document.removeEventListener('mousemove',noticeGShellInside);
10552    }
10553}
10554document.addEventListener('mousemove',noticeGShellInside);
10555noticeGShellInside();
10556
10557const FooterName = 'GshFooter'
10558function 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}
10566function 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<!--
10577    border:20px inset #880;
10578-->
10579
10580</span id="VirtualDesktopCodeSpan">
10581/*
10582<details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
10583<!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS ( -->
10584<style>
10585.VirtualSpace {
10586    z-index:0;
10587    xwidth:1280px !important; xheight:720px !important;
10588    width:5120px; height:2880px;
10589    border-width:0px;
10590    xbackground-color:rgba(32,32,160,0.8);
10591    xbackground-image:url("WD-WallPaper03.png");
10592    xbackground-size:100% 100%;
10593    color:#22a;xfont-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 inset #fff;
10601    color:rgba(192,255,192,0.8);
10602    font-family:Georgia, Courier New;
10603    text-align:right;
10604    vertical-align:middle;
10605    font-size:200px;
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    min-width: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:100px;
10637    top:100px;
10638    display:block;
10639    resize:both !important;
10640    width:540px;
10641    height:320px;
10642    min-width:40px !important; min-height:20px !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:block;
10682    width:60px;
10683    height:20px;
10684    line-height:1.1;
10685    vertical-align:middle;
10686    font-size:14px;
10687    color:#fe4;
10688    background-color:rgba(128,128,128,0.1);
10689    font-family:Georgia;
10690    text-align:left;
10691    left:404px;

```

```

10692}
10693.VirtualBrowserFrame {
10694  xposition:relative;
10695  position:absolute;
10696  xdisplay:inline;
10697  display:block;
10698  z-index:10;
10699  resize:both !important;
10700  width:400px; height:240px;
10701  min-width:60px; min-height:30px;
10702  max-width:5120px; max-height:2880px;
10703  border-radius:4px;
10704  background-color:rgba(255,255,255,0.9);
10705  border-top:20px solid;
10706  border-right:4px solid;
10707  border-bottom:10px solid;
10708}
10709.WinFavicon {
10710  width:16px;
10711  height:16px;
10712  margin:1px;
10713  margin-right:3px;
10714  vertical-align:middle;
10715  background-color:rgba(255,255,255,1.0);
10716}
10717.VirtualDesktopMenuBar {
10718  xposition: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  xbackground-color:rgba(255,255,255,0.2);
10731}
10732.xxxxinput { width:160pt !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:2px !important;
10747  color:#fff !important;
10748  background-color:#228 !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 #44a !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.LiveHtmlCodeviewText {
10766  font-size:10pt;
10767  font-family:Courier New;
10768  xwhite-space:pre;
10769}
10770}
10771}
10772.WD_Panel {
10773  x-index:100 !important;
10774  color:#000 !important;
10775  margin-left:25px !important;
10776  width:800px !important;
10777  padding:4px !important;
10778  border:1px solid #998 !important;
10779  border-radius:10px !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 g ... grid on/off<br>
10800 l ... zoom in<br>
10801 o ... zoom out<br>
10802 s ... save current scope<br>
10803 r ... restore saved scope<br>
10804</span>
10805</menu>
10806<div class="WD_Panel" draggable="true">
10807<p><!-- should be on the frame of the WD -->
10808Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
10809 < <input id="WD_Width_1" class="WD_Config" type="text">
10810 x <input id="WD_Height_1" class="WD_Config" type="text">
10811 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
10812</p>
10813Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
10814 x <input id="WS_1_Width" class="WD_Config" type="text">
10815</p>
10816Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
10817 <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
10818 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
10819 <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
10820 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
10821</p>
10822Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
10823 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
10824 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
10825Shift+wheel for horizontal scroll
10826</p>
10827Scopex <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
10828 <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
10829 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
10830 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
10831</p>
10832Scopey <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
10833 <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
10834 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
10835 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
10836</p>
10837Scopez <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
10838 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
10839 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
10840 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
10841</p>
10842Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
10843</p>
10844Overflo <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">

```

```

10854 "scroll" imprisons windows inside the display
10855 </p>
10856 </div>
10857
10858 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
10859 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
10860 <i>CosmoScreen 0.0.8</i><span id="VirtualDesktop_1_Clock"></span>
10861 </div>
10862 <div id="VirtualDesktop_1_Calender" class="VirtualDesktopCalender" >00:00</div>
10863 <div align="right">VirtualSpace 1.</div>
10864 <div id="VirtualDesktop_1_Content" class="VirtualSpace">
10865
10866 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10867 <div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
10868 <span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
10869 <iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></iframe>
10870 </div>
10871
10872 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10873 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
10874 <span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
10875 <iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></iframe>
10876 </div>
10877
10878 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10879 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
10880 <span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
10881 <iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></iframe>
10882 </div>
10883
10884 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
10885 </div>
10886 </div>
10887
10888 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
10889 <span id="VirtualDesktopCodeview"></span>
10890 <script id="VirtualDesktopScript">
10891 function vd_showHtmlCode(){
10892     codespan = document.getElementById('VirtualDesktopCodeSpan');
10893     showHtmlCode(VirtualDesktopCodeview,codespan);
10894     VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
10895 }
10896 DestroyEventSharingCodeview = function(){
10897     VirtualDesktopCodeview.innerHTML = "";
10898 }
10899
10900 function wdlog(log){
10901     if( GJ_Channel != null ){
10902         GJ_SendMessage('WD '+log);
10903     }
10904     console.log(log);
10905 }
10906 var topMostWin = 10000;
10907 function onEnterWin(e){
10908     t = e.target;
10909     oindex = t.style.zIndex;
10910     //if( oindex == '' ) oindex = 0;
10911     //t.saved_zIndex = oindex;
10912     //t.style.zIndex = 10000;
10913     topMostWin ++ 1;
10914     t.style.zIndex = topMostWin;
10915     nindex = t.style.zIndex;
10916     wdlog('Enter '+t.#'+t.id+'('+oindex+'->'+nindex+')');
10917     e.stopPropagation();
10918     e.preventDefault();
10919 }
10920 function onClickWin(e){ // can detect click on the thick border? t = e.target;
10921     oindex = t.style.zIndex;
10922     topMostWin ++ 1;
10923     t.style.zIndex = topMostWin;
10924     nindex = t.style.zIndex;
10925     wdlog('Click '+t.#'+t.id+'('+oindex+'->'+nindex+')');
10926     //e.stopPropagation();
10927     //e.preventDefault();
10928 }
10929 function onLeaveWin(e){
10930     t = e.target;
10931     //oindex = t.style.zIndex;
10932     //nindex = t.saved_zIndex;
10933     //t.style.zIndex = nindex;
10934     //wdlog('Leave '+e.target.#'+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.wasATX = WinDragstartTX;
10968         t.wasATY = WinDragstartTY;
10969     }
10970     wdlog('DragSTA #' + 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         + ' event0('+WinDragstartX+', '+WinDragstartY+')'
10987         + ' event('+e.x+', '+e.y+')'
10988         + ' diff('+dx+', '+dy+')'
10989         + ' (' + nx + ', ' + ny + ' )'
10990         + ' (' + t.style.left + ', ' + t.style.top + ' )'
10991         + ' wasAt(' + t.wasATX + ', ' + t.wasATY + ' )'
10992     ;
10993     if( e.x != 0 || e.y != 0 ){
10994         if( set == true ){
10995             //t.style.x = 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         }else{
11001             log += ' NotSet';
11002             if( !dolog ){
11003                 log = '';
11004             }
11005         }
11006     }else{
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 propagated to parent in Firefox ?
11017 e.stopPropagation();
11018 }
11019 e.preventDefault();
11020 return false;
11021 }
11022 function onWinDrag(e){
11023   return onWinDragEvent('Ing',e,true,false);
11024 }
11025 function onWinDragend(e){
11026   return onWinDragEvent('End',e,false,true);
11027 }
11028 function onWinDragexit(e){
11029   return onWinDragEvent('Exit',e,false,true);
11030 }
11031 function onWinDragover(e){
11032   return onWinDragEvent('Over',e,false,true);
11033 }
11034 function onWinDragenter(e){
11035   return onWinDragEvent('Enter',e,false,true);
11036 }
11037 function onWinDragleave(e){
11038   return onWinDragEvent('Leave',e,false,true);
11039 }
11040 function onWinDragdrop(e){
11041   return onWinDragEvent('Drop',e,false,true);
11042 }
11043 function onFaviconChange(e){
11044   wdllog("--Favicon #" + e.target.id + " href="+e.details.href);
11045 }
11046 var savedSuppressGJShell = false;
11047 function stopGShell(e){
11048   //wdllog("enter Gsh STOP");
11049   savedSuppressGJShell = SuppressGJShell;
11050   SuppressGJShell = true;
11051   e.stopPropagation();
11052   e.preventDefault();
11053 }
11054 function contGShell(e){
11055   //wdllog("leave Gsh STOP");
11056   SuppressGJShell = savedSuppressGJShell;
11057   e.stopPropagation();
11058   e.preventDefault();
11059 }
11060
11061 function WD_onKeyDown(e){
11062   keycode = e.code;
11063   console.log("keydown #" + e.target.id + " " + keycode);
11064   if( keycode == 'KeyG' ){
11065     WD_setGrid1(WD_Grid1);
11066   }else
11067   if( keycode == 'KeyI' ){
11068     WD_doZoomIN();
11069   }else
11070   if( keycode == 'KeyO' ){
11071     WD_doZoomOUT();
11072   }else
11073   if( keycode == 'KeyR' ){
11074     WD_RestoreScope(null);
11075   }else
11076   if( keycode == 'Keys' ){
11077     WD_SaveScope(null);
11078   }
11079   e.stopPropagation();
11080   e.preventDefault();
11081 }
11082 function WD_onKeyUp(e){
11083   e.stopPropagation();
11084   e.preventDefault();
11085 }
11086 function WD_EventSetup(){
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 settleWin(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       wdllog("href_tag#" + href_id + "\n elem#" + href_id + "\n href="+d);
11099       url = d.innerHTML;
11100       wdllog("---- Load href_tag#" + href_id + "\n elem#" + href_id + "\n href="+d
11101         + "\n url="+url);
11102       wdllog("---- Load target #" + f.id + " with url="+url);
11103       f.src = url;
11104     }else{
11105       alert("unknown command" + command + " " + e.target.id + ", " + l.id + ", " + f.id);
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
11122 if( false ){
11123   wdllog("start settle VirtualBrowser url="+u + "\n"
11124     + "id" + s.id + "\n"
11125     + "width" + s.style.width + "\n"
11126     + "height" + s.style.height
11127   );
11128 }
11129 // very important for WordPress ??
11130 s.style.width = f.style.width = 501; // for WordPress ...??
11131 s.style.height = f.style.height = 271; // for WordPress ...??
11132 if( false ){
11133   wdllog("midway settle VirtualBrowser url="+u + "\n"
11134     + "id" + s.id + "\n"
11135     + "width" + s.style.width + "\n"
11136     + "height" + s.style.height
11137   );
11138 }
11139 s.width = 502; // for WordPress ...??
11140 s.height = 272; // for WordPress ...??
11141 if( false ){
11142   wdllog("midway-2 settle VirtualBrowser url="+u + "\n"
11143     + "id" + s.id + "\n"
11144     + "span-width" + s.width + "\n"
11145     + "span-height" + s.height
11146   );
11147 }
11148
11149 s.style.width = w + 'px';
11150 s.style.height = h + 'px';
11151 f.style.width = w + 'px';
11152 f.style.height = h + 'px';
11153 //f.style.setProperty('webkit-transform','scale('+scale+')');
11154 f.style.setProperty('transform','scale('+scale+')');
11155
11156 //wdllog("--x1-- u="+u" width s="+s.style.width+",f="+f.style.width);
11157 //wdllog("--x2-- u="+u" width s="+s.style.width+",f="+f.style.width);
11158 s.setAttribute('draggable','true');
11159 f.setAttribute('draggable','false'); // why necessary?
11160 l.setAttribute('draggable','false'); // why necessary?
11161 cmd.setAttribute('draggable','false'); // why necessary?
11162 s.addEventListener('dragstart', e => { onWinDragstart(e); });
11163 s.addEventListener('drag', e => { onWinDrag(e); });
11164 s.addEventListener('exit', e => { onWinDragexit(e); });
11165 s.addEventListener('dragend', e => { onWinDragend(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 s.style.left = x+'px';
11179 s.style.y = y+'px';
11180 s.style.top = y+'px';
11181 }else{
11182   s.style.setProperty('position','absolute','important');
11183   s.style.setProperty('x','x+'px','important');
11184   s.style.setProperty('left','x+'px','important');
11185   s.style.setProperty('y','y+'px','important');
11186   s.style.setProperty('top','y+'px','important');
11187 }
11188
11189 favicon = './favicon.ico';
11190 uv1 = u.split('/');
11191 if( 2 <= uv1.length ){
11192   uv2 = uv1[1].split('/');
11193   if( 2 <= uv2.length ){
11194     if( uv1[0] == 'file' ){
11195       //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/');
11196       //favicon = './favicon.ico';
11197       favicon = './favicon.ico';
11198     }else{
11199       favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
11200     }
11201   }
11202 }
11203 //wlog("----- favicon-url="+favicon);
11204 href_id = l.id + 'href';
11205 l.innerHTML = ''
11206 + '<a id="'+href_id+'" class="VirtualBrowserLocation" href="'+u+'>'+u+'</a>';
11207 //l.addEventListener('click', e => { onClickWin(e); });
11208 l.addEventListener('mouseenter', e => { stopGShell(e); });
11209 l.addEventListener('mouseleave', e => { contGShell(e); });
11210 l.addEventListener('keydown', e => { onKeyPress(e); });
11211 l.addEventListener('keyup', e => { onKeyUp(e); });
11212
11213 cmd.href_id = href_id;
11214 wlog('()cmd="#'+cmd.id);
11215 wlog('(1)href_id="'+href_id);
11216 wlog('(2)href_id'+cmd.href_id);
11217 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11218
11219 f.style.borderColor = c;
11220 f.src = u;
11221 //f.addEventListener('mouseenter', e => { onEnterWin(e); });
11222 //f.addEventListener('mouseleave', e => { onLeaveWin(e); });
11223
11224 //s.addEventListener('click', e => { onWinClick(e); });
11225 //f.addEventListener('click', e => { wlog('click wbl'); });
11226 f.addEventListener('mozbrowsericonchange', onFaviconChange);
11227
11228 wlog('done settle VirtualBrowser url="+u +'\n'
11229 + 'id' + s.id + '\n'
11230 + 'width' + s.style.width + ' '
11231 + 'height' + s.style.height + ' '
11232 + 'cmd' + cmd.id
11233 );
11234 }
11235
11236 function WD_EventSetup2(){
11237   dt = VirtualDesktop_1;
11238   dt.style.width = "800px";
11239   dt.style.height = "500px";
11240   dt.addEventListener('dragstart', e => { onWinDragstart(e); });
11241   dt.addEventListener('drag', e => { onWinDrag(e); });
11242   dt.addEventListener('exit', e => { onWinDragexit(e); });
11243 }
11244
11245 function GRonClick(){
11246   WD_SaveScope(null); // should be push
11247   t = event.target;
11248   x = t.getAttribute('data-leftx');
11249   y = t.getAttribute('data-topy');
11250   zoom = WD_Zoom_1_XY.value;
11251   x *= zoom;
11252   y *= zoom;
11253   WD_doScrollXY(event,x,y);
11254   //Alert('scroll #' + t.id + ' x='+x, y='+y);
11255 }
11256 function WD_setGrid(e){
11257   t = e.target;
11258   WD_setGrid(t);
11259 }
11260 function WD_setGrid(t){
11261   //ds = VirtualDesktop_1.Content; // should be VirtualSpace_1
11262   ds = VirtualDesktop_1_GridPlane;
11263   if( t.value == 'GridOn' ){
11264     for( col = 0; col < 16; col++ ){
11265       for( row = 0; row < 16; row++ ){
11266         gl = document.createElement('span');
11267         gl.setAttribute('class', 'VirtualGrid');
11268         leftx = col * 800;
11269         topy = row * 500;
11270         gid = col + '.' + row
11271         label = '<'+span'
11272 + 'id="'+gid+' '+class="WD_GridScroll" '
11273 + 'contenteditable="false" onclick="GRonClick()" '
11274 + 'data-leftx="'+leftx + ' ' + 'data-topy="'+topy + ' '
11275 + '>';
11276         e gid + '<'+span>';
11277         console.log('grid '+label);
11278         gl.innerHTML = label;
11279         gl.position = 'relative';
11280         gl.leftx = leftx;
11281         gl.topy = topy;
11282         gl.style.left = gl.leftx + 'px';
11283         gl.style.top = gl.topy + 'px';
11284         if( col % 2 == row % 2 ){
11285           gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
11286         }
11287         ds.appendChild(gl);
11288       }
11289     }
11290     t.value = 'GridOff';
11291   }else{
11292     ds.innerHTML = '';
11293     t.value = 'GridOn';
11294   }
11295 }
11296
11297 var sav_scrollLeft;
11298 var sav_scrollTop;
11299 var sav_nscale;
11300 function WD_SaveScope(e){
11301   sav_scrollLeft = WD_Left_1.value;
11302   sav_scrollTop = WD_Top_1.value;
11303   sav_nscale = WD_Zoom_1_XY.value;
11304   //console.log("saved zoom="+sav_oscale+', '+sav_nscale);
11305 }
11306 function WD_RestoreScope(e){
11307   WD_Zoom_1_XY.value = sav_nscale;
11308   WD_doZoom();
11309   WD_Left_1.value = sav_scrollLeft;
11310   WD_Top_1.value = sav_scrollTop;
11311   WD_doScroll(null);
11312 }
11313
11314 function ignoreEvent(e){
11315   e.stopPropagation();
11316   //e.preventDefault();
11317 }
11318 function zoomMag(){
11319   return WD_Zoom_1_MAG.value;
11320 }
11321 function WD_EventSetup3(){
11322   WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
11323   WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11324   WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
11325   WD_Width_1.value = dt.style.width;
11326   WD_Width_1.addEventListener('keydown', ignoreEvent);
11327   WD_Width_1.addEventListener('keyup', ignoreEvent);
11328   WD_Height_1.value = dt.style.height;
11329   WD_Height_1.addEventListener('keydown', ignoreEvent);
11330   WD_Height_1.addEventListener('keyup', ignoreEvent);
11331   WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
11332   WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
11333 }
11334
11335 function escale1(e,oscale,nscale){
11336   e.style.setProperty('transform','scale('+nscale+')');
11337   nscale = oscale / nscale;
11338   w = parseInt(e.style.width);
11339   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(WirtualBrowser_1,oscale,nscale);
11348     escale1(WirtualBrowser_1_Location,oscale,nscale);
11349     escale1(WirtualBrowser_1_Command,oscale,nscale);
11350     escale1(WirtualBrowser_1_Frame,oscale,nscale);
11351   }
11352   escale1(WirtualBrowser_2,oscale,nscale);
11353   escale1(WirtualBrowser_2_Location,oscale,nscale);
11354   escale1(WirtualBrowser_2_Command,oscale,nscale);
11355   escale1(WirtualBrowser_2_Frame,oscale,nscale);
11356 }
11357 escale1(WirtualBrowser_3,oscale,nscale);
11358 escale1(WirtualBrowser_3_Location,oscale,nscale);
11359 escale1(WirtualBrowser_3_Command,oscale,nscale);
11360 escale1(WirtualBrowser_3_Frame,oscale,nscale);
11361 }
11362 }
11363 function WD_doZoom(){
11364   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11365   oscale = WD_Zoom_1_XY.ouvalue; // 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.ouvalue = ds.style.zoom;
11370   scaleWD(ds,oscale,nscale);
11371 }
11372 function WD_EventSetup4(){
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 }
11378 function WD_doZoomOUT(){
11379   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11380   oscale = 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.ouvalue = ds.style.zoom;
11388   scaleWD(ds,oscale,nscale);
11389 }
11390 function WD_doZoomIN(){
11391   ds = WirtualDesktop_1_Content; // should be WirtualSpace_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.ouvalue = ds.style.zoom;
11404   scaleWD(ds,oscale,nscale);
11405 }
11406 }
11407 function WD_EventSetup5(){
11408   WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11409   WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11410 }
11411 function WD_doResize(e){
11412   dt = WirtualDesktop_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_doRResize(e){
11422   //alert('Resize Space');
11423   ds = WirtualDesktop_1_Content; // should be WirtualSpace_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_EventSetup6(){
11431   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11432   ds.style.width = '512px';
11433   ds.style.height = '2880px';
11434   WS_1_Width.value = ds.style.width;
11435   WS_1_Height.value = ds.style.height;
11436   WS_1_Width.addEventListener('keydown',ignoreEvent);
11437   WS_1_Height.addEventListener('keydown',ignoreEvent);
11438   WS_1_Height.addEventListener('keyup',ignoreEvent);
11439   WS_Resize_1.addEventListener('click',e => { WD_doRResize(e); });
11440 }
11441 }
11442 function WD_doScrollXY(e,sleft,stop){
11443   dt = WirtualDesktop_1;
11444   dt.scrollLeft = sleft;
11445   dt.scrollTop = stop;
11446   WD_Left_1.value = dt.scrollLeft;
11447   WD_Top_1.value = dt.scrollTop;
11448   console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
11449 }
11450 }
11451 function WD_doScroll(e){
11452   //dt = WirtualDesktop_1_Content;
11453   dt = WirtualDesktop_1;
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   if( false ){
11465     console.log(
11466       'wstop' + WirtualDesktop_1.style.top + ', ' +
11467       'wsx' + WirtualDesktop_1.style.y + ', ' +
11468       'wsse' + WirtualDesktop_1.scrollTop + ', ' +
11469       'wdtop' + WirtualDesktop_1_Content.style.top + ', ' +
11470       'wdkx' + WirtualDesktop_1_Content.style.y + ', ' +
11471       'wds' + WirtualDesktop_1_Content.scrollTop + ', '
11472     );
11473     WD_Left_1.value = WirtualDesktop_1.scrollLeft;
11474     WD_Top_1.value = WirtualDesktop_1.scrollTop;
11475 }
11476 }
11477 function WD_EventSetup7(){
11478   WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
11479   WD_Left_1.addEventListener('keydown',ignoreEvent);
11480   WD_Top_1.addEventListener('keydown',ignoreEvent);
11481   WD_Top_1.addEventListener('keyup',ignoreEvent);
11482 }
11483 }
11484 function WD_EventSetup8(){
11485   WirtualDesktop_1.addEventListener('scroll',showScrollPosition);
11486   WirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
11487 }
11488 }
11489 if( false ){
11490   w = 1000 + 'px';
11491   dt.style.width = w;
11492   dt.style.height = '300px';
11493   dt.style.resize = 'both';
11494   dt.style.borderWidth = 50 + 'px';
11495   dt.style.borderRadius = 25 + 'px';
11496   console.log('--2----- #' + 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.border);
11500 }
11501 }
11502 function onDRresize(e){
11503   dt = e.target;
11504   h = parseInt(dt.style.height);
11505   dt.style.borderWidth = (h * 0.075) + 'px';

```

```

11502 console.log('----- borderWidth'+dt.style.borderWidth);
11503
11504
11505 VirtualDesktopDetails.addEventListener('toggle',VirtualDesktop_init);
11506 function VirtualDesktop_init(){
11507     if (!VirtualDesktopDetails.open){
11508         return;
11509     }
11510     //GJ_Join();
11511     VirtualDesktop_1.addEventListener('resize', e => { onDTRize(e) });
11512     //console.log('----- #'+dt.id
11513     // + " borderWidth'+dt.style.getProperty('border-width' ));
11514
11515     settleWin(
11516         WirtualBrowser_1,
11517         WirtualBrowser_1_Location,
11518         WirtualBrowser_1_Command,
11519         WirtualBrowser_1_Frame,
11520         document.URL,
11521         500,280,50,20, #262',1.0);
11522     settleWin(
11523         WirtualBrowser_2,
11524         WirtualBrowser_2_Location,
11525         WirtualBrowser_2_Command,
11526         WirtualBrowser_2_Frame,
11527         'https://its-more.jp/ja_jp/',
11528         500,280,150,100, #444',1.0);
11529     settleWin(
11530         WirtualBrowser_3,
11531         WirtualBrowser_3_Location,
11532         WirtualBrowser_3_Command,
11533         WirtualBrowser_3_Frame,
11534         '//gshell/gsh.go.html',
11535         '//https://gshell.org/gshell/gsh.go.html',
11536         'https://golang.org',
11537         500,280,250,180, #444',1.0);
11538         //1000,720,0,0, #444',0.125);
11539         //1200,720,-100,-50, #444',0.4);
11540     function WD_ClockUpdate(e){
11541         WirtualDesktop_1_Clock.innerHTML = DateShort();
11542         WirtualDesktop_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_WirtualDesktop = function(){
11558     WirtualDesktop_1.style = "";
11559
11560     WirtualBrowser_1.removeAttribute('style');
11561     WirtualBrowser_1_Location.innerHTML = '';
11562     WirtualBrowser_1_Frame.removeAttribute('src');
11563     WirtualBrowser_1_Frame.removeAttribute('style');
11564     WirtualBrowser_1_Frame.style="";
11565
11566     WirtualBrowser_2.removeAttribute('style');
11567     WirtualBrowser_2_Location.innerHTML = '';
11568     WirtualBrowser_2_Frame.removeAttribute('src');
11569     WirtualBrowser_2_Frame.style="";
11570
11571     WirtualBrowser_3.removeAttribute('style');
11572     WirtualBrowser_3_Location.innerHTML = '';
11573     WirtualBrowser_3_Frame.removeAttribute('src');
11574     WirtualBrowser_3_Frame.style="";
11575
11576     GJFactory_1.style = "";
11577     iframe0.style = "";
11578     WirtualDesktop_1.style = "";
11579 }
11580
11581</script>
11582<!-- WirtualDesktop -->
11583</details>
11584*/ //</span>
11585
11586<!-- Work { ----- -->
11587<span id="SBSidebar_WorkCodeSpan">
11588*
11589<details><summary>SBSidebar</summary>
11590<!-- ----- SBSidebar // 2020-0928 SatoxITS ( -->
11591<h2>SBSidebar</h2>
11592<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11593<input id="SBSidebar_WorkCodeSignature" 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 id="SBSidebar_WorkScript">
11597function SBSidebar_openWorkCodeView(){
11598    function SBSidebar_showWorkCode(){
11599        showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
11600    }
11601    SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
11602}
11603SBSidebar_openWorkCodeView(); // should be invoked by an event
11604
11605console.log('-- Sbslider // 2020-1006-01 SatoxITS --');
11606function SetSidebar(){
11607    sidebar = document.getElementById('secondary');
11608    console.log('primary'+primary+' +secondary'+sidebar+' +main'+main' ');
11609    wrap = sidebar.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    //console.log('-- Sbslider parent is '+wrap, #' +wrap.id+' .'+wrap.class);
11615    nsb = sidebar.cloneNode();
11616    nsb = sidebar;
11617    nsb.style.width = '100%';
11618    slider = document.createElement('div');
11619    slider.id = 'Sbslider';
11620    slider.appendChild(nsb);
11621    slider.setAttribute('class','Sbslider');
11622    nsb.style.position = 'relative';
11623    slider.style.position = 'fixed';
11624    slider.style.display = 'block'; //inline';
11625    slider.style.zIndex = 100000;
11626    // nsb.style.zIndex = 200000;
11627    nsb.style.position = 'absolute';
11628    nsb.style.minWidth = '80px';
11629    nsb.style.left = '0px';
11630    nsb.style.top = '0px';
11631
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('-- added Sbslider');
11643    //nsb.addEventListener('scroll',SbScrolled);
11644
11645    buttons = document.createElement('div');
11646    buttons.id = 'NaviButtons';
11647    buttons.setAttribute('class','NaviButtons');
11648    buttons.align = "center";
11649    buttons.innerHTML += '<+>ca href="#TopOfPost" draggable="true">TOP<+>/a<+>/p';
11650    buttons.innerHTML += '<+>ca href="#EndOfPost" draggable="true">END<+>/p';
11651    page.appendChild(buttons);
11652    buttons.style.position = 'fixed';
11653    buttons.style.zIndex = 30000;
11654    buttons.style.width = '180px';
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
11660DestroyNaviButtons = function(){
11661    nb = document.getElementById('NaviButtons');
11662    if (nb != null){
11663        nb.parentNode.removeChild(NaviButtons);
11664    }

```



```

11664 }
11665 }
11666 </script>
11667
11668 // 2020-1006 its-more.jp-blog-60000-style.css
11669 <!-- {
11670 <style>
11671 #navButtons {
11672   position:fixed;
11673   display:block;
11674   xwidth:100%;
11675   xxtop:100px;
11676   xleft:10px;
11677   z-index:30000;
11678   font-size:10pt;
11679   color:#2ff !important;
11680   text-align:center;
11681   background-color:rgba(230,230,230,0.01);
11682 }
11683
11684 #navButtons a {
11685   color:#2a2 !important;
11686   font-size:20px;
11687   text-align:center;
11688   xtext-shadow:2px 2px #8ff;
11689   resize:both;
11690   padding:6px;
11691   margin:10px;
11692   border:1px solid #288 !important;
11693   border-radius:3px;
11694   background-color:rgba(160,160,160,0.05);
11695 }
11696 #sbsSlider {
11697   overflow:auto;
11698   resize:both !important;
11699   xxoverflow-y:hidden !important;
11700   height:100px !important;
11701   display:inline !important;
11702   position:fixed !important;
11703   left:0px;
11704   top:0px;
11705   xwidth:180px;
11706   width:24%;
11707   min-width:80px;
11708   height:100% !important;
11709   background-color:rgba(100,100,200,0.1);
11710 }
11711 #secondary {
11712   position:fixed;
11713   left:0px;
11714   top:0px;
11715   xxxz-index:60000;
11716   scroll-behavior: overflow !important;
11717   xxxxxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11718   padding-left:4pt;
11719   color:#ff;
11720   font-size:0.5em;
11721   background-color:rgba(64,160,64,0.6) !important;
11722   white-space:nowrap;
11723 }
11724 #secondary a {
11725   color:#fff !important;
11726   text-decoration:disable !important;
11727 }
11728 #primary {
11729   position:relative;
11730   width:75% !important;
11731   left:25% !important;
11732 }
11733 #main {
11734   position:relative;
11735   width:75% !important;
11736   left:25% !important;
11737 }
11738 #site-navigation {
11739   position:relative;
11740   left:120px;
11741 }
11742 #adswc_countertext {
11743   color:#169e1;
11744   font-size:16pt !important;
11745   xxfont-size:10 !important;
11746   font-weight:bold;
11747 }
11748 #nowTime {
11749   color:#0ffa0;
11750   font-size:16pt !important;
11751   xxfont-size:10 !important;
11752   font-weight:bold;
11753   text-shadow:1px 1px #fff;
11754 }
11755 .navigation-top {
11756   color:#2a !important;
11757   border:0px;
11758   background-color:rgba(220,220,220,0.1);
11759 }
11760 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11761   display: block;
11762   xxwidth: 1em;
11763   xxoverflow: auto;
11764   xxheight: 1em;
11765 }
11766 .invisible-scrollbar::-webkit-scrollbar {
11767   xdisplay: none;
11768   width:1px !important;
11769   height:1px !important;
11770 }
11771 .mostly-customized-scrollbar::-webkit-scrollbar {
11772   width: 2px;
11773   height: 2px;
11774   xbackground-color: #aaa; xxx:or add it to the track;
11775 }
11776 .mostly-customized-scrollbar::-webkit-scrollbar-thumb {
11777   background: #000;
11778 }
11779 </style>
11780 <!--
11781
11782 <details>
11783 <!-- $Sidebar_WorkCodeSpan } -->
11784 </span> //</span>
11785 <!-- Work } ----- -->
11786 <!-- Work { ----- -->
11787 </span id="Affiliate_WorkCodeSpan">
11788 </span id="Affiliate_Test"><summary>Affiliate</summary>
11789 <!-- ----- Affiliate // 2020-1010 SatoxZS { -->
11790 <div id="AffViewDock">
11791 <div id="AffView" class="AffView" draggable="true" style="">
11792 <div id="AffSet" class="AffPlate">
11793 <iframe id="aff_0" class="AffItem"></iframe>
11794 <iframe id="aff_1" class="AffItem"></iframe>
11795 <iframe id="aff_2" class="AffItem"></iframe>
11796 <iframe id="aff_3" class="AffItem"></iframe>
11797 <iframe id="aff_4" class="AffItem"></iframe>
11798 <iframe id="aff_5" class="AffItem"></iframe>
11799 </div>
11800 <div><div><div>Supportive Affiliate</h2>
11801 <style>
11802 .AffView {
11803   z-index:0;
11804   overflow-x:scroll;
11805   overflow-y:scroll;
11806   position:fixed;
11807   max-width:2500px;
11808   max-height:100%;
11809   width:270px;
11810   left:75%;
11811   height:95%;
11812   resize:both;
11813   margin-top:40px;
11814   xleft:0;
11815   xxalign:right;
11816   display:block;
11817   border:4px inset rgba(255,255,255,0.1);
11818   background-color:rgba(255,255,255,0.1);
11819 }
11820 }

```

```

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 #888;
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:2800px;
11842  margin-top:10px;
11843  margin-bottom:10px;
11844  margin-left:4px;
11845  width:300px;
11846  xheight:1440px;
11847}
11848.AffItem {
11849  overflow-x:visible;
11850  overflow-y:scroll;
11851  max-width:2560px;
11852  max-height:1440px;
11853  z-index:0;
11854  display:block;
11855  xposition:fixed;
11856  xposition:absolute;
11857  position:relative;
11858  //left:300px;
11859  xresize:both;
11860  padding:0px;
11861  width:600px;
11862  height:400px;
11863  max-height:800px !important;
11864  margin-top:0%;
11865  margin-left:0%;
11866  margin-right:0% !important;
11867  border:16px inset #ccc;
11868  transform:scale(0.5);
11869  background-color:rgba(255,255,255,0.2);
11870  xalign:right;
11871}
11872.AffItem: hover {
11873  border:16px inset #bbf;
11874}
11875</style>
11876<input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11877<input id="Affiliate_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11878<input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11879<span id="Affiliate_WorkCodeView"></span>
11880<script id="Affiliate_WorkCodeView">
11881function Affiliate_openWorkCodeView(){
11882  function Affiliate_showWorkCode(){
11883    showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
11884  }
11885  Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
11886}
11887Affiliate_openWorkCodeView(); // should be invoked by an event
11888
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>
11891var Aff_isSetup = false;
11892Affiliate_Test.addEventListener('click',Aff_Setup);
11893function 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
11900  var off = 100;
11901  zoom = 0.5;
11902  ozoom = 0.3;
11903  leftx = window.innerWidth - 300;
11904  left = leftx + 'px';
11905  left = -120 + 'px';
11906  console.log('aff-init window.innerWidth='+window.innerWidth);
11907  w = 1000;
11908  h = 560;
11909
11910  aff_0.src='./gshell/gsh.go.html';
11911  aff_1.src='https://golang.org';
11912  aff_2.src='https://drafts.csswg.org/';
11913  aff_3.src='https://html.spec.whatwg.org/dev/';
11914  aff_4.src='https://wikipedia.org';
11915  aff_5.src='https://www.bing.com/translator';
11916
11917  //parent.appendChild(aff_0);
11918  aff_0.style.width = zoom*w+'px';
11919  aff_0.style.height = zoom*h+'px';
11920  aff_0.style.left = left;
11921  //aff_0.style.top = off+'px'; off += ozoom*h;
11922  aff_0.draggable = 'true';
11923
11924  //parent.appendChild(aff_1);
11925  aff_1.style.width = zoom*w+'px';
11926  aff_1.style.height = zoom*h+'px';
11927  aff_1.style.left = left;
11928  //aff_1.style.top = off+'px'; off += ozoom*h;
11929  aff_1.style.top = '-150px';
11930  aff_1.draggable = 'true';
11931
11932  //parent.appendChild(aff_2);
11933  aff_2.style.width = zoom*w+'px';
11934  aff_2.style.height = zoom*h+'px';
11935  aff_2.style.left = left;
11936  //aff_2.style.top = off+'px'; off += ozoom*h;
11937  aff_2.style.top = '-300px';
11938  aff_2.draggable = 'true';
11939
11940  //parent.appendChild(aff_3);
11941  aff_3.style.transform = 'scale(0.25)';
11942  aff_3.style.width = 2*zoom*w+'px';
11943  aff_3.style.height = 2*zoom*h+'px';
11944  aff_3.style.border = '32px inset #ccc';
11945  //aff_3.style.left = -390 + 'px'; //left*2;
11946  //aff_3.style.left = (leftx - 265) + 'px';
11947  aff_3.style.left = -395 + 'px';
11948  //aff_3.style.top = (-155+off)+'px'; off += ozoom*h;
11949  aff_3.style.top = '-600px';
11950  aff_3.draggable = 'true';
11951
11952  //parent.appendChild(aff_4);
11953  aff_4.style.width = zoom*w+'px';
11954  aff_4.style.height = zoom*h+'px';
11955  aff_4.style.left = left;
11956  //aff_4.style.top = off+'px'; off += ozoom*h;
11957  aff_4.style.top = '-900px';
11958  aff_4.draggable = 'true';
11959
11960  //parent.appendChild(aff_5);
11961  aff_5.style.transform = 'scale(0.300)';
11962  aff_5.style.width = zoom*(w*1.67)+'px';
11963  aff_5.style.height = zoom*(h*1.67)+'px';
11964  aff_5.style.border = '28px inset #ccc';
11965  aff_5.style.left = -308+'px';
11966  //aff_5.style.left = (-175+leftx)+'px';
11967  //aff_5.style.top = (-95+off)+'px'; off += ozoom*h;
11968  //aff_5.style.left = '0px';
11969  //aff_5.style.top = '0px';
11970  aff_5.style.top = '-150px';
11971  //aff_5.style-align = 'right';
11972  aff_5.draggable = 'true';
11973
11974  window.addEventListener('resize',affresize);
11975}
11976function affresize(){
11977  AffView.style.left = (window.innerWidth - 280) + 'px';
11978  leftx = window.innerWidth - 400;
11979  left = leftx + 'px';
11980  console.log('aff-resize window.innerWidth='+window.innerWidth);
11981}
11982//window.addEventListener('resize',affresize);
11983//document.addEventListener('resize',affresize);
11984//gsh.addEventListener('resize',affresize);
11985
11986function ResetAffView(){
11987  AffViewDock.appendChild(AffView);

```

```

11988 AffView.removeAttribute('style');
11989 aff_0.removeAttribute('src');
11990 aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
11991 aff_1.removeAttribute('src');
11992 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
11993 aff_2.removeAttribute('src');
11994 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
11995 aff_3.removeAttribute('src');
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 * //</span>
12006 <!--<!-- Work } -----> -->
12007
12008
12009
12010 <!--<!-- Work { -----> -->
12011 <span id="TextCanvas_WorkCodeSpan">
12012 *
12013 <details id="TextCanvas_Section" open=""><summary id="TextCanvas_Summary">TextCanvas</summary>
12014 <!-- ----- TextCanvas // 2020-1013 Satoxts { -->
12015
12016 <details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selection</summary>
12017 <h2>Font Selection</h2>
12018 <div>
12019 <div id="FontList"></div>
12020 </div>
12021 <style>
12022 #FontList {
12023   overflow:visible;
12024   background-color:rgba(240,245,255,1.0) !important;
12025 }
12026 #FontList td {
12027   font-size:16px;
12028   padding:0px;
12029   padding-left:2px;
12030   padding-right:2px;
12031   margin:0px;
12032   line-height:1.2;
12033   border:0px;
12034 }
12035 #FontList td:hovert {
12036   background-color:#228;
12037 }
12038 #FontList tr:hovert {
12039   color:#fff;
12040   background-color:#000;
12041   xxborder:1px solid #000;
12042 }
12043 .xcourier { colr:#000; font-size:16px; font-family:courier; }
12044 .xcursive { colr:#000; font-size:16px; font-family:cursive; }
12045 .xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
12046 .xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
12047 .xmonospace { colr:#000; font-size:16px; font-family:monospace; }
12048 </style>
12049 <script>
12050 function fontstr(name,text){
12051   //tr = '<'+tr style='font-family:'+name+'>'\>\n';
12052   tr = '<'+tr style='font-family:'+name+'>'\>\n';
12053   tr += '<'+td style='font-family:Arial;font-size:12pt;>'>'+name+'<'+'/td>';
12054   tr += '<'+td data-fsty="a">'>text'+<'+'/td>';
12055   tr += '<'+td data-fsty="b">'>'+b'+text'+<'+'/b'+<'+'/td>';
12056   tr += '<'+td data-fsty="i">'>'+i'+text'+<'+'/i'+<'+'/td>';
12057   tr += '<'+td data-fsty="b">'>'+b'+<'+i'+text'+<'+'/i'+<'+'/b'+<'+'/td>';
12058   tr = '<'+tr';
12059   return tr;
12060 }
12061
12062 function lsfont(){
12063   text = 'GShell-G0012';
12064
12065   fl = '';
12066   fl += <table>\n'
12067   fl += fontstr('Arial',text);
12068   fl += fontstr('Courier',text);
12069   fl += fontstr('Georgia',text);
12070   fl += fontstr('Helvetica',text);
12071   fl += fontstr('Verdana',text);
12072   fl += fontstr('Times',text);
12073
12074   fl += fontstr('Osaka',text);
12075   fl += fontstr('Meiryo',text);
12076   fl += fontstr('YuMincho',text);
12077
12078   //fl += fontstr('Roman',text);
12079   //document.fonts.load("30px cursive");
12080   fl += fontstr('Serif',text);
12081   fl += fontstr('Sans-Serif',text);
12082   fl += fontstr('System-UI',text);
12083   fl += fontstr('Monospace',text);
12084   fl += fontstr('Cursive',text);
12085   fl += fontstr('Fantasy',text);
12086   fl += </table>\n'
12087
12088   if( false ){
12089     fss = document.fonts.entries(); // FontFaceSet
12090     console.log('FSS'+fss);
12091     while( true ){
12092       font = fss.next();
12093       if( font.done ){
12094         break;
12095       }
12096       fl += font.value[0] + '<br>';
12097     }
12098   }
12099   FontList.innerHTML = fl;
12100 }
12101 function selectFont(e){
12102   t = e.target;
12103   let fsty = '';
12104   for( i = 0; i < 4; i++ ){
12105     //console.log('FontSelect '+t.nodeName+' #' +t.id+' '+t.style);
12106     if( t.nodeName == 'TD' ){
12107       //console.log('FontSelect '+t.outerHTML);
12108       if( t.hasAttribute('data-fsty') ){
12109         fsty = t.getAttribute('data-fsty');
12110       }
12111       //console.log('FontSelect = ' + fsty);
12112     }
12113     if( t.nodeName != 'TD' ){
12114       if( t.style != '' ){
12115         if( t.style.fontFamily != '' ){
12116           break;
12117         }
12118       }
12119     }
12120     t = t.parentNode;
12121   }
12122   if( t.style != '' ){
12123     font = t.style.fontFamily;
12124     //console.log('FontSelect: '+font);
12125     //console.log('FontSelect == '+ fsty);
12126     if( font != '' ){
12127       sel = document.getElementById("TextCanvas_1_Font");
12128       if( sel != null ){
12129         if( fsty != '' ){
12130           TextCanvas_1_Bold.checked = 0 <= fsty.indexOf('b');
12131           TextCanvas_1_Italic.checked = 0 <= fsty.indexOf('i');
12132         }
12133         sel.value = font;
12134         RedrawTextCanvas();
12135       }else{
12136         alert('Event: ' + e.target.nodeName + ' #' + font);
12137       }
12138     }
12139   }
12140 }
12141 FontList.addEventListener('click',selectFont);
12142 document.fonts.onloadingdone = function(fsse){
12143   //alert('font-loaded '+fsse.fontfaces.length);
12144 }
12145 function FontList_Setup(){
12146   if( FontSelect_Summary.open ){
12147     lsfont();
12148   }
12149 }

```

```

12155 FontSelect_Summary.addEventListener('click',lsfont);
12156 </script>
12157 </details>
12158
12159 <h2>Drawing Text on Canvas</h2>
12160 <!-- 2020-1012 -- Drawing Text on Canvas // SatoxIFS ( -->
12161 <div id="TextCanvas_1_Panel" class="CanvasLabel">
12162 <input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
12163 <input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
12164 <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
12165 <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">Bold
12166 <input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">Italic
12167 <br>
12168 <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12169 <input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">
12170 <!-- to be PBlue series ? -->
12171 <p>
12172 <input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
12173 </p>
12174 <div class="CanvasLabel">
12175 <input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
12176 <input id="TextCanvas_1_Clear" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
12177 <input id="TextCanvas_1_ToJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
12178 <input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
12179 <div class="CanvasImage"><br><img id="TextCanvas_1_Image" class="CanvasImage" src="">(inline image)</div>
12180 <div id="TextCanvas_1_BgImage" class="CanvasImage"><br>(background image)</div>
12181 (Data URL)
12182 <div id="TextCanvas_1_DataUrlView" class="DataURLView"><span id="TextCanvas_1_DataUrlText"></span></div>
12183 </div>
12184 <style>
12185 .CommandUsageText {
12186   font-family:Courier New;
12187 }
12188 .TextCanvas {
12189   border:1px solid #000;
12190   resize:both;
12191   display:inline !important;
12192 }
12193 .DataURLView {
12194   width:100%;
12195   font-size:10pt;
12196   font-family:Courier New, monospace;
12197   color:#000;
12198   xbackground-color:#eee;
12199   margin-bottom:10px;
12200   xdisplay:block;
12201   xoverflow:scroll;
12202 }
12203 .CanvasImage {
12204   border:1px dashed #000;
12205 }
12206 .TextCanvasText {
12207   font-size:12pt;
12208   width:100%;
12209 }
12210 .CanvasLabel {
12211   font-size:10pt;
12212   color:#000;
12213 }
12214 .CanvasImage {
12215   width:100%;
12216   height:160px;
12217   margin-bottom:10px;
12218   color:rgba(32,160,32,0.5);
12219   text-shadow:3px 3px #eee;
12220   background-color:#eee;
12221   xborder:1px solid #000;
12222   font-size:18pt;
12223   vertical-align:middle;
12224 }
12225 .PanelRadio {
12226   font-size:12pt !important;
12227   color:#000 !important;
12228   vertical-align:middle;
12229 }
12230 .CanvasBox {
12231   vertical-align:middle;
12232   margin-left:4px !important;
12233   margin-right:2px !important;
12234 }
12235 .CanvasPanel {
12236   vertical-align:middle !important;
12237   height:14pt !important;
12238   width:inherit !important;
12239   padding:2px !important;
12240   margin:4px !important;
12241   margin-left:4px !important;
12242   margin-right:2px !important;
12243   font-size:10pt !important;
12244   font-family:Georgia !important;
12245   color:#000;
12246   display:inline !important;
12247 }
12248 .TextCanvasPanel {
12249   vertical-align:middle;
12250   font-size:10pt !important;
12251   font-family:Georgia !important;
12252   color:#000;
12253   display:inline !important;
12254 }
12255 .PanelButton {
12256   font-size:10pt !important;
12257   font-family:Georgia !important;
12258   vertical-align:middle;
12259   width:45pt !important;
12260   height:14pt !important;
12261   line-height:1.2 !important;
12262   padding:3px !important;
12263   margin:1px !important;
12264   display:inline !important;
12265   padding:3px !important;
12266   color:#fff !important;
12267   background-color:#228 !important;
12268 }
12269 </style>
12270 <script>
12271 function DrawTextCanvas(){
12272   ctx = TextCanvas_1.getContext('2d');
12273   var textfont = "";
12274   if (TextCanvas_1_Italic.checked ) textfont += ' italic';
12275   if (TextCanvas_1_Bold.checked ) textfont += ' bold';
12276   textfont += ' '+TextCanvas_1_Size.value+'px';
12277   textfont += ' '+TextCanvas_1_Font.value;
12278   //ctx.font = 'italic bold 64px Georgia';
12279   //console.log('TxFont='+textfont);
12280   ctx.fillStyle = TextCanvas_1_Color.value; // #22a;
12281   ctx.font = textfont;
12282   ctx.fillText(TextCanvas_1_Text.value,10,80);
12283 }
12284 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12285 function ClearTextCanvas(){
12286   cv = TextCanvas_1;
12287   ctx = cv.getContext('2d');
12288   ctx.clearRect(0,0,cv.width,cv.height);
12289 }
12290 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12291 function RedrawTextCanvas(){
12292   ClearTextCanvas();
12293   DrawTextCanvas();
12294 }
12295
12296 function ab2str(buf) {
12297   return String.fromCharCode.apply(null, new Uint16Array(buf));
12298 }
12299
12300 // 2020-1024, canvas to image
12301 function CanvasToImage(){
12302   canvas = TextCanvas_1;
12303   // https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
12304   if (TextCanvas_1_ToPNG.checked ){
12305     url = canvas.toDataURL("image/png");
12306   }else{
12307     url = canvas.toDataURL("image/jpeg",1.0);
12308   }
12309   //alert('CanvasToImage: length='+url.length+'\n'+url);
12310   TextCanvas_1_Image.src = url;
12311   TextCanvas_1_BgImage.style.backgroundImage = 'url('+url+')';

```

```

12312 if( TextCanvas_1_DataURL_checked ){
12313 //TextCanvas_1_DataURLView.InnerHTML = url;
12314 txa = TextCanvas_1_DataURLText;
12315 utxa = document.createElement('textarea');
12316 utxa.id = "TextCanvas_1_DataURLText";
12317 utxa.style.width = '100%';
12318 utxa.style.height = '50pt';
12319 utxa.value = url;
12320 txa.parentNode.replaceChild(utxa,txa);
12321 }
12322 return TextCanvas_1_Image;
12323 }
12324 var image = new Image();
12325 image.src = url;
12326 urla = str2ab(url);
12327 blob = new Blob(urla, {type: 'text/plain'});
12328 link = document.createElement('a');
12329 link.href = URL.createObjectURL(blob);
12330 link.download = 'character.txt';
12331 link.click();
12332 return image;
12333 }
12334 TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12335 }
12336 if( TextCanvas_Section.open ){
12337 DrawTextCanvas();
12338 }
12339 TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
12340 </script>
12341 <!-- } -->
12342
12343 <script>
12344 //TextCanvas_1_Panel.addEventListener('mouseover',OffGJShell);
12345 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
12346 </script>
12347 <!-- Clicking the textarea is necessary to see upto the end of text. why? -->
12348 <input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12349 <input id="TextCanvas_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12350 <input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12351 <span id="TextCanvas_WorkCodeView"></span>
12352 <script id="TextCanvas_WorkScript">
12353 function TextCanvas_openWorkCodeView(){
12354 function TextCanvas_showWorkCode(){
12355 showHtmlCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
12356 }
12357 TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
12358 }
12359 TextCanvas_openWorkCodeView(); // should be invoked by an event
12360 </script>
12361 </details>
12362 <!-- TextCanvas_WorkCodeSpan } -->
12363 /* </span>
12364 <!-- ===== Work } ===== -->
12365
12366
12367 <!-- ===== Work { ===== -->
12368 <span id="Shading_WorkCodeSpan">
12369 /*
12370 <details><summary>Shading Canvas</summary>
12371 <!-- ===== Shading Canvas // 2020-1011 SatoxITS // -->
12372 <h2>Shading Canvas</h2>
12373 <note class="CommandUsageText">
12374 <p>Command:</p>
12375 Placement Node:<br>
12376 a ... apply (into absolute position)<br>
12377 j ... bring down (ArrowDown)<br>
12378 k ... bring up (ArrowUp)<br>
12379 h ... bring left (ArrowLeft)<br>
12380 l ... bring right (ArrowRight)<br>
12381 0 ... z-index = 0<br>
12382 + ... z-index += 1<br>
12383 - ... z-index -= 1<br>
12384 r ... return to here (relative position)<br>
12385 c ... clear the log text<br>
12386 Note: the HTML text must be contenteditable to catch Key Event.<br>
12387 </note>
12388
12389
12390 <div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
12391 <div id="Shading_1_html" class="ShadingHtml" draggable="true"></div>
12392 <div id="Shading_1_Log" class="ShadingLog" draggable="true" style=""></div>
12393
12394 <div id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</div>
12395 </style>
12396 ShadingPlate {
12397 z-index:0;
12398 position:static;
12399 overflow:scroll;
12400 display:block;
12401 width:100%;
12402 height:400px;
12403 font-size:9pt;
12404 font-family:Courier New;
12405 border:1px dashed #000;
12406 color:#444;
12407 }
12408 ShadingLog {
12409 z-index:0;
12410 position:relative;
12411 display:block;
12412 top:0px;
12413 left:0px;
12414 overflow:scroll;
12415 width:100%;
12416 font-size:9pt;
12417 font-family:Courier New;
12418 color:#666;
12419 overflow:scroll;
12420 background:rgba(200,255,200,0.4);
12421 height:400px;
12422 }
12423 ShadingHtml {
12424 z-index:2;
12425 position:relative;
12426 display:block;
12427 top:0px;
12428 left:0px;
12429 overflow:scroll;
12430 width:100%;
12431 font-size:12pt;
12432 font-family:Courier New;
12433 color:#666;
12434 overflow:scroll;
12435 background:rgba(200,255,200,0.4);
12436 height:400px;
12437 }
12438 ShadingCanvas {
12439 z-index:3;
12440 position:relative;
12441 xdisplay:block;
12442 top:0px;
12443 left:100px;
12444 resize:both;
12445 border:1px solid #000;
12446 }
12447 </style>
12448 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12449 <input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12450 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12451 <span id="Shading_WorkCodeView"></span>
12452 <script id="Shading_WorkScript">
12453 function Shading_openWorkCodeView(){
12454 function Shading_showWorkCode(){
12455 showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
12456 }
12457 Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
12458 }
12459 const BR = '<+><br>';
12460 Shading_openWorkCodeView(); // should be invoked by an event
12461 function sh_onClick(e){
12462 Shading_1_Log.InnerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
12463 + ' offset('+e.offsetX+', '+e.offsetY+')'
12464 + ' client('+e.clientX+', '+e.clientY+')'
12465 + ' page('+e.pageX+', '+e.pageY+')'
12466 + ' screen('+e.screenX+', '+e.screenY+')'
12467 + BR;
12468 e.stopPropagation();
12469 e.preventDefault();
12470 }
12471 function sh_onKeyUp(e){
12472 if( Shading_1.style.zIndex == '' ){
12473 Shading_1.style.zIndex = 0;

```

```

12474     }
12475     zi = parseInt(Shading_1.style.zIndex);
12476
12477     if( e.key.length == 1 ){
12478       Shading_1.html.innerHTML += e.key;
12479     }
12480
12481     if( e.key == '0' ){ zi = 0; }else
12482     if( e.key == '1' ){ zi = 1; }else
12483     if( e.key == '-' ){ zi = -1; }else
12484     if( e.key == 'c' ){
12485       Shading_1_Log.innerHTML = '';
12486     }else
12487     if( e.key == 'r' ){
12488       Shading_1.style.position = "relative";
12489       Shading_1.style.top = '0px';
12490       Shading_1.style.left = '0px';
12491       zi = 0;
12492     }else
12493     if( e.key == 'j' || e.code == 'ArrowDown' ){
12494       topx = parseInt(Shading_1.style.top) + 50;
12495       Shading_1.style.top = topx + 'px';
12496     }else
12497     if( e.key == 'k' || e.code == 'ArrowUp' ){
12498       topx = parseInt(Shading_1.style.top) - 50;
12499       Shading_1.style.top = topx + 'px';
12500     }else
12501     if( e.key == 'l' || e.code == 'ArrowRight' ){
12502       lefty = parseInt(Shading_1.style.left) + 50;
12503       Shading_1.style.left = lefty + 'px';
12504     }else
12505     if( e.key == 'h' || e.code == 'ArrowLeft' ){
12506       lefty = parseInt(Shading_1.style.left) - 50;
12507       Shading_1.style.left = lefty + 'px';
12508     }else
12509     if( e.key == 'a' ){
12510       Shading_1.style.position = "absolute";
12511       Shading_1.style.top = '0px';
12512       Shading_1.style.left = '0px';
12513     }else{
12514     }
12515     Shading_1.style.zIndex = zi;
12516     Shading_1_Log.innerHTML += 'Keypup..' + e.target.nodeName + '#' + e.target.id
12517     + 'Up(' + e.key + '/' + e.code + ')';
12518     + 'z-index:' + zi + '/' + Shading_1.style.zIndex
12519     + 'top:' + Shading_1.style.top
12520     + BR;
12521     e.stopPropagation();
12522     e.preventDefault();
12523   }
12524   function sh_onKeyDown(e){
12525     Shading_1_Log.innerHTML += 'Keydown' + e.target.nodeName + '#' + e.target.id
12526     + 'Down(' + e.key + '/' + e.code + ')'+BR;
12527     e.stopPropagation();
12528     e.preventDefault();
12529   }
12530 }
12531 function Shading_Setup(){
12532   Shading_1_Log.innerHTML += '<'+>Click here<'+>/>';
12533   Shading_1.addEventListener('keydown',sh_onKeyDown);
12534   Shading_1.addEventListener('keyup',sh_onKeyUp);
12535   Shading_1.addEventListener('click',sh_onClick);
12536   Shading_1.addEventListener('click',sh_onClick);
12537
12538   Shading_1_Log.style.top = "-400px";
12539   Shading_1_Log.style.left = "200px";
12540
12541   Shading_1.appendChild(Shading_1_Canvas);
12542   Shading_1_Canvas.style.width = "300px";
12543   Shading_1_Canvas.style.height = "300px";
12544   Shading_1_Canvas.style.position = "relative";
12545   Shading_1_Canvas.style.top = "-750px";
12546   Shading_1_Canvas.style.left = "100px";
12547
12548   const ctx = Shading_1_Canvas.getContext('2d');
12549   ctx.fillStyle = 'rgba(160,0,0,0.9)';
12550   ctx.fillRect(50,50,40,40);
12551   ctx.fillStyle = 'rgba(0,160,0,0.9)';
12552   ctx.fillRect(60,60,40,40);
12553   ctx.fillStyle = 'rgba(0,0,160,0.9)';
12554   ctx.fillRect(70,70,40,40);
12555 }
12556 function Reset_ShadingCanvas(){
12557   Shading_1_Log.removeAttribute('style');
12558   Shading_1_Log.innerHTML = '';
12559   Shading_1_Canvas.style = "";
12560   //Shading_1_Canvas.removeAttribute('style');
12561 }
12562 </script>
12563 </details>
12564 <!-- Shading_WorkCodeSpan -->
12565 <!-- />
12566 <!-- Work -->
12567
12568 <!-- Work { -->
12569 </span id="Charmap_WorkCodeSpan">
12570 />
12571 <details id="Charmap_Work">summary>Character Map Mandala</summary>
12572 <!-- UnicodeCharmap // 2020-1015 SatoxITS -->
12573 <h2>Unicode Character Map</h2>
12574 <not>code 0x0000 - 0xFFFF / 16px / 3200 x 3200 px / zoom:0.25</not>
12575 <div id="Charmap_1_Frame">
12576 <div id="Charmap_1_Text" class="Charmap">
12577 </div>
12578 </div>
12579 <br>
12580 <style>
12581 #Charmap_1_Frame {
12582   overflow:scroll;
12583   height:3200px; width:3200px;
12584   xtransform:scale(0.5);
12585   zoom:0.25;
12586   resize:both;
12587   background-color:#fff;
12588 }
12589 .Charmap {
12590   xzoom:0.25;
12591   font-size:16px;
12592   line-height:1.0;
12593   xfont-family:Georgia;
12594   color:#000;
12595 }
12596 </style>
12597 <script>
12598 function charmapgen(){
12599   text = '';
12600   for( cc = 0; cc < 0x10000; cc++){
12601     text += String.fromCharCode(cc);
12602   }
12603   Charmap_1_Text.innerHTML = text;
12604 }
12605 Charmap_Work.addEventListener('click',charmapgen);
12606 //charmapgen();
12607 </script>
12608 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12609 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12610 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12611 <span id="Charmap_WorkCodeView"></span>
12612 <script id="Charmap_WorkScript">
12613 function Charmap_openWorkCodeView(){
12614   function Charmap_showWorkCode(){
12615     showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12616   }
12617   Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCode);
12618 }
12619 Charmap_openWorkCodeView(); // should be invoked by an event
12620 </script>
12621 </details>
12622 <!-- Charmap_WorkCodeSpan -->
12623 <!-- />
12624 <!-- Work -->
12625
12626 <!-- Work { -->
12627 </span id="Pointillism_WorkCodeSpan">
12628 />
12629 <details>summary>Collaborated Pointillism</summary>
12630 <!-- CollaboratedPointillism // 2020-1016 SatoxITS -->
12631 <h2><a name="Pointillism" class="Pointillism"></a><a href="#Pointillism">Collaborated Pointillism</a></h2>

```

```

12636
12637<input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
12638<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
12639<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
12640<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
12641<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
12642<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Save">
12643<input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Load">
12644<div id="Pointillism_1" class="Pointillism">
12645
12646<span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
12647<span id="Pointillism_1_XY_1" class="Pointillism_XY">XY1</span>
12648<span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
12649<canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12650</span>
12651
12652<span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
12653<span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
12654<span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
12655<canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12656</span>
12657
12658</div>
12659<br>
12660
12661<style>
12662.Pointillism {
12663  xdisplay:block;
12664  resize:both;
12665  width:600px;
12666  height:380px;
12667  min-width:240px;
12668  min-height:270px;
12669  background-color:#eee;
12670  overflow:scroll;
12671  font-size:16px;
12672  font-family:Georgia;
12673  color:#000;
12674  vertical-align:middle;
12675}
12676.Pointillism_Unit {
12677  position:relative;
12678  top:0px;
12679  display:block;
12680  overflow:scroll;
12681  width:300px;
12682  height:350px;
12683  margin:5px;
12684  padding:10px;
12685  background-color:rgba(255,255,127,0.7);
12686}
12687.Pointillism_XY {
12688  display:block;
12689  vertical-align:middle;
12690  width:290px;
12691  xxheight:20px;
12692  font-size:12px;
12693  line-height:1.2;
12694  padding:5px;
12695  margin:0px;
12696  color:#fff;
12697  background-color:#44c;
12698}
12699.Pointillism_XY_Remote {
12700  display:block;
12701  vertical-align:middle;
12702  width:290px;
12703  xxheight:20px;
12704  font-size:12px;
12705  line-height:1.2;
12706  padding:5px;
12707  color:#fff;
12708  background-color:#4a4;
12709}
12710.Pointillism_Canvas {
12711  display:block;
12712  position:relative;
12713  xpadding:20px;
12714  xleft:20px;
12715  xtop:20px;
12716  background-color:#333;
12717}
12718</style>
12719<script>
12720var points = [];
12721var replay = [];
12722var replayx = 0;
12723function pClearCanvas(can){
12724  ctx = can.getContext('2d');
12725  ctx.clearRect(0,0,can.width,can.height);
12726}
12727function Pointillism_1_ClearCanvas(){
12728  pClearCanvas(Pointillism_1_Canvas_1);
12729  pClearCanvas(Pointillism_1_Canvas_2);
12730}
12731function PointsReset(){
12732  points = [];
12733  replay = [];
12734  inRepeat = false;
12735  inReplay = false;
12736  Pointillism_1_ClearCanvas();
12737}
12738function Pointillism_1_ResetCanvas(){
12739  PointsReset();
12740  if( Pointillism_1_Share.checked ){
12741    //alert('---broadcast reset\n');
12742    GJ_BcastMessage('DRAW RESET');
12743  }
12744}
12745function Pointillism_1_ResetCanvasReceive(){
12746  //alert('---received reset\n');
12747  PointsReset();
12748}
12749function drawPoint(can,x,y,r,g,b){
12750  const ctx = can.getContext('2d');
12751  ctx.fillStyle = 'rgba('+r+','+g+','+b+',0.7)';
12752  ctx.fillRect(x,y,8,8);
12753}
12754function waitMs(serno,ms){
12755  console.log('-- wait #' +serno+ ' '+ms+'ms');
12756  until = new Date();
12757  now = until.getTime();
12758  untilMs = now + ms;
12759  for( wi = 0; ; wi++){
12760    now = new Date();
12761    nowMs = now.getTime();
12762    remMs = untilMs - nowMs;
12763    //console.log('wait '+wi+' '+remMs+'/'+ms);
12764    if( remMs < 0 ){
12765      break;
12766    }
12767  }
12768}
12769var inReplay = false;
12770function replay(){
12771  rx = replayx;
12772  if( replay.length <= rx ){
12773    return;
12774  }
12775  replayx ++;
12776  pi = replay[rx];
12777  if( pi[1] == 1 ){
12778    can = Pointillism_1_Canvas_1;
12779  }else{
12780    can = Pointillism_1_Canvas_2;
12781  }
12782  drawPoint(can,pi[2],pi[3],pi[4],pi[5],pi[6]);
12783  if( inReplay == false ){
12784    console.log('wait '+replayx+' Stopped');
12785    return;
12786  }
12787  if( rx < replay.length-1 ){
12788    prevMs = replay[rx][0].getTime();
12789    nextMs = replay[rx+1][0].getTime();
12790    delayMs = nextMs - prevMs;
12791    //console.log('wait '+replayx+' '+delayMs+'ms');
12792    window.setTimeout(replay,delayMs);
12793  }else{
12794    console.log('wait '+replayx+' Finished');
12795    if( inRepeat ){
12796      window.setTimeout(repeat1,1000);
12797    }
12798  }
12799}

```

```

12798     }
12799 }
12800 function Pointillism_1_ReplayCanvas(can){
12801     Pointillism_1_ClearCanvas();
12802     replay = points;
12803     replayx = 0;
12804     inReplay = true;
12805     replayI();
12806 }
12807 var inRepeat = false;
12808 function repeatI(){
12809     Pointillism_1_ClearCanvas();
12810     replay = points;
12811     replayx = 0;
12812     replayI();
12813     if( inRepeat ){
12814         //window.setTimeout(repeatI,1000);
12815     }
12816 }
12817 function Pointillism_1_RepeatCanvas(can){
12818     if( inRepeat ){
12819         inRepeat = false;
12820         inReplay = false;
12821     }else{
12822         inRepeat = true;
12823         inReplay = true;
12824         repeatI();
12825     }
12826 }
12827 }
12828 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
12829 function Pointillism_Setup(){
12830     var moveCount1 = 0;
12831     var moveCount2 = 0;
12832 }
12833 var gjlinked = false;
12834 function GJdraw(msg){
12835     if( gjlinked == false ){
12836         //GJLink_Section.open = true;
12837         GJ_Join();
12838         gjlinked = true;
12839     }
12840     GJ_BcastMessage('DRAW '+msg);
12841 }
12842 function showXYI(e){
12843     moveCount1 += 1;
12844     x = e.offsetX;
12845     y = e.offsetY;
12846     Pointillism_1_XY_1.InnerHTML = 'XYI: '+ 'x'+x +', y'+y + ' /*'+moveCount1+'*/'+points.length;
12847     Pointillism_1_XY_2_Remote.InnerHTML = 'XYI: '+ 'x'+x +', y'+y + ' /*'+moveCount1;
12848     if( e.buttons | CopyLocal() ){
12849         points.push(new Date(),1,x,y,64,64,255);
12850         drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
12851         if( CopyLocal() ){
12852             drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
12853         }
12854         GJdraw('1','x+', 'y');
12855     }
12856 }
12857 function showXYZ(e){
12858     moveCount2 += 1;
12859     x = e.offsetX;
12860     y = e.offsetY;
12861     Pointillism_1_XY_2.InnerHTML = 'XYZ: '+ 'x'+x +', y'+y + ' /*'+moveCount2+'*/'+points.length;
12862     Pointillism_1_XY_1_Remote.InnerHTML = 'XYZ: '+ 'x'+x +', y'+y + ' /*'+moveCount1;
12863     if( e.buttons | CopyLocal() ){
12864         points.push(new Date(),2,x,y,64,255,64);
12865         drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
12866         if( CopyLocal() ){
12867             drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
12868             //GJdraw('2','x+', 'y');
12869         }
12870     }
12871 }
12872 Pointillism_1_Canvas_1.addEventListener('mousemove',showXYI);
12873 Pointillism_1_Canvas_2.addEventListener('mousemove',showXYZ);
12874 Pointillism_1_Unit_2.style.left = '340px';
12875 Pointillism_1_Unit_2.style.top = '-375px';
12876 }
12877 function Pointillism_Remotedraw(arg){
12878     //alert('Draw at '+arg);
12879     //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
12880     if( arg == 'RESET' ){
12881         Pointillism_1_ResetCanvasReceive();
12882     }else{
12883         argv = arg.split(',');
12884         x = argv[1];
12885         y = argv[2];
12886         Pointillism_1_XY_2_Remote.InnerHTML = 'XYR: '+ 'x'+x +', y'+y + ' /*'+points.length;
12887         drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
12888     }
12889 }
12890 </script>
12891 }
12892 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12893 <input id="Pointillism_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12894 <input id="Pointillism_WorkCodesSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12895 <span id="Pointillism_WorkCodeView"></span>
12896 <script id="Pointillism_WorkScript">
12897     function Pointillism_openWorkCodeView(){
12898         function Pointillism_showWorkCode(){
12899             showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
12900         }
12901         Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_showWorkCode);
12902     }
12903 }
12904 Pointillism_openWorkCodeView(); // should be invoked by an event
12905 </script>
12906 </details>
12907 <!-- Pointillism_WorkCodeSpan -->
12908 </span>
12909 </!-- ===== Work -->
12910
12911
12912
12913 </!-- ===== Work { ===== -->
12914 <span id="StatCounter_WorkCodeSpan">
12915 </span>
12916 <details><summary>StatCounter</summary>
12917 <!-- ===== StatCounter// 2020-1018 SatoxITS -->
12918 <2>StatCounter</2>
12919
12920 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
12921 (counter as image tag)</div>
12922 <style>
12923 .statcounter {
12924     vertical-align:middle;
12925 }
12926 #sc_SatoxITS {
12927     color:#000;
12928     font-size:12pt;
12929     height:30px;
12930     width:100%;
12931     background-color:#ddd;
12932 }
12933 </style>
12934 </div>
12935 </script>
12936 var sc_project=12411639;
12937 var sc_invisible=0;
12938 var sc_security="1aeb2a3a";
12939 var sc_https=1;
12940 var sc_host="https://";
12941 var scJsHost = "https://";
12942 </script>
12943 <!-- script src="https://statcounter.com/counter/counter.js" -->
12944 <!-- /script --> (counter by inline script)
12945 </div>
12946
12947 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12948 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12949 <input id="StatCounter_WorkCodesSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12950 <span id="StatCounter_WorkCodeView"></span>
12951 <script id="StatCounter_WorkScript">
12952     function StatCounter_openWorkCodeView(){
12953         function StatCounter_showWorkCode(){
12954             showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
12955         }
12956         StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
12957     }
12958 }
12959 StatCounter_openWorkCodeView(); // should be invoked by an event

```



```

12966</script>
12967
12968</details>
12969<!-- StatCounter_WorkCodeSpan -->
12970<!--</span> -->
12971</span>
12972</div>
12973</div>
12974</div>
12975</div>
12976</div>
12977</div>
12978</div>
12979</div>
12980</div>
12981</div>
12982</div>
12983</div>
12984</div>
12985</div>
12986</div>
12987</div>
12988</div>
12989</div>
12990</div>
12991</div>
12992</div>
12993</div>
12994</div>
12995</div>
12996</div>
12997</div>
12998</div>
12999</div>
13000</div>
13001</div>
13002</div>
13003</div>
13004</div>
13005</div>
13006</div>
13007</div>
13008</div>
13009</div>
13010</div>
13011</div>
13012</div>
13013</div>
13014</div>
13015</div>
13016</div>
13017</div>
13018</div>
13019</div>
13020</div>
13021</div>
13022</div>
13023</div>
13024</div>
13025</div>
13026</div>
13027</div>
13028</div>
13029</div>
13030</div>
13031</div>
13032</div>
13033</div>
13034</div>
13035</div>
13036</div>
13037</div>
13038</div>
13039</div>
13040</div>
13041</div>
13042</div>
13043</div>
13044</div>
13045</div>
13046</div>
13047</div>
13048</div>
13049</div>
13050</div>
13051</div>
13052</div>
13053</div>
13054</div>
13055</div>
13056</div>
13057</div>
13058</div>
13059</div>
13060</div>
13061</div>
13062</div>
13063</div>
13064</div>
13065</div>
13066</div>
13067</div>
13068</div>
13069</div>
13070</div>
13071</div>
13072</div>
13073</div>
13074</div>
13075</div>
13076</div>
13077</div>
13078</div>
13079</div>
13080</div>
13081</div>
13082</div>

```